



Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Koordinationsorgan für Geoinformation des Bundes

Basismodule des Bundes für «minimale Geodatenmodelle»

VERSION 1.0 / 2011-08-30

Vorbemerkung

Dieses Dokument erklärt die Basismodule des Bundes für «minimale Geodatenmodelle», CHBase. In Teil 1 ab S. 9 wird die Motivation für die Entwicklung von CHBase dargelegt worauf in Teil 2 ab S. 15 die Anwendung von CHBase in der Geodatenmodellierung ausführlich und mit Beispielen erläutert wird. Teil 3 ab S. 39 stellt die CHBase-Module technisch-formal als UML-Diagramme dar, wonach ein Anhang ab S. 45 den vollständigen INTERLIS-Code wiedergibt, das komplette Modellbeispiel abbildet und mit einer kurzen Anleitung zum praktischen Arbeiten mit einem Software-Werkzeug abschliesst.



Diese Dokumentation ist mit zahlreichen Querverweisen versehen, welche die physisch voneinander entfernt liegenden Teile gleicher Thematik einander logisch näher bringen sollen.

Inhalt

Teil 1 – Einführung

1.	Einleitung	9
1.1.	Grundlagen	9
1.2.	Vier Anwendergruppen.....	9
2.	Warum Basismodule des Bundes?	10
3.	Konzept der Basismodule des Bundes	11
3.1.	Grundkonzept	11
3.2.	Modellimport	12
3.3.	Datenmodellablage	12
4.	Einführung in das Modellbeispiel	13
4.1.	Beschreibung	13
4.2.	Skizze zum Modellbeispiel.....	14

Teil 2 – Benutzeranleitung

5.	Vorbemerkung	15
6.	Nutzung der Basismodule des Bundes in MGDM	15
7.	Objektidentifikation	16
8.	Geometrie.....	17
9.	Qualität von geometrischen Daten	20
10.	Mehrsprachigkeit: Lokalisierungs-Codes.....	21
11.	Mehrsprachigkeit: lokalisierte/mehrsprachige Texte	22
12.	Mehrsprachigkeit: Übersetzungslisten.....	23
13.	Kataloge: einfache dynamische Aufzählungen	24
14.	Kataloge: hierarchische dynamische Aufzählungen.....	28
15.	Administrative Strukturen: Codes.....	29
16.	Administrative Strukturen: Verwaltungsgliederung.....	29
17.	Nachführungsproblematik.....	32
18.	Grafikhinweise	36

Teil 3 – Modellspezifikation

19.	Modellbeschreibung, Detailspezifikation.....	39
20.	Konzeptionelles Datenmodell: UML-Klassendiagramme	39
20.1.	Lesehilfe.....	39

20.2.	Geometrie	40
20.3.	Mehrsprachigkeit: Lokalisierungs-Codes	40
20.4.	Mehrsprachigkeit: lokalisierte und mehrsprachige Texte.....	40
20.5.	Mehrsprachigkeit: Übersetzungslisten	41
20.6.	Kataloge: einfache und hierarchische dynamische Aufzählungen	41
20.7.	Administrative Einheiten: Modell- und Themenübersicht.....	41
20.8.	Administrative Einheiten: Administrative Einheiten	43
20.9.	Administrative Einheiten: Administrative Einheiten in der Schweiz.....	43
20.10.	Nachführungsproblematik	44
20.11.	Grafikhinweise.....	44

Anhang

A.	Konzeptionelles Datenmodell: INTERLIS 2.3-Code	45
	Teil 1: Geometrie	46
	Teil 2: Mehrsprachigkeit.....	48
	Teil 3: Kataloge.....	50
	Teil 4: Administrative Strukturen	52
	Teil 5: Nachführungsproblematik	55
	Teil 6: Grafikhinweise.....	56
B.	Komplettes Modellbeispiel	58
	Konzeptionelles Datenmodell: UML-Klassendiagramme	58
	Konzeptionelles Datenmodell: INTERLIS-Code.....	63
C.	MGDM modellieren mit der UML-Vorlage.....	71
	Datenmodell, Modellimport.....	71
	Datenthema, Klassen erzeugen	73
	Klassen, Vererbung von CHBase-Konstrukten, Attribute erzeugen.....	74
	Attribut-Typen aus Modulen verwenden	75
	Strukturattribute aus Modulen verwenden	76
	Modellieren von Referenzattributen.....	78
	Übersicht.....	80
	Modell prüfen und als INTERLIS-Code exportieren	80

Quellen, Dokumente

- [1] GKG (2011): *Allgemeine Empfehlungen zur Methodik der Definition «minimaler Geodatenmodelle»*. Online: <http://www.geo.admin.ch> → Geodaten → Geobasisdaten → Geodatenmodelle → Empfehlungen zur Geodatenmodellierung
- [2] GKG (2009): *Zeitplan für die Einführung der «Minimalen Geodatenmodelle»*. Online: <http://www.geo.admin.ch> → Geodaten → Geobasisdaten → Zeitplan → Zeitplan für die Einführung der «Minimalen Geodatenmodelle»
- [3] International Organization for Standardization ISO (2001): *ISO 11578:1996 Information technology – Open Systems Interconnection – Remote Procedure Call (RPC)*. International standard
- [4] International Organization for Standardization ISO (2002): *ISO 639-1:2002 Codes for the representation of names of languages – Part 1: Alpha-2 code*. International standard
- [5] International Organization for Standardization ISO (2006): *ISO 3166-1:2006 Codes for the representation of names of countries and their subdivisions – Part 1: Country codes*. International standard
- [6] KOGIS (2006): *INTERLIS 2 – Referenzhandbuch*. Online: http://www.interlis.ch/interlis2/docs23/ili2-refman_2006-04-13_d.pdf
- [7] Object Management Group OMG (2010): *OMG Unified Modeling Language™ (OMG UML), Infrastructure*. Version 2.3, OMG Document Number formal/2010-05-03
- [8] Object Management Group OMG (2010): *OMG Unified Modeling Language™ (OMG UML), Superstructure*. Version 2.3, OMG Document Number formal/2010-05-05

Abkürzungen

CHBase	Basismodule (des Bundes für minimale Geodatenmodelle)
GeolG	Bundesgesetz über Geoinformation, Geoinformationsgesetz
GeolV	Verordnung über Geoinformation, Geoinformationsverordnung
GKG	Koordinationsorgan für Geoinformation des Bundes
KOGIS	Koordination, Geoinformation und Services (Amtsbereich der swisstopo)
MGDM	Minimales Geodatenmodell
NGDI	Nationale Geodaten-Infrastruktur (der Schweiz)
SR	Systematische Sammlung des Bundesrechts
UML	Unified Modeling Language [7,8]

TEIL 1 – EINFÜHRUNG

1. Einleitung

1.1. Grundlagen

Die Umsetzung des Geoinformationsgesetzes (GeolG, SR 510.62) und der zugehörigen Geoinformationsverordnung (GeolV, SR 510.620) verlangen unter anderem die Definition «minimaler Geodatenmodelle» (Art. 8 GeolV). Die minimalen Geodatenmodelle (MGDM) werden so gestaltet, dass ein optimaler Datenaustausch zwischen den verschiedenen Stellen möglich wird und dass sie die im Rahmen der Nationalen Geodaten-Infrastruktur (NGDI) angestrebte Datenharmonisierung unterstützen.

In der Reihe der Hilfsmittel für die Definition der MGDM sind in erster Linie die *Allgemeinen Empfehlungen zur Definition «minimaler Geodatenmodelle»* [1] zu erwähnen. Die dort beschriebenen Empfehlungen zur Modellierungsmethodik werden durch die vorliegende Benutzeranleitung und Spezifikation der «Basismodule des Bundes» vervollständigt.

1.2. Vier Anwendergruppen

Die Basismodule des Bundes können und sollen auf unterschiedlichen Ebenen verstanden und entsprechend benutzt werden. Dazu lassen sich vier typische Anwendergruppen charakterisieren:

① *Entscheidungssträger*: Management-Ebene, wenig Fokus auf technische oder abstrakte Aspekte. Vor allem der organisatorische Rahmen von CHBase interessiert Entscheidungssträger. Allgemeine Informationen zu Konzept und Gehalt der Basismodule sind wesentlich. → [Teil 1](#) dieser Dokumentation steht im Vordergrund des Interesses.

② *Fachanwender*: Fachexperten zu bestimmten Themen, für die MGDM zu entwickeln sind. Fachanwender haben thematischen Fokus und sollen ein Grundverständnis der konzeptionellen Datenmodellierung aufweisen. Kenntnisse der Modulinhalt von CHBase sind wichtig. → [Teil 1](#) und in zweiter Linie [Teil 2](#) richten sich u.a. an Fachanwender.

③ *Modellierer*: Technische Spezialisten, die Fachanwender bei der Entwicklung der MGDM begleiten und die technisch-konzeptionelle Datenmodellierung vornehmen [1]. Detailkenntnis der Modulinhalte und deren Anwendung steht für Modellierer im Vordergrund. → Teil 2 ist für Modellierer zentral; Teil 3 eher am Rande.

④ *Modellierungsexperten*: Technische Experten, die insbesondere fundierte Kenntnisse der Sprache INTERLIS haben. Modellierungsexperten fungieren auch als technische Berater im Modellierungsprozess und entwickeln ggf. CHBase weiter. Sie sollen die Konzeption, den Aufbau und den Inhalt der CHBase-Module im Detail kennen. → Für Modellierungsexperten steht neben Teil 2 vor allem die Spezifikation in Teil 3 im Vordergrund.

Hinweis



Für das Verständnis von Teil 2 und Teil 3 dieser Dokumentation werden fortgeschrittene Kenntnisse der Sprachen UML [7,8] und INTERLIS 2 [6] vorausgesetzt.

2. Warum Basismodule des Bundes?

Eines der grossen Anliegen der NGDI und der Umsetzung von GeolG und GeolV ist die Datenharmonisierung. Die Datenharmonisierung findet auf verschiedenen Ebenen statt: rechtlich (Gesetz, Verordnungen), organisatorisch (Zeitplan für die Einführung der minimalen Geodatenmodelle [2], Bildung von Fachinformationsgemeinschaften) und technisch (minimale Geodatenmodelle, Geowebdienste, [nationale] Geodateninfrastruktur). Die letzte Ebene ist die inhaltliche Datenharmonisierung. Daten, die in der gleichen Form und mit der gleichen Bedeutung in mehreren minimalen Geodatenmodellen modelliert werden, *sollen gleich modelliert* werden. Diese gleichen Modellinhalte müssen aber nicht mehrmals neu erfunden werden, sie können dank einer einheitlichen, allgemeinen Definition wieder verwendet werden.

Die Basismodule des Bundes CHBase stellen eine Sammlung solcher einheitlicher, allgemeiner Definitionen dar. Durch die einheitliche Verwendung der Module aus CHBase wird die technische und inhaltliche Datenharmonisierung unterstützt. Die Anwendung von CHBase ist nicht zwingend im Sinne einer Norm, sondern im Sinne der Datenharmonisierung empfohlen.

Dank der vorliegenden Definition dieser allgemeinen Modellaspekte wird den Modellierern, die MGDM entwickeln, Arbeit abgenommen. Sie müssen nicht Gleiches mehrmals selbst entwickeln, sondern können die vorbereiteten Modellteile direkt verwenden.

Hinweis



In der Folge sind die Begriffe «Basismodule des Bundes», «Basismodule», «CHBase» und «CHBase-Module» gleichbedeutend zu verstehen.

Hinweis



Die Basismodule des Bundes sind öffentlich und frei verfügbar. Selbstverständlich kann CHBase als Basis für beliebige andere Geodatenmodelle ausserhalb der «minimalen Geodatenmodelle» nach GeolG/GeoIV angewendet werden.

3. Konzept der Basismodule des Bundes

3.1. Grundkonzept

Die Basismodule des Bundes sind als «Werkzeugkiste» konzipiert. Das bedeutet, dass die in CHBase definierten Teilmodelle weitgehend unabhängige *Module* darstellen, die einzeln in ein Fach- oder Anwendungsmodell (MGDM) importiert werden können Vgl. dazu Abbildung 1.



Abbildung 1 – Prinzip Werkzeugkiste

Durch den modularen Aufbau von CHBase ist es ohne weiteres möglich, später – bei gegebenem Bedürfnis – weitere Basismodule zu entwickeln. Solche Module können fachspezifischer ausgerichtet sein als die in der Grunddefinition vorbereiteten. Auf diese Weise kann im Laufe der Zeit der «Werkzeugkasten» CHBase wachsen und einen zunehmenden Beitrag zur inhaltlichen Harmonisierung der Geobasisdaten leisten.

Hinweis



Das konzeptionelle Geodatenmodell der Basismodule des Bundes ist nicht «in Stein gemeisselt». Natürlich wurde bei der Entwicklung von CHBase stark darauf geachtet, möglichst nachhaltige Module zu definieren; es wäre nicht sinnvoll, beispielsweise alle sechs Monate eine neue Version der Basismodule zu publizieren!

Gleichwohl wird es nach einiger Zeit unumgänglich sein, das eine oder andere Modul zu überarbeiten. Um eine langfristige eindeutige Zuordnung zu gewährleisten, werden die CHBase-Module mit einer Versionsnummer publiziert. So bleiben später nach einer Überarbeitung die älteren Versionen verfügbar. Bestehende MGDM werden somit nicht sofort unbrauchbar, sondern können bei einer allfälligen späteren Revision auf die neuen CHBase-Modulversionen angepasst werden.

Wenn fachliche Gründe bestehen, zusätzliche CHBase-Module zu entwickeln und diese Gründe eine solche Entwicklung aus breitem fachlichem Interesse auch rechtfertigen, soll dies koordiniert erfolgen. Zu diesem Zweck ist mit GKG/KOGIS über models@geo.admin.ch Kontakt aufzunehmen.

3.2. Modellimport

Die Module von CHBase werden durch den *Import* der entsprechenden Modelle in einem MGDM verfügbar gemacht. Analog dazu werden die vordefinierten Modelle aus dem INTERLIS 2-Referenzhandbuch [6] je nach Bedürfnis in das MGDM importiert. Vgl. dazu Abbildung 2.

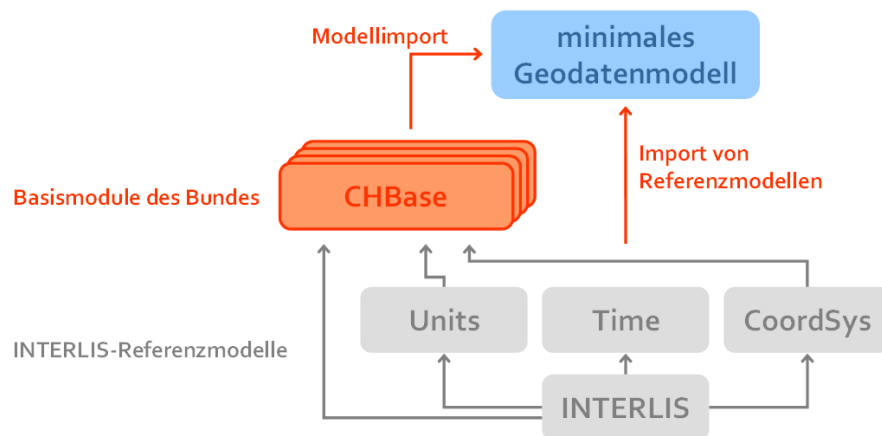


Abbildung 2 – Konzept des Modellimports in ein MGDM

Durch den Modellimport stehen die im importierten Modell definierten Elemente zur Verfügung. Die Verwendung der einzelnen Module wird in der Benutzeranleitung in Teil 2 ab S. 15 und im Anhang ab S. 45 dieser Dokumentation detailliert beschrieben.

3.3. Datenmodellablage

Im Dokument *Allgemeine Empfehlungen zur Methodik der Definition «minimaler Geodatenmodelle»* [1] wird das Prinzip und der Gebrauch der so genannten Datenmodell-Ablage erläutert.

Die Datenmodell-Ablage ermöglicht es, auf Datenmodelle als http-Ressource zuzugreifen und diese mit entsprechend befähigten Software-Werkzeugen zu nutzen, etwa zur Datenprozessierung. Der Vorteil für den Benutzer liegt darin, dass die eigentliche Modelldatei nicht lokal vorhanden sein muss – nur der Modellname muss bekannt sein – und dass immer die aktuelle Modellversion verfügbar ist. Die Basismodule des Bundes sind in der Bundes-Datenmodell-Ablage unter folgender Adresse verfügbar: <http://models.geo.admin.ch/CHBase>.

4. Einführung in das Modellbeispiel

4.1. Beschreibung

Zum klareren Verständnis der Basismodule wurde ein durchgängiges Modellbeispiel für die Benutzeranleitung in Teil 2 dieser Dokumentation konstruiert. Zu jedem beschriebenen Basismodul wird der entsprechende Aspekt im Modellbeispiel als INTERLIS 2.3-Codeschnipsel dargestellt. Im Anhang wird das Modellbeispiel zusammengesetzt und komplett als UML-Klassendiagramme (S. 58) und INTERLIS-Code (S. 63) abgebildet.

Als Beispiel wurden – stark vereinfachte – Modelle gewählt, welche das Gewässernetz und Kläranlagen inklusive Vorfluter umfasst. Die Modelle beschreiben, wer die Wasserqualität von Gewässern überprüft und wer die politische Verantwortung für eine Kläranlage hat. Für die Themen der Modelle wurden unterschiedliche Nachführungskonzepte (der Daten) gewählt.

Das fiktive Beispiel erhebt keinesfalls den Anspruch auf ausreichende und richtige Behandlung des dargestellten Fachthemas. Vielmehr ist es die Absicht, anhand dieses Beispiels die verschiedenen Elemente der Basismodule des Bundes vorzustellen.

Das **Modellbeispiel Gewässermanagement** ist konzeptionell wie folgt gegliedert:

1. Das Modell für Kartenkataloge *CHBaseEx_MapCatalogue_V1* mit dem Topic
 - *MapCatalogue_WithLatestModification*, ein Katalog mit verschiedenen Kartentypen zur grafischen Darstellung der Modelldaten.
2. Das Modell für das Gewässernetz *CHBaseEx_WaterNet_V1* mit den Topics
 - *WaterNet_Modifications* zur Spezifizierung der Nachführungsinformationen – im Sinne eines Mutationsverzeichnisses – für das nächste Topic:
 - *WaterNet_WithModificationObjects*, welches aus Quellen, Flüssen und Seen besteht und von einer Behörde hinsichtlich der Wasserqualität beaufsichtigt wird;
 - *LakeNames* zur Definition der Übersetzungslisten für Seenamen.

3. Das Modell *CHBaseEx_Sewage_V1* für das Abwasserreinigungs-Wesen, mit den Topics
 - *SewageCatalogue_WithOneState* zur Katalogisierung von Typen von Kläranlagen;
 - der Abwasserreinigung *Sewage_WithLatestModification*, die aus Kläranlagen und Vorflutern besteht und natürlich ebenfalls einer Aufsichtsbehörde unterstellt ist.

Hinweis



Das Modellbeispiel verwendet alle Basismodul-Typen, allerdings kommt nicht jede einzelne Definition aus allen Basismodulen explizit zum Einsatz.

4.2. Skizze zum Modellbeispiel

Das Modellbeispiel umfasst eine Kläranlage, eine Quelle, fünf Fließgewässer und drei Seen. Ein Fließgewässer wird nach der Kläranlage zum Vorfluter. In der Skizze sind die Modellelemente rot angeschrieben; Beziehungen sind pink dargestellt. Nicht dargestellt werden die Kataloge, die Übersetzungsliste für die Seenamen sowie die Nachführungskonzepte. Die Kartendarstellung sei implizit durch die Skizze in Abbildung 3 gegeben.

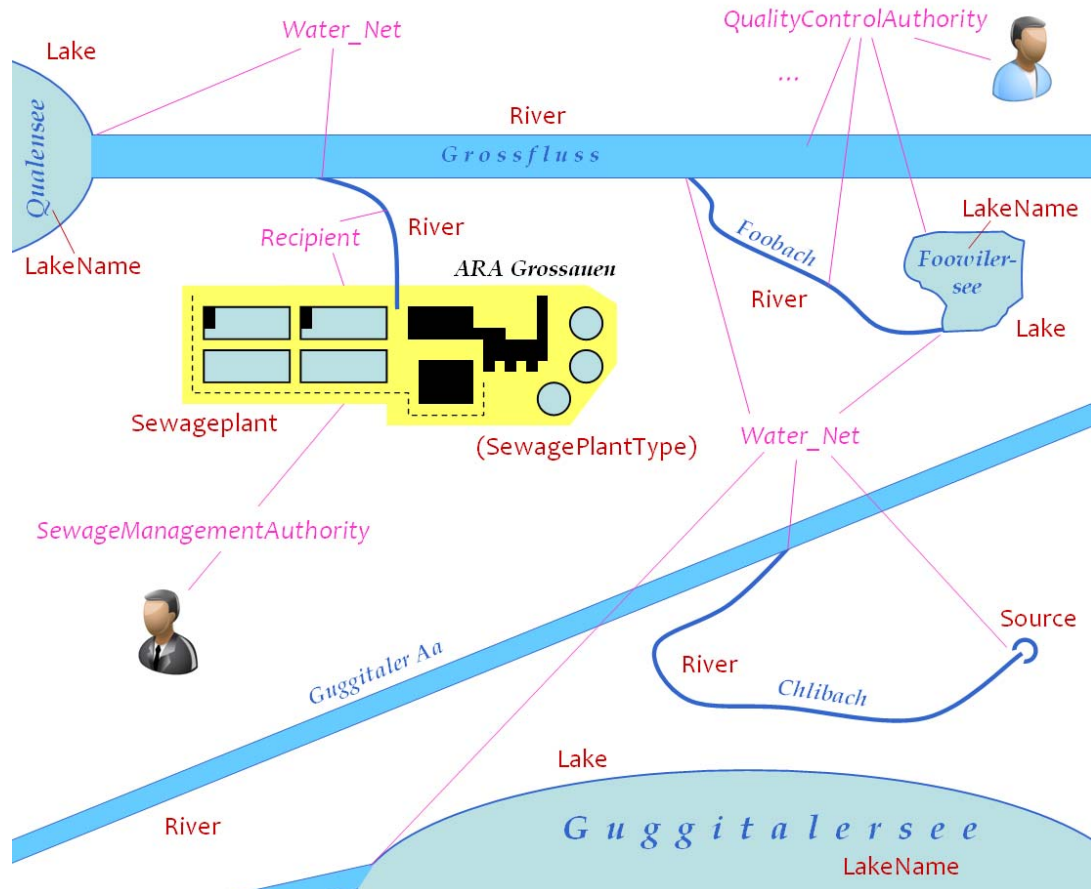






Abbildung 3 – Skizze zum Modellbeispiel «Gewässermanagement»

TEIL 2 – BENUTZERANLEITUNG

5. Vorbemerkung

Die Benutzeranleitung für die Anwendung der Basismodule des Bundes in Teil 2 dieser Dokumentation ist als «Kochbuch» aufgebaut. Jedes Modul wird zunächst thematisch eingeführt und dann wird die Verwendung dieses Moduls erklärt. Allfällige Besonderheiten oder Hinweise werden speziell herausgehoben. Schliesslich wird anhand des Modellbeispiels «Gewässermanagement» die Verwendung des entsprechenden Moduls gezeigt. Für die Strukturierung der Benutzeranleitung wird folgende Symbolik verwendet:

Thema	Verwendung	Hinweis	Modellbeispiel
			

6. Nutzung der Basismodule des Bundes in MGDM

CHBase bietet verschiedene Definitionen an, die auf unterschiedliche Weise in MGDM genutzt werden. (1) *Wertebereichsdefinitionen* («Domains») werden direkt als Attributstypen verwendet. (2) *Strukturen* können als Strukturattribute ebenfalls direkt verwendet werden. (3) *Klassenkonstrukte*, die gegebenenfalls auch Beziehungen beinhalten, sind durch Vererbung im MGDM zu konkretisieren. Für die Modellierung mit UML ist auf Anhang, ab Seite 71 verwiesen.

(1) Wertebereichsdefinitionen: direkte Verwendung

```
MODEL CHBaseModule [...] =  
  DOMAIN  
    CHBaseDomain = (a,b,c);  
END CHBaseModule.
```



```
MODEL MyModel [...] =  
  IMPORTS CHBaseModule;  
  MyClass =  
    MyAttr : CHBaseModule.CHBaseDomain;  
  END MyClass;  
END MyModel.
```

(2) Strukturdefinitionen: direkte Verwendung; ggf. auch Vererbung, siehe (3)

```
MODEL CHBaseModule [...] =  
  STRUCTURE CHBaseStruct =  
    StrAttr2 : (a,b,c);  
    StrAttr1 : TEXT*50;  
  END CHBaseStruct;  
END CHBaseModule.
```



```
MODEL MyModel [...] =  
  IMPORTS CHBaseModule;  
  MyClass =  
    MyAttr : CHBaseModule.CHBaseStruct;  
  END MyClass;  
END MyModel.
```

(3) Abstrakte (Klassen-)Konstrukte: Vererbung, Konkretisierung

```
MODEL CHBaseModule [...] =  
  CLASS CHBaseClass (ABSTRACT) =  
    Attr1 : TEXT*50;  
  END CHBaseClass;  
END CHBaseModule.
```



```
MODEL MyModel [...] =  
  IMPORTS CHBaseModule;  
  MyClass EXTENDS CHBaseModule.CHBaseClass=  
    MyAttr : (a,b,c);  
  END MyClass;  
END MyModel.
```

Schliesslich gibt es bei den nachfolgenden Beschreibungen auch «Anleitungen» oder Vorschriften, wie etwas modelliert werden soll (z.B. Objektidentifikationen oder «Multi-Area-Geometrien») obwohl in CHBase kein entsprechendes Konstrukt vorbereitet wurde. Diese Anleitungen sind in den MGDM anzuwenden.

7. Objektidentifikation

Thema



Problemstellung: «Wie soll mit Objektidentifikatoren umgegangen werden? Welche Objektidentifikation soll verwendet werden?»

Grundsätzlich muss unterschieden werden zwischen der Identifikation des Datenobjekts und der Identifikation des Realweltobjekts das durch diese Daten repräsentiert wird.

Ohne spezielle Festlegung im INTERLIS-Modell erhält jedes Datenobjekt eine *Transferidentifikation*. Eine Transferidentifikation gilt nur innerhalb eines einzelnen Datentransfers, ist also weder global eindeutig noch zeitlich stabil. Beim nächsten Transfer kann dasselbe Datenobjekt eine andere Transferidentifikation erhalten. Die Transferidentifikation ermöglicht Beziehungen zwischen den Datenobjekten innerhalb eines Transfers. Identifikationen, welche sich nicht von Transfer zu Transfer ändern, also zeitlich stabil sind, werden als *Objektidentifikatoren* (OID) bezeichnet. Die INTERLIS-Spezifikation definiert zwei Wertebereiche vor:

- Die OID-Wertebereichsdefinition *STANDARDOID* (vgl. [6], Anhang D) – braucht weniger Speicherplatz, muss aber auf jedem System korrekt konfiguriert werden. Standard-OID funktionieren nach dem «Präfix-Postfix»-Prinzip, wobei jeder Präfix von einer zentralen Stelle vergeben wird.
- Die OID-Wertebereichsdefinition *UUIDOID* nach ISO 11578 [3], braucht mehr Platz, jedoch keine Konfiguration. UUID-OID können jederzeit völlig unabhängig, global eindeutig und zeitlich stabil generiert werden. Für UUID-OID existieren verschiedene Implementierungen.

Verwendung



Zur Verwendung von stabilen und global eindeutigen OID muss man den entsprechenden Wertebereich bei der Definition eines *Modell-Topics* festlegen.

- Für die *Identifikation von Datenobjekten in MGDM* sollen UUIDOID verwendet werden, damit keine individuelle System-Konfiguration benötigt wird.
- Für die *Identifikation von Realweltobjekten* soll bei der jeweiligen Modellklasse ein entsprechendes fachlich identifizierendes Attribut modelliert werden. Der gültige Wertebereich dieses Identifikations-Attributs soll maximal 10–20 Stellen lang sein und eine Prüfziffer beinhalten. Diese Wertebereiche dürfen aber keine anderen identifizierenden Merkmale enthalten, wie etwa eine Gemeindenummer o.ä. und müssen pro Klasse eindeutig sein. Weiter ist zu regeln, wie eine Zuordnung für bestehende Datenobjekte erfolgt und wer die Vergabe für neue Realweltobjekte vornimmt. Weitere Ausführungen siehe [1].

Modellbeispiel



Die beiden Fachthemen des Modellbeispiels wenden die UUID-Definition für OID gemäss ISO 11578 an:

```
TOPIC Waternet_WithModificationObjects =  
  OID AS UUIDOID;  
  
TOPIC Sewage_WithLatestModification =  
  OID AS UUIDOID;
```

8. Geometrie

Thema



Problemstellung: «*Ich brauche geeignete Koordinatenwertebereiche und spezielle Geometrietypen, z.B. Multiflächen. Wie sieht es mit LV03 resp. LV95 aus?*»

Für geometrische Objekte bietet CHBase vordefinierte Koordinatenwertebereiche (2D und 3D) sowie eine Reihe von geometrischen Strukturen an. CHBase beinhaltet je ein Geometriemodul für die alte Landesvermessung LV03 und für die neue Landesvermessung LV95. Die beiden Module unterscheiden sich lediglich im benutzten Koordinatenbezugssystem und damit verbunden natürlich auch in den definierten Koordinatenwertebereichen.

Module: **GeometryCHLV03**, **GeometryCHLV95** ⇒ UML, S. 40 ⇒ ILI, S. 46

Definitionen:

- Koordinatenwertebereichsdefinition: *Coord2* (2D), *Coord3* (3D)

- Kurznamen für Flächen (*Surface*, *Area*), Flächen mit einer zulässigen Selbstschnitt-Pfeilhöhe ([6] Kap.2.8.12.2, Fig. 15) von 2 Millimeter (*SurfaceWithOverlaps2mm*, *AreaWithOverlaps2mm*), ungerichtete Linienzüge (*Line*), gerichtete Linienzüge (*DirectedLine*) sowie für Linienzüge mit 3D-Stützpunktkoordinaten (*LineWithAltitude*, *DirectedLineWithAltitude*)
- Orientierung in Gon/Neugrad (*Orientation*)
- Strukturen für Multilinienzüge (*MultiLine*, *MultiDirectedLine*) und Multiflächen (*MultiSurface*)

Verwendung



Die Koordinatenwertebereichsdefinitionen, Kurznamen und Strukturen, welche in den Geometriemodulen definiert sind, können in einem MGDM durch *direkte Verwendung* als Attributstypen angewendet werden.

Das gewünschte Geometriemodul wird in einem MGDM *unqualifiziert* importiert:

```
MODEL MyModel_V1 (en) [...] =
  IMPORTS UNQUALIFIED GeometryCHLV03_V1;
END MyModel_V1.
```

Damit ist man bei einem Wechsel des Bezugssystems, z.B. von LV03 auf LV95, nur gezwungen, den Importbefehl anzupassen; alle weiteren Modellteile sind davon nicht betroffen:

```
MODEL MyModel_V2 (en) [...] =
  IMPORTS UNQUALIFIED GeometryCHLV03_V1;
  TOPIC MyTopic =
    CLASS MyClass =
      Geom: Coord2;
    END MyClass;
  END MyTopic;
END MyModel_V2.
```

Diagramm: Ein grüner Checkmark-Symbol ist über dem Code eingezeichnet. Ein roter Pfeil zeigt von 'Geom: Coord2;' auf 'GeometryCHLV03_V1' und ein weiterer roter Pfeil zeigt von 'Geom: Coord2;' auf 'GeometryCHLV95_V1'. Ein rotes 'X' markiert 'GeometryCHLV03_V1'.

Falls in einem MGDM beispielsweise LV03 und LV95 parallel verwendet werden sollen, müssen beide Module qualifiziert importiert werden, damit bei der Verwendung der entsprechenden Definitionen zwischen LV03 und LV95 unterschieden werden kann:

```
MODEL MyModel (en) [...] =
  IMPORTS GeometryCHLV03_V1, GeometryCHLV95;
  TOPIC MyTopic =
    CLASS MyClass =
      Geom_Old: GeometryCHLV03_V1.Coord2;
      Geom_New: GeometryCHLV95_V1.Coord2;
    END MyClass;
  END MyTopic;
END MyModel.
```

Die Bildung von Multi-AREA-Geometrien, also von Multiflächen aus Gebietseinteilungen kann nicht genau gleich erledigt werden wie die «gewöhnlichen» Multiflächen, weil der AREA-Begriff gemäss Definition dies nicht zulässt [6]. Für Multi-AREA-Geometrien ist eine normale Modellklasse zu modellieren, welche als Geometrietyp MultiSurface verwendet. Weiter ist eine Konsistenzbedingung zu formulieren, welche für alle Objekte der entsprechenden Klasse gilt, nämlich dass alle Flächen einer Gebietseinteilung genügen. Dazu kann die von INTERLIS vordefinierte Funktion *areAreas* verwendet werden:

```
CLASS MyClass =  
  Geometry: MultiSurface;  
  SET CONSTRAINT areAreas(ALL, >> Geometry->Surfaces,  
                           >> SurfaceStructure->Surface);  
END MyClass;
```

Hinweis



In den Domainnamen zur Beschreibung von Längenmassen sollen die Begriffe «Länge», «Distanz» und «Radius» allein oder als Zusatz (nach einem Unterstrich) mit folgenden Bedeutungen verwendet werden:

- *Länge/Length*: Wert in Bezug zu einer echten oder gedachten gerichteten Linie. Bei positivem Vorzeichen in Richtung der Linie sonst in Gegenrichtung.
- *Distanz/Distance*: Abstand zwischen zwei geometrischen Orten. Kein Vorzeichen.
- *Radius*: Abstand des Kreisbogens vom Kreiszentrum. Kein Vorzeichen.

Hinweis



Nach dem gleichen Muster, wie die beiden vorbereiteten Module konstruiert sind, können weitere Koordinatenbezugssysteme implementiert werden. Grundlage dazu sind die im Referenzmodell CoordSys vordefinierten Koordinatenbezugssysteme.

Hinweis



Nach dem gleichen Muster, wie die Flächen mit zulässiger Selbstschnitt-Pfeilhöhe von 2 Millimeter definiert sind (SurfaceWithOverlaps2mm, AreaWithOverlaps2mm), können weitere Flächengeometrie-Typen mit anderen Overlap-Beträgen modelliert werden.

Hinweis



Der unqualifizierte Import des Geometriemoduls ist mit der aktuellen Version des UML/INTERLIS-Editors nicht möglich.

Modellbeispiel



Die Elemente des Gewässernetzes erhalten spezifische Geometrietypen, ebenso werden die Kläranlagen als Flächen definiert.

```
CLASS Source =  
    Position: Coord2;  
END Source;
```

```
CLASS River =  
    Geometry: DirectedLine;  
END River;
```

```
CLASS Lake =  
    Geometry: SurfaceWithOverlaps2mm;  
END Lake;
```

```
CLASS SewagePlant =  
    Site: SurfaceWithOverlaps2mm;  
END SewagePlant;
```

9. Qualität von geometrischen Daten

Thema



Problemstellung: *«Ich möchte die Qualität der Daten dokumentieren. Dabei stehen für mich die entsprechenden Eigenschaften der geometrischen Objekte im Vordergrund. Wie kann ich dabei vorgehen?»*

Generelle Angaben zur Datenqualität und -herkunft gehören zweifellos zu den Metadaten und sind damit unabhängig vom eigentlichen Modell der Fachdaten.

Detaillierte Qualitätsangaben zu einzelnen Objekten können meist nur mittels spezifischer Attribute der Fachobjekte gemacht werden. Für Anwendungen der Daten ausserhalb des engeren Fachbereichs interessiert aber häufig eine generelle Qualitätsangabe, insbesondere über die Genauigkeit der Lagegeometrie.

Module: **GeometryCHLV03**, **GeometryCHLV95** ⇒ UML, S. 40 ⇒ ILI, S. 46

Definitionen:

Um weder unnötigen Aufwand auszulösen, noch falsche Erwartungen zu wecken, wird ein Aufzähltyp *Accuracy* mit groben Genauigkeitsangaben vordefiniert:

```
Accuracy = (cm, cm50, m, m10, m50, vague);
```

Es wird also nur festgehalten welches der ungefähre Genauigkeitsbereich ist – im Sinne einer *qualitativen Grössenordnung*.

Als allgemeine Qualitätsaussage kann auch eine Aussage dienen, wie die Lage bestimmt wurde (*Method*):

```
Method = (measured, sketched, calculated);
```

- *measured*: auf Grund der Realität eingemessen und dann ins System aufgenommen;
- *sketched*: auf Grund der Realität skizziert (d.h. *nicht* gemessen) und ins System aufgenommen;

– *calculated*: einer Absicht entsprechend auf Grund anderer Objekte berechnet.

Verwendung



Die Angaben zu Genauigkeit und Methode können in einem MGDM durch *direkte Verwendung* als Attributstypen angewendet werden.

Modellbeispiel



```
CLASS Source =  
  Position: Coord2;  
  Position_Accuracy: Accuracy;  
END Source;  
  
CLASS River =  
  Geometry: DirectedLine;  
  Geometry_Accuracy: Accuracy;  
END River;  
  
CLASS Lake =  
  Geometry: SurfaceWithOverlaps;  
  Geometry_Accuracy: Accuracy;  
END Lake;
```

10. Mehrsprachigkeit: Lokalisierungs-Codes

Thema



Problemstellung: «Meine Datenmodelle sollen Mehrsprachigkeit unterstützen. Wie können Hinweise zur Sprache und zur Lokalisierung erfasst werden?»

Neben den Dateninhalten, die sprachspezifisch sein können, kann es ein Bedürfnis sein, im Datenmodell mehrsprachige oder zumindest sprachspezifische Informationen zu definieren. Mit der Kennzeichnung der Sprache oder eines Landes können Hinweise zur Lokalisierung der Modellinhalte spezifiziert werden. Dazu dient das Mehrsprachigkeitsmodul mit Lokalisierungs-Codes (sprachspezifisch = lokalisiert).

Modul: **InternationalCodes** ⇒ UML, S. 40 ⇒ INTERLIS, S. 48

Definitionen:

- Zweistellige Codes zur Identifikation der Sprache nach der Norm ISO 639-1 [4];
z.B. «de» für Deutsch (*LanguageCode_ISO639_1*);
- Dreistellige Codes zur Bezeichnung der Staaten nach der Norm ISO 3166-1 [5];
z.B. «CHE» für die Schweiz (*CountryCode_ISO_3166_1*).

Verwendung



Die Wertebereichsdefinitionen der Länder- und Sprachcodes werden *direkt* als Attributstypen verwendet.

Hinweis



Die Ländercodes werden bei den Administrativen Strukturen (S. 29) verwendet.

Um bei den Ländercodes einen Konflikt mit dem INTERLIS-Schlüsselwort «AND» (logisches Und) zu vermeiden, musste das Kürzel für Andorra mit einem Unterstrich ergänzt werden und heisst nun in CountryCode_ISO_3166_1 «AND_».

Die Sprachcodes werden bei der Definition sprachspezifischer resp. mehrsprachiger Texte (S. 22) verwendet.

11. Mehrsprachigkeit: lokalisierte/mehrsprachige Texte

Thema



Problemstellung: «*Meine Datenmodelle sollen Mehrsprachigkeit unterstützen. Wie können mehrsprachige Texte modelliert werden?*»

Neben den Dateninhalten, die sprachspezifisch (lokalisiert) sein können, kann es ein Bedürfnis sein, im Datenmodell mehrsprachige oder zumindest sprachspezifische Informationen zu definieren. Für sprachspezifische und mehrsprachige Modellinhalte stehen zwei Module zur Verfügung.

Module: **Localisation**, **LocalisationCH** ⇒ UML, S. 40 ⇒ INTERLIS, S. 48

Definitionen im Modul *Localisation*:

- Einzeiliger und mehrzeiliger sprachspezifischer Text (*LocalisedText*, *LocalisedMText*);
- Einzeiliger und mehrzeiliger mehrsprachiger Text als Sammlung («BAG OF») sprachspezifischen Textes (*MultilingualText*, *MultilingualMText*).

Das zweite Modul für die Mehrsprachigkeit, *LocalisationCH*, ist eine Erweiterung von *Localisation* und schränkt die möglichen Sprachen ein auf «de», «fr», «it», «rm» und «en».

Verwendung



Die Strukturdefinitionen für sprachspezifische resp. mehrsprachige Texte werden direkt als Attributstypen verwendet.

Hinweis



Die mehrsprachigen Texte beinhalten neben dem sprachspezifischen Text auch eine lokal gültige Konsistenzbedingung, die voraussetzt, dass in einer mehrsprachigen Textbezeichnung pro Sprache nur ein Eintrag vorkommt.

Modellbeispiel



Beschreibung des Kartentyps und des Kläranlagentyps, Name der Kläranlage sowie Name des Flusses:

```
CLASS MapKind EXTENDS CatalogueObjects_V1.Catalogues.Item =
    UserText: LocalisationCH_V1.MultilingualText;
END MapKind;

[...]

CLASS River =
    Name: LocalisationCH_V1.MultilingualText;
END River;

[...]

CLASS SewagePlantType EXTENDS CatalogueObjects_V1.Catalogues.Item =
    UserText: LocalisationCH_V1.MultilingualText;
END SewagePlantType;

[...]

CLASS SewagePlant =
    Name: LocalisationCH_V1.MultilingualText;
END SewagePlant;
```

12. Mehrsprachigkeit: Übersetzungslisten

Thema



Problemstellung: «Meine Datenmodelle sollen Mehrsprachigkeit unterstützen. Ist es möglich, Modellinhalte mehrsprachig anzeigen zu lassen? Was ist im Datenmodell dazu zu unternehmen?»

Für Listen von lokalisierten Modellinhalten, insbesondere für Anzeigetexte in Anwendungen, ist es hilfreich, wenn man Übersetzungslisten anlegen kann. Ein solcher «Diktionär» gibt Auskunft über die Sprache der entsprechenden Übersetzungsliste und beinhaltet natürlich die eigentliche Begriffsliste in der jeweiligen Sprache. Pro Sprache muss dann eine konkrete Liste als INTERLIS-XML-Datensatz angelegt werden.

Module: **Dictionaries**, **DictionariesCH** ⇒ [UML](#), S. 41 ⇒ [INTERLIS](#), S. 48

Definitionen im Modul *Dictionaries*:

- Eine Klasse *Dictionary*, die einen Sprachcode (vgl. [Lokalisierungs-Codes](#), S. 21) und eine Liste von Einträgen (*Entry*) beinhaltet.

Das zweite Modul für Übersetzungslisten, *DictionariesCH*, ist eine Spezialisierung von *Dictionaries* und schränkt die möglichen Sprachen ein auf «de», «fr», «it», «rm» und «en».

Verwendung



Die Klasse Dictionary aus dem Modul Dictionaries respektive DictionariesCH wird in einem MGDM durch Vererbung/Erweiterung angewendet und gegebenenfalls spezialisiert. Die Grunddefinition der Übersetzungslisten wurde bewusst unabhängig von anderen Klassen vorgenommen. Der Zusammenhang zu Fachobjekten kann auf verschiedene Weise hergestellt werden:

- mittels einer Referenz (*REFERENCE TO*), vergleichbar mit den Katalogobjekten (im Modellbeispiel angewendet);
- mittels eines Codeattributes (o.ä.), welches identifizierend wirkt und mit Hilfe eines (*EXISTENCE CONSTRAINT*) im Fachobjekt verlangt wird. Diese Möglichkeit wurde im Basismodul *AdministrativeUnits* (S. 29) für die Übersetzung von Ländernamen angewendet.

Modellbeispiel



Für die Bezeichnungen der Seen werden Übersetzungslisten geführt.

```
TOPIC LakeNames EXTENDS Dictionaries_V1.Dictionaries;  
DEPENDS ON DictionariesCH_V1.Dictionaries;  
  
STRUCTURE LakeName EXTENDS Entry =  
  Ref: MANDATORY REFERENCE TO (EXTERNAL)  
    WaterNet_WithModificationObjects.Lake;  
END LakeName;  
  
CLASS LakeNameTranslations EXTENDS Dictionary =  
  Entries (EXTENDED): LIST OF LakeName;  
END LakeNameTranslations;  
  
END LakeNames;
```

Jeder Listeneintrag erhält eine Referenz auf das entsprechend bezeichnete Objekt «Lake»

Zusätzlich: *XML-Datei* (S. 69) mit der konkreten Übersetzungsliste der Seennamen.

13. Kataloge: einfache dynamische Aufzählungen

Thema



Problemstellung: «Ich habe z.B. Codelisten, die ich modellieren möchte oder meine Wertelisten bestehen aus mehreren Feldern. Es ist anzunehmen, dass mehrere Leute an diesen Codelisten arbeiten. Auch mehrsprachige (Code-)Listen sind denkbar. Mit der Zeit können sich diese Listen verändern; dann will ich aber das Datenmodell nicht anpassen müssen. Wie geht das?»

Der INTERLIS-Basistyp «Aufzählung» wird verwendet, wenn ein Attribut eine vor-

definierte Anzahl von bezeichneten Werten annehmen kann [6]. Diese Aufzählungen sind statisch im Datenmodell integriert. Eine Modifikation an einer Aufzählung bedeutet eine Modelländerung, was «teuer» ist und meist vermieden werden möchte. Vgl. dazu Abbildung 4:

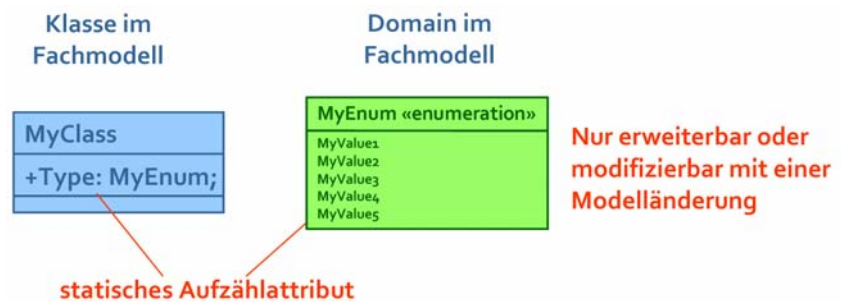


Abbildung 4 – Statische Aufzähltypen gemäss INTERLIS-Spezifikation [6]

Mit den so genannten «Katalogobjekten» bietet CHBase die Möglichkeit, Aufzählungen dynamisch zu gestalten. Die möglichen Aufzählwerte werden dabei nicht direkt im Modell sondern mittels Daten beschrieben, die einem Modell entsprechen müssen. Als Grundlage dafür wird das Basismodell *CatalogueObjects* angeboten. Die Attribute der Katalogobjekte – z.B. «Value», «Description» (auch mehrsprachig möglich!) – werden dann in einem Fachmodell als Erweiterung definiert. Die eigentlichen Katalogeinträge (also die konkreten Aufzählwerte) werden als Daten in einer INTERLIS-XML-Datei festgehalten¹. Damit Fachobjekten ein bestimmter Aufzählwert aus dem Katalog zugewiesen werden kann, wird in der entsprechenden Klasse des Fachmodells ein Referenzattribut eingefügt, das auf die Klasse der Katalogeinträge verweist.

Einfache dynamische Aufzählungen ermöglichen die Bildung von flachen Katalogen ohne Strukturierung beziehungsweise Hierarchie (S. 28).

Modul: **CatalogueObjects** ⇨ UML, S. 41 ⇨ INTERLIS, S. 50

Definitionen:

- Das abstrakte Topic Catalogues, das die Basis für alle einfachen dynamischen Aufzählungen bildet.
- Die Klasse *Item*, die einen Katalogeintrag darstellt. Die Gesamtheit der Item-

¹ Wenn ein Katalog in einem eigenen Topic oder sogar in einem separaten Modell modelliert wird, gilt bei der XML-Kodierung der Katalogobjekte, dass ein eigener Behälter (Topic-Instanz) erzeugt wird. Die Erzeugung einer separaten XML-Datei ist möglich, aber nicht zwingend.

Objekte als INTERLIS-XML-Datei bildet einen konkreten Katalog.

- Strukturen für die optionale oder zwingende Referenzierung von Katalogen (*CatalogueReference*, *MandatoryCatalogueReference*).

Verwendung



Um in einem MGDM eine dynamische Aufzählung zu erzeugen, wird das Topic Catalogues erweitert und die Klasse Item durch Vererbung konkretisiert. Dies erfolgt in einem eigenen Topic oder sogar in einem separaten Modell um den Katalog noch unabhängiger vom Rest des Datenmodells zu gestalten. Attribute für die Bezeichnung und ggf. der fachlichen Identifikation der Katalogeinträge müssen hinzugefügt werden (s. Thema und Modellbeispiel).

Ebenso werden die Strukturen für die Referenzierung des Katalogs geerbt/spezialisiert, indem das Referenzattribut in Beziehung zum eigenen, konkreten Katalog gesetzt wird. In jener Klasse des MGDM, welche die dynamische Aufzählung verwendet, wird ein Strukturattribut vom Typ der Referenzierungsstruktur eingeführt, um den eigenen Katalog verwenden zu können (s. Bsp).

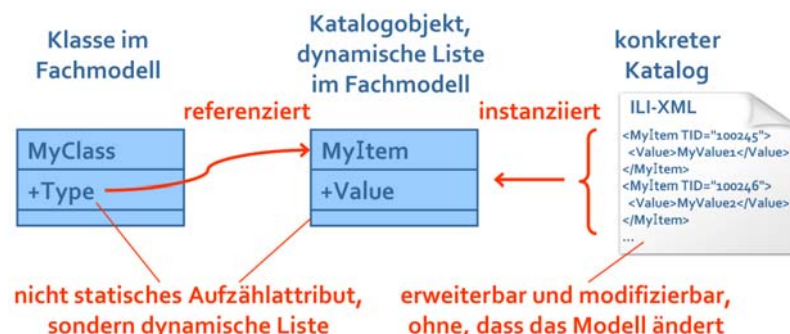


Abbildung 5 – Dynamische Aufzählungslisten

Welche konkreten Einträge zusammen einen Katalog bilden, muss festgelegt werden. Dies kann auf zwei Arten geschehen: 1. durch eine Konvention, dass jeder Katalog genau in einem Behälter kodiert wird; 2. durch die Modellierung einer identifizierenden Eigenschaft im Katalog. Die Wahl muss in jedem Fall in der Modelldokumentation [1] erläutert werden.

Hinweis



Die Strukturdefinitionen zur Referenzierung von Katalogeinträgen *CatalogueReference*, *MandatoryCatalogueReference* beinhalten je eine Konsistenzbedingung, die sicherstellt, dass nur Einträge referenziert werden, die als *IsUseable* gekennzeichnet sind, also aktuell verwendbare Einträge darstellen.

Hinweis



Obwohl die eigentlichen Aufzählwerte (respektive die Codeliste oder der Katalog) nicht im Modell drin hart kodiert sind, sondern als XML-Daten vorliegen, können die einzelnen Einträge genau angesprochen werden, etwa, um Konsistenzbedingungen zu formulieren.

Wenn z.B. in der Klasse *MyClass* ein Katalog *MyItem* referenziert wird (über *MyRef*), und der Katalogeintrag *MyValue1* im XML-Daten angesprochen werden will, wird dazu folgende Konsistenzbedingung formuliert:

«Wenn der Eintrag *Value* im Katalog *MyItem* den Wert «*MyValue1*» hat, dann...»:

```
CLASS MyItem EXTENDS Item =
  Value : MANDATORY TEXT*200;
END MyItem;

STRUCTURE MyRef
EXTENDS CatalogueReference =
  Ref : REFERENCE TO MyItem;
END MyRef;

CLASS MyClass =
  Type : MyRef;
MANDATORY CONSTRAINT Type -> Ref : Value == "MyValue1" [...];
END MyClass;
```

XML-Datensatz zum Katalog «MyItem»:
[...]
<MyItem TID="100245">
 <Value>MyValue1</Value>
</MyItem>
[...]

Modellbeispiel



Das Modellbeispiel hat zwei Kataloge, *MapKind* und *SewagePlantType*. Beide Kataloge werden genau gleich definiert. Hier als Beispiel nur *SewagePlantType*:

```
TOPIC SewageCatalogue_WithOneState
EXTENDS CatalogueObjects_V1.Catalogues =
  DOMAIN
    SewagePlantTypeValue = MANDATORY TEXT;

  CLASS SewagePlantType
  EXTENDS CatalogueObjects_V1.Catalogues.Item =
    Value: SewagePlantTypeValue;
    UserText: LocalisationCH_V1.MultilingualText;
    UNIQUE Value;
  END SewagePlantType;

  STRUCTURE SewagePlantTypeRef
  EXTENDS CatalogueObjects_V1.Catalogues.CatalogueReference =
    Reference(EXTENDED):
      MANDATORY REFERENCE TO (EXTERNAL) SewagePlantType;
  END SewagePlantTypeRef;
END SewageCatalogue_WithOneState;
[...]
```

Definition des Katalogs

Definition der Katalogreferenz

Referenzierung des Katalogs in der Modellklasse «SewagePlant»:

```
CLASS SewagePlant = [...]
Type: CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantTypeRef;
END SewagePlant;
```

Zusätzlich: XML-Datei (S. 67) mit dem konkreten Katalog der Kartentypen und XML-Datei (S. 68) mit dem konkreten Katalog der Kläranlagen-Typen.

14. Kataloge: hierarchische dynamische Aufzählungen

Thema



Problemstellung: «Für meine Zwecke genügt der INTERLIS-Aufzähltyp nicht. Ich habe z.B. Codelisten, die ich modellieren möchte oder meine Wertelisten bestehen aus mehreren Feldern. Auch mehrsprachige (Code-)Listen sind denkbar. Schliesslich möchte ich hierarchische (strukturierte) Aufzählungen modellieren. Diese Listen können ändern, ohne dass ich das Datenmodell anpassen muss!

Dynamische Aufzählungen können auch strukturierte oder hierarchische Kataloge abbilden, wenn das hier vorgestellte Modul verwendet wird. Damit können «Bäume» von Katalogeinträgen erzeugt werden. Ein Katalogeintrag kann einem anderen Eintrag untergeordnet werden. Er ist dann in der Rolle des «Kind-» oder «Untereintrags», der andere Eintrag hingegen in der Rolle des «Eltern-» oder «Obereintrags».

Modul: **CatalogueObjectTrees** ⇒UML, S. 41 ⇒INTERLIS, S. 50

Definitionen:

- Die erweiterte Klasse *Item*, die zusätzlich zwei Attribute enthält: 1) *IsSuperItem* das einen Katalogeintrag als «Ober-» oder «Elterneintrag» ausweist. Ein Eintrag kann nur dann «Obereintrag» sein, wenn *IsSuperItem* den Wert TRUE aufweist. 2) *IsUseable*, das Katalogeinträge auszeichnet (Wert TRUE), die direkt für die Zuweisung von Fachobjekten verwendbar sind. «Obereinträge» können, müssen aber nicht verwendbar sein. Nicht verwendbare «Obereinträge» sind somit reine Hierarchieknoten.
- eine Bestandteile—Ganzes-Beziehung *EntriesTree*, die die Zuordnung von «Untereinträgen» zu «Obereinträgen» darstellt.

Verwendung



Um in einem MGDM eine dynamische Aufzählung zu erzeugen, wird die Klasse *Item* durch Vererbung konkretisiert. Dies kann in einem eigenen Topic oder sogar in einem separaten Modell erfolgen. Attribute für die Bezeichnung der Katalogeinträge müssen hinzugefügt werden (s. Thema und Modellbeispiel).

Ebenso werden die Strukturen für die Referenzierung des Katalogs geerbt/spezialisiert, indem das Referenzattribut in Beziehung zum eigenen, konkreten Katalog gesetzt wird. In der Klasse des MGDM, welche die dynamische Aufzählung verwendet, wird ein Strukturattribut vom Typ der Referenzierungsstruktur

eingeführt, um den eigenen Katalog verwenden zu können (s. Beispiel).

Hinweis



Genauso wie INTERLIS-Aufzählungen können hierarchische dynamische Aufzählungen aus CatalogueObjectTrees beliebig oft geschachtelt werden.

Hinweis



Bei Katalogeinträgen, die nicht «Ober-» oder «Elterneintrag» (IsSuperItem = FALSE) sind, muss IsUsable = TRUE sein, da der Eintrag sonst keinen Sinn macht.

15. Administrative Strukturen: Codes

Thema



Problemstellung: *«Im Umfeld der Verwaltung sind immer wieder administrative Strukturen abzubilden. Dazu möchte ich die üblichen Abkürzungen resp. Kennungen für Kantone und Gemeinden verwenden.»*

Im Zusammenhang mit administrativen Strukturen werden oft bestimmte Codes verwendet. Dazu werden hier zwei Wertebereichsdefinitionen vorbereitet.

Modul: **CHAdminCodes** ⇒ UML, S. 41 ⇒ INTERLIS, S. 52

Definitionen:

- zweistellige Codes zur Identifizierung der Schweizer Kantone gemäss allgemein üblicher Abkürzung; z.B. «GL» für Glarus (*CHCantonCode*).
- die Identifikationsnummer des Bundesamtes für Statistik für politische Gemeinden in der Schweiz; die so genannte «BFS-Nummer» (*CHMunicipalityCode*).

Verwendung



Die Wertebereichsdefinitionen der Kantonscodes und BFS-Nummern werden direkt als Attributstypen verwendet.

Hinweis



Die Codes für administrative Strukturen werden für die Verwaltungsgliederung (folgender Abschnitt) verwendet.

16. Administrative Strukturen: Verwaltungsgliederung

Thema



Problemstellung: *«Im Umfeld der Verwaltung sind immer wieder administrative Strukturen abzubilden. Die Modellierung der Verwaltungsgliederung und von Zweckverbänden ist relativ komplex. Es macht Sinn, diese Gliederung einmal zu modellieren und dann wieder zu verwenden. Gibt es entsprechende Konstrukte?»*

Um die Zusammenhänge zwischen den einzelnen Ebenen der Verwaltung und speziell zu Zweckverbänden und «Stellen» darzustellen, müssen verschiedene Themen (Topics) und Beziehungen modelliert werden.

Administrative Einheiten (*AdministrativeUnit*) sind Gemeinden, Bezirke, Kantone oder Länder. Administrative Einheiten sind hierarchisch organisiert: Gemeinden gehören zu einem Bezirk oder zu einem Kanton (bei Kantonen ohne Bezirke), Bezirke zu einem Kanton und Kantone zur Schweiz. Oft werden Vereinigungen von administrativen Einheiten (*AdministrativeUnion*) für bestimmte Zwecke gebildet: als so genannte Zweckverbände oder Konkordate. Der Begriff administratives Element (*AdministrativeElement*) umfasst sowohl administrative Einheiten als auch Vereinigungen davon. Der Begriff umfasst also alle hoheitlichen Körperschaften.

Die konkreten hoheitlichen Aufgaben werden durch einzelne Stellen, also Amtsstellen, Beauftragte etc. geleistet (*Agencies*). Sie können einerseits organisatorisch (Beziehung *Authority*) und andererseits auftragsmässig (Beziehung *Organisation*) anderen *Agencies* oder administrativen Elementen zugeordnet werden. So ist beispielsweise die Abteilung «Gemeindewerke» der entsprechenden Gemeinde unterstellt, kann aber durchaus auch die operativen Tätigkeiten für einen Zweckverband ausführen.

Module: **AdministrativeUnits** ⇒ UML, S. 43, **AdministrativeUnitsCH**
⇒ UML, S. 43 ⇒ INTERLIS, S. 52

Im Modul *AdministrativeUnits* sind die abstrakten Grundlagen sowie die Konkretisierung für Länder (*Countries*) definiert. Länder werden in der Klasse *Country* (Erweiterung von *AdministrativeUnit*) mittels Länderkürzel gemäss ISO 3166-1 identifiziert. Ländernamen (*CountryNames*) können darüber hinaus mit einer Übersetzungsliste (S. 23) in mehrere Sprachen übersetzt werden (*CountryNamesTranslation* als Erweiterung von *Dictionaries*).

Das Modul *AdministrativeUnitsCH* umfasst die schweizerischen Konkretisierungen für Kantone (*CHCantons*), Gemeinden (*CHMunicipalities*), Zweckverbände (*CHAdministrativeUnion*) und Stellen (*CHAgencies*). Als Attribute sind bei allen ein mehrsprachiger Name (S. 22) und die Webadresse vorgesehen, bei Kantonen

und Gemeinden zusätzlich der entsprechende Code (Kantonskürzel resp, BFS-Nummer), bei Zweckverbänden und Stellen zusätzlich eine Beschreibung.

Verwendung



Im Basismodul AdministrativeUnits werden abstrakte Grundkonstrukte für die Verwaltungsgliederung definiert. Diese sind für die Anwendung in Fachmodellen mittels Vererbung zu konkretisieren. Daneben gibt es konkrete Konstrukte im Topic Countries, die direkt in einem Fachmodell verwendet werden können (Erinnerung: durch den Modellimport stehen diese Konstrukte zur Verfügung!)

Das Modul AdministrativeUnitsCH umfasst wie erwähnt Schweiz-spezifische Konkretisierungen, welche in Fachmodellen direkt verwendet werden können.

In manchen Fällen macht es durchaus Sinn, auch abstrakte Konstrukte direkt zu verwenden, etwa wenn man mittels einer Beziehung auf ein abstraktes Konstrukt (z.B. AdministrativeElement) über die Vererbungsstruktur alle davon abgeleiteten Definitionen referenzieren möchte. Siehe dazu das Modellbeispiel.

Hinweis



Da in diesem Bereich noch wenig vertiefte Erfahrungen vorliegen, ist dieses Basismodul als *Vorschlag* zu verstehen, wie das Anliegen in einer vereinfachten aber brauchbaren Form gelöst werden *kann*.

Erfahrungen und Probleme mit diesen Modulen sollen mitgeteilt werden unter models@geo.admin.ch.

Modellbeispiel



Im Modellbeispiel werden die Definitionen aus den Basismodulen direkt verwendet, ohne sie zu erweitern. Damit wird der Bezug zu einer Aufsichtsbehörde hergestellt:

```
ASSOCIATION QualityControlAuthority =  
  WaterSubject -- Source OR River OR Lake;  
  Controller (EXTERNAL) -- {0..1}  
    AdministrativeUnitsCH_V1.CHAgencies.Agency;  
END QualityControlAuthority;  
[...]  
  
ASSOCIATION SewageManagementAuthority =  
  Authority (EXTERNAL) -<> {1..1}  
    AdministrativeUnits_V1.AdministrativeUnits.AdministrativeElement;  
  SewagePlant -- SewagePlant;  
END SewageManagementAuthority;
```

Referenzierung auf abstraktes Basiskonstrukt → über die Vererbung werden alle davon abgeleiteten Konstrukte effektiv referenziert.

17. Nachführungsproblematik

Thema



Problemstellung: «Die Historisierung oder mindestens die Versionierung von Objektdaten ist ein sehr wichtiges Thema. Auf verschiedenen Stufen soll dies möglich sein: einfache Objektstatus-Informationen, Lösch- und Modifikationsdatum und weitere Update-Informationen. Wie wird die Objektversionierung im Datenmodell angegangen?»

Erste, vielleicht etwas provokative Aussage: *Die Fachmodelle sollen sich nicht mit der Nachführungsproblematik befassen.* Damit ist aber keineswegs gemeint, dass die Nachführung der Daten inklusive ihrer Dokumentation kein wichtiges Anliegen ist! Im Gegenteil: Es ist so wichtig und auch derart anspruchsvoll, dass sich nicht jedes Fachmodell im Detail darum kümmern soll.

Die verschiedenen Anliegen im Bereich der Nachführung werden dafür typisiert. Für jeden Nachführungstyp stellt das Basismodell die nötigen Grundlagen bereit. Unter dem Begriff «Nachführungstyp» sollen verschiedene Prinzipien verstanden werden, wie die Objektnachführung organisiert wird. Beispiele:

- Der Datenbestand beschreibt den aktuellen Zustand. Eine Änderung führt zu geänderten Daten, ohne dass irgendwelche Zusatzinformationen festgehalten werden (z.B. *wann* die Änderung erfolgte).
- Die letzte oder sogar alle erfolgten Änderungen sollen dokumentiert werden: wer, wann etc.
- Die Änderungsinformation soll nicht direkt bei den Fachobjekten sondern bei so genannten Modifikationsobjekten gespeichert werden. Fachobjekte verweisen dann auf diese Modifikationsobjekte.
- Objekte, die nicht mehr oder noch nicht gültig sind, sollen zum Datenbestand gehören können.
- Frühere oder zukünftige Versionen desselben Fachobjekts sollen zum Datenbestand gehören können.

Diese Nachführungsprinzipien sind unabhängig vom Fachmodell. Man kann sich vorstellen, dass Daten zu einem Fachmodell bei einer Stelle (z.B. bei einem Kanton) mit Objektversionierung verwaltet werden, während Kopien dieser Daten bei anderen Stellen (z.B. Bund, Ingenieurbüros) in vereinfachter Form geführt werden

(z.B. Objekte mit einem bestimmten zeitlichen Gültigkeitsbereich oder nur aktueller Zustand).

Aktuell sind folgende Nachführungstypen durch entsprechende Module vorbereitet.

Module: ***WithOneState***, ***WithLatestModification***, ***WithModification-Objects*** ⇒ UML, S. 44 ⇒ INTERLIS, S. 55

- *WithOneState*: Beschreibt nur einen bestimmten Zustand, oft den aktuellen;
- *WithLatestModification*: Bei den Objekten werden Informationen über den Beginn und das Ende der zeitlichen Gültigkeit («Lebensdauer») und die letzte Veränderung festgehalten. Es ist zulässig, dass der Gültigkeits-Beginn vorerst undefiniert ist. Ein solches Objekt bleibt bis auf weiteres ungültig, kann aber trotzdem modifiziert werden. Ist das Ende der zeitlichen Gültigkeit erreicht, machen weitere Änderungen keinen Sinn mehr. Der Informationsumfang kann durch den Fachbereich festgelegt werden;
- *WithModificationObjects*: Die Modifikationsinformationen werden auf eigenständigen Modifikations- oder Mutationsobjekten geführt. Ein Fachobjekt verweist auf jene Modifikationsobjekte, in deren Rahmen es erzeugt, verändert und gelöscht wurde. Der Informationsumfang der Modifikationsobjekte kann durch den Fachbereich festgelegt werden. Dieser Nachführungstyp kann statt *WithLatestModification* insbesondere dann gewählt werden, wenn die Dokumentation der Veränderung als Ganzes einen gewissen Umfang erreicht.

Verwendung



Themen in Fachmodellen können durch einen Namenszusatz hinsichtlich des gewählten Nachführungstyps ausgezeichnet werden:

```
TOPIC MyTopic_WithOneState =
```

Gemeinsam bei allen vorgestellten Nachführungstypen ist die Definition einer Zustands-Struktur *ModInfo*. Diese Struktur wird im Fachmodell erweitert und in den Fachmodell-Klassen als Attribut eingeführt:

```
TOPIC MyTopic_WithOneState =  
  
STRUCTURE MyModInfo EXTENDS WithOneState_V1.ModInfo =  
END MyModInfo;  
  
CLASS MyClass =  
  Modification: MyModInfo;  
  MyAttr: TEXT*30;  
END MyClass;
```

Soll nun ein Topic eines Fachmodells mit einem anderen Nachführungstyp verwendet werden, genügt es, das ganze Topic zu kopieren und die Struktur *ModInfo* auf das entsprechende Nachführungsmodul abzustützen, z.B.:

```
TOPIC MyTopic_WithLatestModification =  
  
    STRUCTURE MyModInfo EXTENDS WithLatestModification_V1.ModInfo =  
        !! allfällige weiteren Attribute  
    END MyModInfo;  
  
    CLASS MyClass =  
        Modification: MyModInfo;  
        MyAttr: TEXT*30;  
    END MyClass;
```

Beim Nachführungstyp WithModificationObjects müssen zusätzlich definiert werden:

- Das Topic der Modifikations- oder Mutationsobjekte als Erweiterung des Basistopics *Modifications*;
- Die Angabe, dass das Topic der Fachobjekte vom Topic der Modifikationsobjekte abhängig ist (DEPENDS ON).

```
TOPIC MyModifications EXTENDS WithModificationObjects_V1.Modifications;  
  
    CLASS MyModification  
        EXTENDS WithModificationObjects_V1.Modifications.Modification =  
    END MyModification;  
  
END MyModifications;  
  
TOPIC MyTopic_WithModificationObjects =  
    DEPENDS ON MyModifications;  
  
    STRUCTURE MyModInfo EXTENDS WithModificationObjects_V1.ModInfo =  
    END MyModInfo;  
  
    CLASS MyClass =  
        Modification: MyModInfo;  
        MyAttr: TEXT*30;  
    END MyClass;
```

Hinweis



Auf die Definition eines Nachführungstyps mit mehreren Zuständen desselben Objektes (z.B. Projektvarianten) wurde vorläufig verzichtet. Wenn verschiedene Objektversionen einfach als selbstständige Datenobjekte desselben Realweltobjektes aufgefasst würden, ergeben sich daraus Probleme im Umgang mit Beziehungen. Durch das neu entstandene Objekt müsste auch ein neues Beziehungselement erzeugt werden, woraus ein erheblicher Koordinationsaufwand entstünde. Eine Lösung dafür ist noch pendent.

Modellbeispiel



Im Modellbeispiel wurden bewusst alle verschiedenen Arten der Statusinformation modelliert. Speziell zu beachten ist, dass für *WithModificationObjects* zunächst die Klasse *Modification* spezialisiert wird.

```
TOPIC MapCatalogue_WithLatestModification [...] =  
    STRUCTURE ModInfo EXTENDS WithLatestModification_V1.ModInfo =  
    END ModInfo;  
  
    CLASS MapKind [...] =  
        Modification: ModInfo;  
        [...]  
    END MapKind;  
[...]  
  
TOPIC WaterNet_Modifications  
EXTENDS WithModificationObjects_V1.Modifications =  
    CLASS Modification (EXTENDED) =  
        ModificationTime(EXTENDED): MANDATORY XMLDateTime;  
        Description: MANDATORY TEXT;  
        ExecutedBy: MANDATORY TEXT;  
    END Modification;  
END WaterNet_Modifications;  
  
TOPIC WaterNet_WithModificationObjects = [...]  
DEPENDS ON [...]WaterNet_Modifications;  
  
    STRUCTURE ModInfo EXTENDS WithModificationObjects_V1.ModInfo =  
    END ModInfo;  
  
    CLASS Source = Modification: ModInfo; [...] END Source;  
    CLASS River = Modification: ModInfo; [...] END River;  
    CLASS Lake = Modification: ModInfo; [...] END Lake;  
[...]  
  
TOPIC SewageCatalogue_WithOneState [...] =  
    STRUCTURE ModInfo EXTENDS WithOneState_V1.ModInfo =  
    END ModInfo;  
  
    CLASS SewagePlantType [...] =  
        Modification: ModInfo;  
    END SewagePlantType;  
[...]  
  
TOPIC Sewage_WithLatestModification =  
    STRUCTURE ModInfo EXTENDS WithLatestModification_V1.ModInfo =  
    END ModInfo;  
  
    CLASS SewagePlant =  
        Modification: ModInfo; [...]  
    END SewagePlant;  
[...]
```

Spezialisierte Anwendung
der «Modification»

18. Grafikhinweise

Thema



Problemstellung: «*Ich möchte in meinen Datenmodellen Eigenschaften definieren, die die grafische Darstellung der Daten unterstützen; v.a. zur Ausrichtung. Wie sollen solche Grafikhinweise modelliert werden?*»

Grundsätzlich sollen die Fachmodelle keine Grafikinformatoren wie Signaturen, Farben usw. enthalten. Präsentationen wie Listen, Karten oder Pläne sollen mittels dafür geeigneten Programmen mit zusätzlichen Grafikdefinitionen aus den Daten produziert werden können.

Mit dem heutigen Stand der Technik kann mit diesem Grundsatz aber meist keine genügende Grafik erzeugt werden. Insbesondere können Symbole und Texte kaum geeignet platziert und ausgerichtet werden. Die Module *GraphicCHLV03* und *GraphicCHLV95* (analog zu Geometriedefinitionen für LV03 und LV95) bieten dafür einfache Grundlagen an.

Module: **GraphicCHLV03**, **GraphicCHLV95** ⇒ UML, S. 44 ⇒ INTERLIS, S. 56

Definitionen in beiden Modulen:

- Die Struktur *SymbolGraphic* enthält ein Attribut «*Orientation*» zur Ausrichtung der Symbole (Signaturen);
- Die Struktur *TextGraphic* enthält eine Position, eine Orientierung sowie eine horizontale und vertikale Ausrichtung.

Verwendung



Direkte Anwendung/Modellierung nach Vorschrift (Attributpfad)

Will man solche Grafikzusätze für verschiedene Plantypen definieren, sollen diese Strukturen erweitert werden. In der Erweiterung wird der Plantyp mittels eines Attributs festgehalten (meist durch eine Aufzählung oder den Verweis auf einen Katalog).

Um festzuhalten, welches Klassenattribut (Text, Lage) durch die entsprechende Grafik dargestellt werden soll, wird dieses Attribut in einem zusätzlichen «TRANSIENT-Attribut» beschrieben (TRANSIENT, damit es in den Daten nicht vorkommt), welches den entsprechenden Attributzugang (>>Name) deklariert:

```

CLASS MyClass =
  Name: Localisation_V1.MultilingualText;
  Graphic: BAG OF TextGraphic;
  Symbol (TRANSIENT): ATTRIBUTE := >>Name;
END MyClass;

```

Modellbeispiel



Zunächst werden die vordefinierten Strukturen spezialisiert und im zweiten Schritt in den Modellklassen *Source*, *River* und *Lake* angewendet.

```

STRUCTURE WaterSymbolGraphic EXTENDS SymbolGraphic =
  MapKind: [...]MapCatalogue_WithLatestModification.MapKindRef;
END WaterSymbolGraphic;

STRUCTURE WaterTextGraphic EXTENDS TextGraphic =
  MapKind: [...]MapCatalogue_WithLatestModification.MapKindValue;
  EXISTENCE CONSTRAINT MapKind REQUIRED IN
    [...]MapCatalogue_WithLatestModification.MapKind: Value;
END WaterTextGraphic;

CLASS Source =
  Position: Coord2;
  Graphic: WaterSymbolGraphic;
  Depiction(TRANSIENT): ATTRIBUTE := >>Position;
END Source;

CLASS River =
  Name: LocalisationCH_V1.MultilingualText;
  Graphic: BAG OF WaterTextGraphic;
  Depiction(TRANSIENT): ATTRIBUTE := >>Name;
END River;

```


TEIL 3 – MODELLSPEZIFIKATION

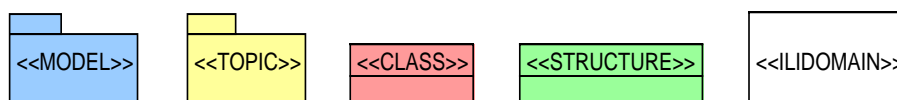
19. Modellbeschreibung, Detailspezifikation

Die Beschreibung der Modellsemantik nach [1] wird im [Teil 2](#) ab Seite 15 dieser Dokumentation, der Benutzeranleitung, abgehandelt. Als Modellbeschreibung dienen die jeweiligen Abschnitte «Thema» zu jedem Basismodul des Bundes.

20. Konzeptionelles Datenmodell: UML-Klassendiagramme

20.1. Lesehilfe

Die in den nachfolgenden UML-Klassendiagrammen dargestellten Modellelemente sind gemäss folgender Abbildung zur besseren Verständlichkeit farblich differenziert:

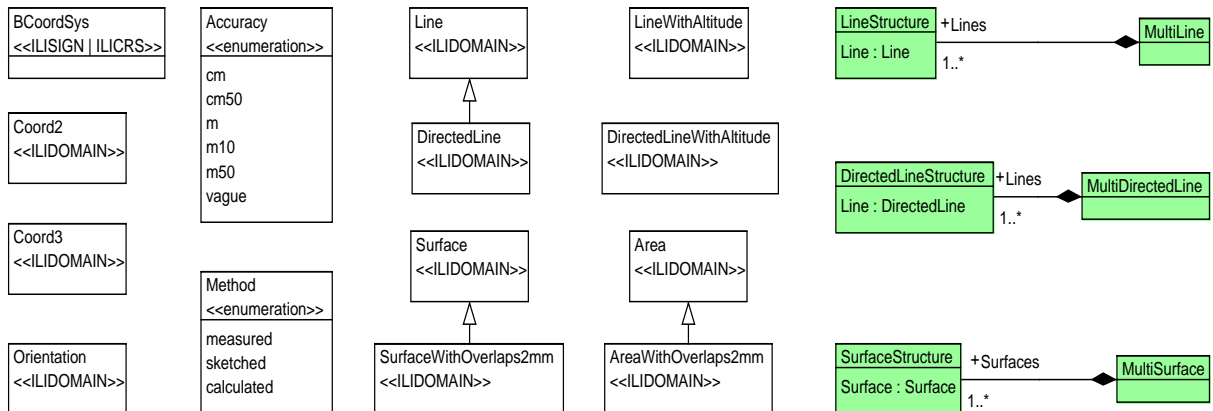


Zusätzlich werden externe Modellelemente, die im entsprechenden Diagramm aus anderen Modellen oder Themen eingefügt werden, grau dargestellt.

Die UML-Klassendiagramme sind «vertikal zu lesen». Das heisst, bei der Darstellung von Modell- oder Themenpaketen werden die zum entsprechenden Paket gehörigen Elemente *unter* diesem Paket positioniert. Durch die vertikale Anordnung wird die Eigenschaft «enthält» dargestellt.

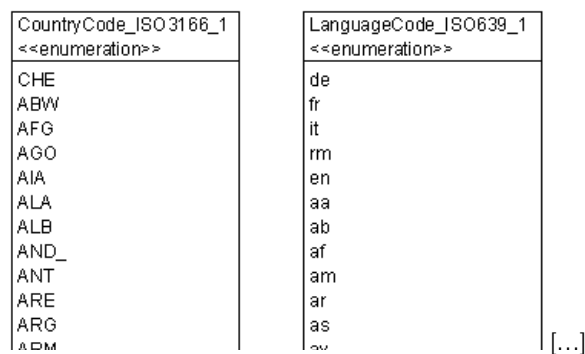
20.2. Geometrie

Die Diagramme für die Module GeometryCHLV03 und GeometryCHLV95 (S. 17ff.) sehen identisch aus:



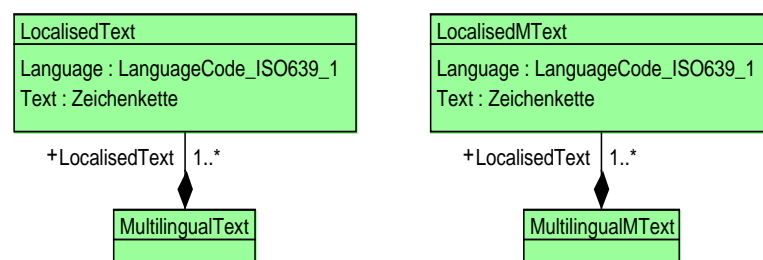
20.3. Mehrsprachigkeit: Lokalisierungs-Codes

Die Listen der InternationalCodes (S. 21) sind aus Gründen der Übersichtlichkeit nur Ausschnittsweise dargestellt. Die vollständige Liste ist dem INTERLIS-Code (S. 48) zu entnehmen:



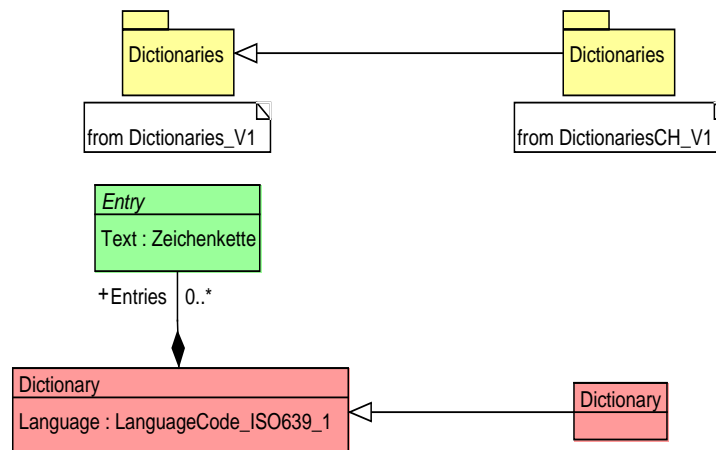
20.4. Mehrsprachigkeit: lokalisierte und mehrsprachige Texte

Die Diagramme für die Module Localisation_V1 und LocalisationCH_V1 (S. 22) sehen identisch aus:



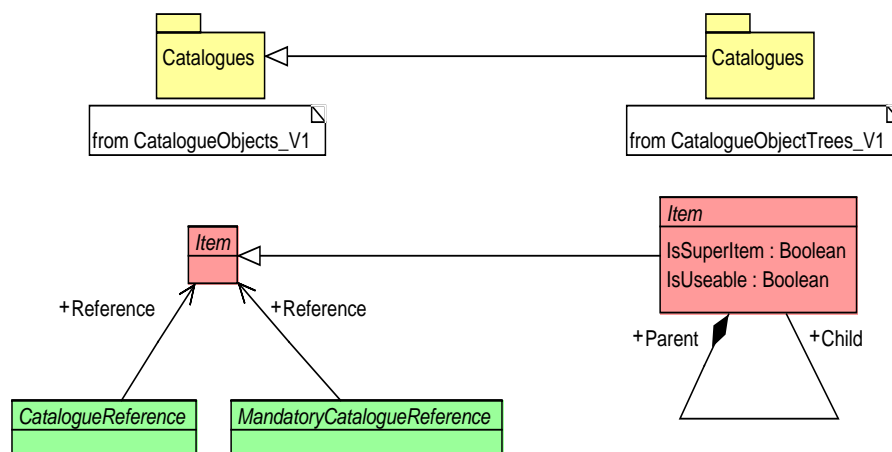
20.5. Mehrsprachigkeit: Übersetzungslisten

Die Diagramme für die Module Dictionaries V1 und DictionariesCH V1 (S. 23) werden zusammengefasst:



20.6. Kataloge: einfache und hierarchische dynamische Aufzählungen

Die Diagramme für die Module CatalogueObjects V1 (S. 24) und CatalogueObjectTrees V1 (S. 28) werden zusammengefasst:



20.7. Administrative Einheiten: Modell- und Themenübersicht

Das Modul der administrativen Einheiten umfasst drei Modelle: AdministrativeUnits, AdministrativeUnitsCH (S. 29ff.) und CHAdminCodes (S. 29). Zunächst werden die drei Modelle als Übersicht dargestellt, gefolgt von einer Gesamtübersicht, welche die Vererbungen und Abhängigkeiten darstellt:

```
classDiagram
    class AdministrativeUnits_V1
    class AdministrativeUnits
    class Agencies
    class Countries
    class CountryNames
    AdministrativeUnits_V1 --|> AdministrativeUnits
    Agencies ..> AdministrativeUnits
    Countries --|> AdministrativeUnits
    CountryNames ..> Countries
```

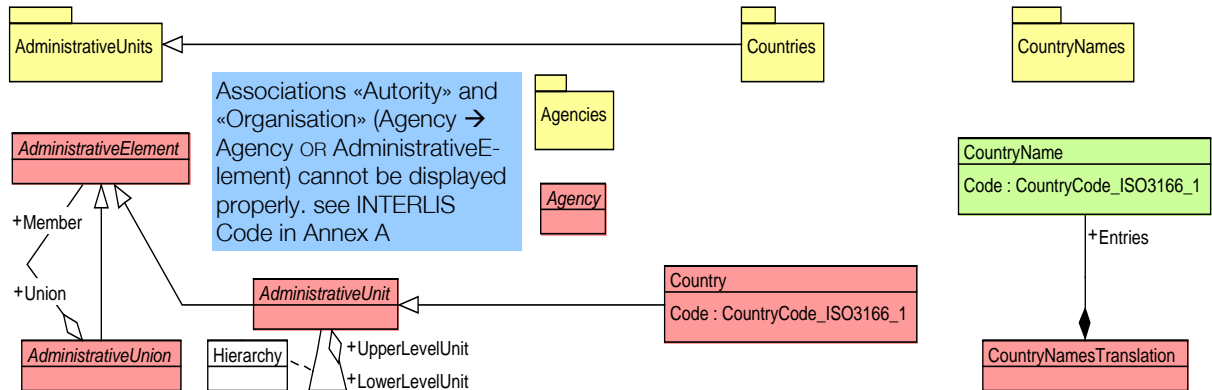
CHAdminCodes_V1	CHCantonCode <<enumeration>>
	ZH
	BE
	LU
	UR
	SZ
CHMunicipalityCode <<LIDOMAIN>>	OW
	NW
	GL
	ZG
	FR
	SO
	BS
	BL
	SH
	AR
	AI
	SG
	GR
	AG
	TG
	TI
	VD
	VS
	NE
	GE
	JU

```
graph TD;
    A[AdministrativeUnitsCH_V1] --> B[CHAgencies];
    A --> C[CHMunicipalities];
    B --> D[CHCantons];
    B --> E[CHDistricts];
    C --> E;
    C --> F[CHAdministrativeUnions];
    E --> D;
    E --> F;
```

```
classDiagram
    class AdministrativeUnits
    class Countries
    class CHAdministrativeUnits
    class CHMunicipalities
    class CHCantons
    class CHDistricts
    class CHAgencies
    class CountryNames
    class Dictionaries

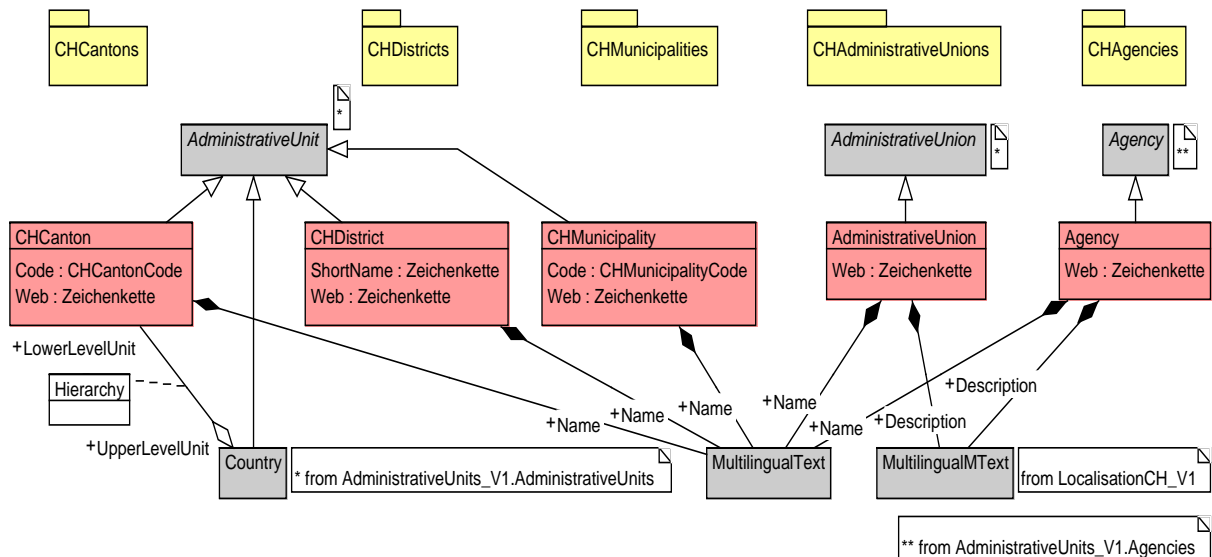
    AdministrativeUnits <|-- Countries
    AdministrativeUnits <|-- CHAdministrativeUnits
    AdministrativeUnits <|-- CHAgencies
    Countries <|-- CHAdministrativeUnits
    Countries <|-- CHMunicipalities
    Countries <|-- CHCantons
    Countries <|-- CHDistricts
    Countries <|-- CHAgencies
    CountryNames --> Dictionaries
    CountryNames ..> Countries
    CountryNames ..> CHAdministrativeUnits
    CountryNames ..> CHMunicipalities
    CountryNames ..> CHCantons
    CountryNames ..> CHDistricts
    CountryNames ..> CHAgencies
    Dictionaries ..> CountryNames
    Dictionaries ..> CHAdministrativeUnits
    Dictionaries ..> CHMunicipalities
    Dictionaries ..> CHCantons
    Dictionaries ..> CHDistricts
    Dictionaries ..> CHAgencies
    CHAdministrativeUnits <|-- CHMunicipalities
    CHAdministrativeUnits <|-- CHCantons
    CHAdministrativeUnits <|-- CHDistricts
    CHAdministrativeUnits <|-- CHAgencies
```

20.8. Administrative Einheiten: Administrative Einheiten



20.9. Administrative Einheiten: Administrative Einheiten in der Schweiz

Zwischen Kantonen (CHCanton) und Bezirken (CHDistrict) bestehen analoge Hierarchien wie zwischen Country und CHCanton. Durch die Vererbung von AdministrativeUnit wird diese Beziehung realisiert. Zwischen Bezirken resp. Kantonen und Gemeinden (CHMunicipality) besteht ebenfalls eine Hierarchiebeziehung die im Diagramm nicht dargestellt wird. Details siehe INTERLIS-Code auf Seite 52.

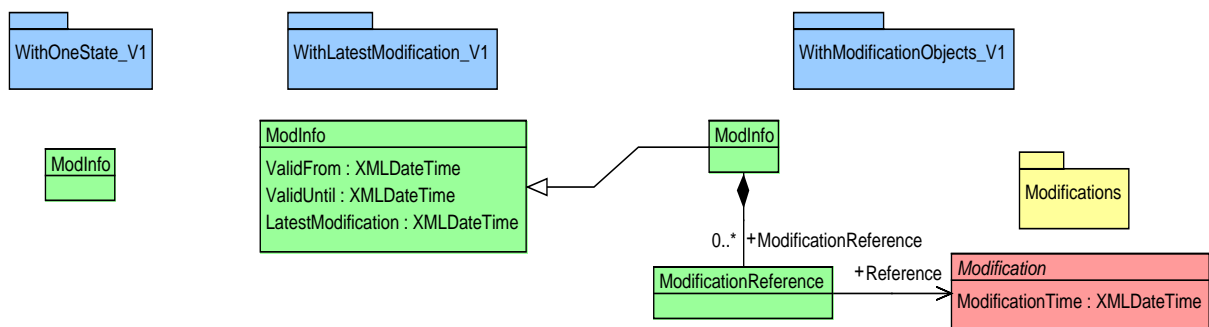


20.10. Nachführungsproblematik

Für jeden unterstützten Nachführungstyp wurde ein eigenes Modell geschaffen, das je nach Bedarf verwendet werden kann. In jedem solchen Modell wird insbesondere die Struktur *ModInfo* definiert, welche die zur Dokumentation der Modifikationen (gemäss dem verlangten Nachführungstyp) enthält.

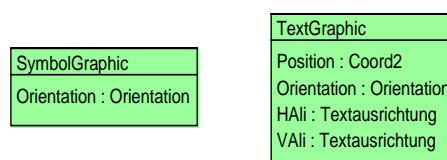
Jedes Topic soll den gefragten Nachführungstyp im Namen enthalten und vor den fachlichen Definitionen die Struktur *ModInfo* als Erweiterung der Struktur des gewünschten Nachführungstyps definieren. In den Fachdefinitionen kann dann diese Struktur eingesetzt werden, sodass der restliche Teil des Fachmodells unabhängig vom Nachführungstyp ist und damit für die Definition eines Topics eines anderen Nachführungstyps problemlos kopiert werden kann. Damit dies erreicht werden kann, enthält das Modell des Nachführungstyps *WithOneState* (bei der keine Nachführungsinformation anfällt) die Struktur *ModInfo*, obwohl sie keine Attribute enthält! Beim Nachführungstyp *WithModificationObjects* wird in der Struktur *ModInfo* eine Liste derjenigen Modifikationen geführt, die Änderungen an diesem Objekt vorgenommen haben (die jüngste Änderung zuerst).

Das Modul der Objektversionierung (S. 32ff.) umfasst drei Modelle: *WithOneState*, *WithLatestModification* und *WithModificationObjects*.



20.11. Grafikhinweise

Die Diagramme für die Module *GraphicCHLV03* und *GraphicCHLV95* (S. 36) sehen identisch aus:



ANHANG

A. Konzeptionelles Datenmodell: INTERLIS 2.3-Code

Die Basismodule des Bundes setzen sich aus mehreren INTERLIS-Modelldateien zusammen:

Bezeichnung		INTERLIS-Modelldatei-Name	
Teil 1: <u>Geometrie</u>	S. 17	<u>CHBase_Part1_GEOMETRY_20110830.ili</u>	S. 46
Teil 2: <u>Mehrsprachigkeit</u>	S. 21	<u>CHBase_Part2_LOCALISATION_20110830.ili</u>	S. 48
Teil 3: <u>Kataloge</u>	S. 24	<u>CHBase_Part3_CATALOGUEOBJECTS_20110830.ili</u>	S. 50
Teil 4: <u>Administrative Strukturen</u>	S. 29	<u>CHBase_Part4_ADMINISTRATIVEUNITS_20110830.ili</u>	S. 52
Teil 5: <u>Nachführungsproblematik</u>	S. 32	<u>CHBase_Part5_MODIFICATIONINFO_20110830.ili</u>	S. 55
Teil 6: <u>Grafikhinweise</u>	S. 36	<u>CHBase_Part6_GRAPHICANNOTATIONS_20110830.ili</u>	S. 56

Die Module werden in dieser Reihenfolge unten abgebildet. Sämtliche Basismodule des Bundes resp. oben genannte INTERLIS-Dateien (*.ili) sind in der Datenmodell-Ablage des Bundes unter

<http://models.geo.admin.ch/CHBase>

verfügbar.

Jede INTERLIS-Modelldatei weist denselben Kopf auf. Dieser wird hier nur einmal abgebildet:

```
/* #####
CHBASE - BASE MODULES OF THE SWISS FEDERATION FOR MINIMAL GEODATA MODELS
=====
BASISMODULE DES BUNDES          MODULES DE BASE DE LA CONFEDERATION
FÜR MINIMALE GEODATENMODELLE    POUR LES MODELES DE GEODONNEES MINIMAUX

PROVIDER: GKG/KOGIS - GCS/COSIG      CONTACT: models@geo.admin.ch
PUBLISHED: 2011-08-30
#####
*/

INTERLIS 2.3;

/* #####
```

[./]

Teil 1: Geometrie

```
#####
PART I -- GEOMETRY
- Package GeometryCHLV03
- Package GeometryCHLV95
*/
!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL GeometryCHLV03_V1 (en)
AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

IMPORTS UNQUALIFIED INTERLIS;
IMPORTS Units;
IMPORTS CoordSys;

REFSYSTEM BASKET BCoordSys ~ CoordSys.CoordsysTopic
  OBJECTS OF GeoCartesian2D: CHLV03
  OBJECTS OF GeoHeight: SwissOrthometricAlt;

DOMAIN
  Coord2 = COORD
    460000.000 .. 870000.000 [m] {CHLV03[1]},
    45000.000 .. 310000.000 [m] {CHLV03[2]},
    ROTATION 2 -> 1;

  Coord3 = COORD
    460000.000 .. 870000.000 [m] {CHLV03[1]},
    45000.000 .. 310000.000 [m] {CHLV03[2]},
    -200.000 .. 5000.000 [m] {SwissOrthometricAlt[1]},
    ROTATION 2 -> 1;

  Surface = SURFACE WITH (STRAIGHTS, ARCS) VERTEX Coord2;
  Area = AREA WITH (STRAIGHTS, ARCS) VERTEX Coord2;
  Line = POLYLINE WITH (STRAIGHTS, ARCS) VERTEX Coord2;
  DirectedLine EXTENDS Line = DIRECTED POLYLINE;
  LineWithAltitude = POLYLINE WITH (STRAIGHTS, ARCS) VERTEX Coord3;
  DirectedLineWithAltitude = DIRECTED POLYLINE WITH (STRAIGHTS, ARCS) VERTEX Coord3;

  /* minimal overlaps only (2mm) */
  SurfaceWithOverlaps2mm EXTENDS Surface = SURFACE WITHOUT OVERLAPS > 0.002;
  AreaWithOverlaps2mm EXTENDS Area = AREA WITHOUT OVERLAPS > 0.002;

  Orientation = 0.00000 .. 359.99999 CIRCULAR [Units.Angle_Degree] <Coord2>;

  Accuracy = (cm, cm50, m, m10, m50, vague);
  Method = (measured, sketched, calculated);

  STRUCTURE LineStructure =
    Line: Line;
  END LineStructure;

  STRUCTURE DirectedLineStructure =
    Line: DirectedLine;
  END DirectedLineStructure;

  STRUCTURE MultiLine =
    Lines: BAG {1..*} OF LineStructure;
  END MultiLine;

  STRUCTURE MultiDirectedLine =
    Lines: BAG {1..*} OF DirectedLineStructure;
  END MultiDirectedLine;
```

```

STRUCTURE SurfaceStructure =
  Surface: Surface;
END SurfaceStructure;

STRUCTURE MultiSurface =
  Surfaces: BAG {1..*} OF SurfaceStructure;
END MultiSurface;

END GeometryCHLV03_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL GeometryCHLV95_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS Units;
  IMPORTS CoordSys;

  REFSYSTEM BASKET BCoordSys ~ CoordSys.CoordsysTopic
    OBJECTS OF GeoCartesian2D: CHLV95
    OBJECTS OF GeoHeight: SwissOrthometricAlt;

  DOMAIN
    Coord2 = COORD
      2460000.000 .. 2870000.000 [m] {CHLV95[1]},
      1045000.000 .. 1310000.000 [m] {CHLV95[2]},
      ROTATION 2 -> 1;

    Coord3 = COORD
      2460000.000 .. 2870000.000 [m] {CHLV95[1]},
      1045000.000 .. 1310000.000 [m] {CHLV95[2]},
      -200.000 .. 5000.000 [m] {SwissOrthometricAlt[1]},
      ROTATION 2 -> 1;

    Surface = SURFACE WITH (STRAIGHTS, ARCS) VERTEX Coord2;
    Area = AREA WITH (STRAIGHTS, ARCS) VERTEX Coord2;
    Line = POLYLINE WITH (STRAIGHTS, ARCS) VERTEX Coord2;
    DirectedLine EXTENDS Line = DIRECTED POLYLINE;
    LineWithAltitude = POLYLINE WITH (STRAIGHTS, ARCS) VERTEX Coord3;
    DirectedLineWithAltitude = DIRECTED POLYLINE WITH (STRAIGHTS, ARCS) VERTEX Coord3;

    /* minimal overlaps only (2mm) */
    SurfaceWithOverlaps2mm EXTENDS Surface = SURFACE WITHOUT OVERLAPS > 0.002;
    AreaWithOverlaps2mm EXTENDS Area = AREA WITHOUT OVERLAPS > 0.002;

    Orientation = 0.00000 .. 359.99999 CIRCULAR [Units.Angle_Degree] <Coord2>;

    Accuracy = (cm, cm50, m, m10, m50, vague);
    Method = (measured, sketched, calculated);

  STRUCTURE LineStructure =
    Line: Line;
  END LineStructure;

  STRUCTURE DirectedLineStructure =
    Line: DirectedLine;
  END DirectedLineStructure;

  STRUCTURE MultiLine =
    Lines: BAG {1..*} OF LineStructure;
  END MultiLine;

```

```

STRUCTURE MultiDirectedLine =
  Lines: BAG {1..*} OF DirectedLineStructure;
END MultiDirectedLine;

STRUCTURE SurfaceStructure =
  Surface: Surface;
END SurfaceStructure;

STRUCTURE MultiSurface =
  Surfaces: BAG {1..*} OF SurfaceStructure;
END MultiSurface;

END GeometryCHLV95_V1.

!! #####

```

Teil 2: Mehrsprachigkeit

```

#####
PART II -- LOCALISATION
- Package InternationalCodes
- Packages Localisation, LocalisationCH
- Packages Dictionaries, DictionariesCH
*/

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL InternationalCodes_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

DOMAIN
  LanguageCode_ISO639_1 = (de,fr,it,rm,en,
    aa,ab,af,am,ar,as,ay,az,ba,be,bg,bh,bi,bn,bo,br,ca,co,cs,cy,da,dz,el,
    eo,es,et,eu,fa,fi,fj,fo,fy,ga,gd,gl,gn,gu,ha,he,hi,hr,hu,hy,ia,id,ie,
    ik,is,iu,ja,jw,ka,kk,kl,km,kn,ko,ks,ku,ky,la,ln,lo,lt,lv,mg,mi,mk,ml,
    mn,mo,mr,ms,mt,my,na,ne,nl,no,oc,om,or,pa,pl,ps,pt,qu,rn,ro,ru,rw,sa,
    sd,sg,sh,si,sk,sl,sm,sn,so,sq,sr,ss,st,su,sv,sw,ta,te,tg,th,ti,tk,tl,
    tn,to,tr,ts,tt,tw,ug,uk,ur,uz,vi,vo,wo,xh,yi,yo,za,zh,zu);

  CountryCode_ISO3166_1 = (CHE,
    ABW,AFG,AGO,AIA,ALA,ALB,AND_,ANT,ARE,ARG,ARM,ASM,ATA,ATF,ATG,AUS,
    AUT,AZE,BDI,BEL,BEN,BFA,BGD,BGR,BHR,BHS,BIH,BLR,BLZ,BMU,BOL,BRA,
    BRB,BRN,BTN,BVT,BWA,CAF,CAN,CCK,CHL,CHN,CIV,CMR,COD,COG,COK,COL,
    COM,CPV,CRI,CUB,CXR,CYM,CYP,CZE,DEU,DJI,DMA,DNK,DOM,DZA,ECU,EGY,
    ERI,ESH,ESP,EST,ETH,FIN,FJI,FLK,FRA,FRO,FSM,GAB,GBR,GEO,GGY,GHA,
    GIB,GIN,GLP,GMB,GNB,GNQ,GRC,GRD,GRL,GTM,GUF,GUM,GUY,HKG,HMD,HND,
    HRV,HTI,HUN,IDN,IMN,IND,IOT,IRL,IRN,IRQ,ISL,ISR,ITA,JAM,JEY,JOR,
    JPN,KAZ,KEN,KGZ,KHM,KIR,KNA,KOR,KWT,LAO,LBN,LBR,LBY,LCA,LIE,LKA,
    LSO,LTU,LUX,LVA,MAC,MAR,MCO,MDA,MDG,MDV,MEX,MHL,MKD,MLI,MLT,MMR,
    MNE,MNG,MNP,MOZ,MRT,MSR,MTQ,MUS,MWI,MYS,MYT,NAM,NCL,NER,NFK,NGA,
    NIC,NIU,NLD,NOR,NPL,NRU,NZL,OMN,PAK,PAN,PCN,PER,PHL,PLW,PNG,POL,
    PRI,PRK,PRT,PRY,PSE,PYF,QAT,REU,ROU,RUS,RWA,SAU,SDN,SEN,SGP,SGS,
    SHN,SJM,SLB,SLE,SLV,SMR,SOM,SPM,SRB,STP,SUR,SVK,SVN,SWE,SWZ,SYC,
    SYR,TCA,TCD,TGO,THA,TJK,TKL,TKM,TLS,TON,TTO,TUN,TUR,TUV,TWN,TZA,
    UGA,UKR,UMI,URY,USA,UZB,VAT,VCT,VEN,VGB,VIR,VNM,VUT,WLF,WSM,YEM,
    ZAF,ZMB,ZWE);

END InternationalCodes_V1.

!! #####

```



```

!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL Localisation_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED InternationalCodes_V1;

  STRUCTURE LocalisedText =
    Language: LanguageCode_ISO639_1;
    Text: MANDATORY TEXT;
  END LocalisedText;

  STRUCTURE LocalisedMText =
    Language: LanguageCode_ISO639_1;
    Text: MANDATORY MTEXT;
  END LocalisedMText;

  STRUCTURE MultilingualText =
    LocalisedText : BAG {1..*} OF LocalisedText;
    UNIQUE (LOCAL) LocalisedText:Language;
  END MultilingualText;

  STRUCTURE MultilingualMText =
    LocalisedText : BAG {1..*} OF LocalisedMText;
    UNIQUE (LOCAL) LocalisedText:Language;
  END MultilingualMText;

END Localisation_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL LocalisationCH_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED InternationalCodes_V1;
  IMPORTS Localisation_V1;

  STRUCTURE LocalisedText EXTENDS Localisation_V1.LocalisedText =
    MANDATORY CONSTRAINT
      Language == #de OR
      Language == #fr OR
      Language == #it OR
      Language == #rm OR
      Language == #en;
  END LocalisedText;

  STRUCTURE LocalisedMText EXTENDS Localisation_V1.LocalisedMText =
    MANDATORY CONSTRAINT
      Language == #de OR
      Language == #fr OR
      Language == #it OR
      Language == #rm OR
      Language == #en;
  END LocalisedMText;

  STRUCTURE MultilingualText EXTENDS Localisation_V1.MultilingualText =
    LocalisedText(EXTENDED) : BAG {1..*} OF LocalisedText;
  END MultilingualText;

  STRUCTURE MultilingualMText EXTENDS Localisation_V1.MultilingualMText =
    LocalisedText(EXTENDED) : BAG {1..*} OF LocalisedMText;
  END MultilingualMText;

```

```

END LocalisationCH_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL Dictionaries_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED InternationalCodes_V1;

  TOPIC Dictionaries (ABSTRACT) =

    STRUCTURE Entry (ABSTRACT) =
      Text: MANDATORY TEXT;
    END Entry;

    CLASS Dictionary =
      Language: MANDATORY LanguageCode_ISO639_1;
      Entries: LIST OF Entry;
    END Dictionary;

  END Dictionaries;

END Dictionaries_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL DictionariesCH_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED InternationalCodes_V1;
  IMPORTS Dictionaries_V1;

  TOPIC Dictionaries (ABSTRACT) EXTENDS Dictionaries_V1.Dictionaries =

    CLASS Dictionary (EXTENDED) =
      MANDATORY CONSTRAINT
        Language == #de OR
        Language == #fr OR
        Language == #it OR
        Language == #rm OR
        Language == #en;
    END Dictionary;

  END Dictionaries;

END DictionariesCH_V1.

! #####

```

Teil 3: Kataloge

```

#####
PART III -- CATALOGUE OBJECTS
- Package CatalogueObjects
- Package CatalogueObjectTrees
*/

!! #####
!!@technicalContact=models@geo.admin.ch

```

```

!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL CatalogueObjects_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;

  TOPIC Catalogues (ABSTRACT) =

    CLASS Item (ABSTRACT) =
      END Item;

    STRUCTURE CatalogueReference (ABSTRACT) =
      Reference: REFERENCE TO (EXTERNAL) Item;
    END CatalogueReference;

    STRUCTURE MandatoryCatalogueReference (ABSTRACT) =
      Reference: MANDATORY REFERENCE TO (EXTERNAL) Item;
    END MandatoryCatalogueReference;

  END Catalogues;

END CatalogueObjects_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL CatalogueObjectTrees_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS CatalogueObjects_V1;

  TOPIC Catalogues (ABSTRACT) EXTENDS CatalogueObjects_V1.Catalogues =

    CLASS Item (ABSTRACT,EXTENDED) =
      IsSuperItem: MANDATORY BOOLEAN;
      IsUseable: MANDATORY BOOLEAN;
    MANDATORY CONSTRAINT
      IsSuperItem OR IsUseable;
    END Item;

    ASSOCIATION EntriesTree =
      Parent -<#> Item;
      Child -- Item;
    MANDATORY CONSTRAINT
      Parent->IsSuperItem;
    END EntriesTree;

    STRUCTURE CatalogueReference (ABSTRACT,EXTENDED) =
      Reference(EXTENDED): REFERENCE TO (EXTERNAL) Item;
    MANDATORY CONSTRAINT
      Reference->IsUseable;
    END CatalogueReference;

    STRUCTURE MandatoryCatalogueReference (ABSTRACT,EXTENDED) =
      Reference(EXTENDED): MANDATORY REFERENCE TO (EXTERNAL) Item;
    MANDATORY CONSTRAINT
      Reference->IsUseable;
    END MandatoryCatalogueReference;

  END Catalogues;

END CatalogueObjectTrees_V1.

```

```
!! #####
```

Teil 4: Administrative Strukturen

```
#####
PART IV -- ADMINISTRATIVE UNITS
- Package CHAdminCodes
- Package AdministrativeUnits
- Package AdministrativeUnitsCH
*/

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL CHAdminCodes_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  DOMAIN
    CHCantonCode = (ZH,BE,LU,UR,SZ,OW,NW,GL,ZG,FR,SO,BS,BL,SH,AR,AI,SG,
                    GR,AG,TG,TI,VD,VS,NE,GE,JU);
    CHMunicipalityCode = 1..9999;  !! BFS-Nr

END CHAdminCodes_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL AdministrativeUnits_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS UNQUALIFIED CHAdminCodes_V1;
  IMPORTS UNQUALIFIED InternationalCodes_V1;
  IMPORTS Localisation_V1;
  IMPORTS Dictionaries_V1;

  TOPIC AdministrativeUnits (ABSTRACT) =

    CLASS AdministrativeElement (ABSTRACT) =
      END AdministrativeElement;

    CLASS AdministrativeUnit (ABSTRACT) EXTENDS AdministrativeElement =
      END AdministrativeUnit;

    ASSOCIATION Hierarchy =
      UpperLevelUnit (EXTERNAL) -<> {0..1} AdministrativeUnit;
      LowerLevelUnit -- AdministrativeUnit;
    END Hierarchy;

    CLASS AdministrativeUnion (ABSTRACT) EXTENDS AdministrativeElement =
      END AdministrativeUnion;

    ASSOCIATION UnionMembers =
      Union -<> AdministrativeUnion;
      Member -- AdministrativeElement;
    END UnionMembers;

  END AdministrativeUnits;

  TOPIC Countries EXTENDS AdministrativeUnits =

    CLASS Country EXTENDS AdministrativeUnit =
```

```

        Code: MANDATORY CountryCode_ISO3166_1;
    UNIQUE Code;
    END Country;

END Countries;

TOPIC CountryNames EXTENDS Dictionaries_V1.Dictionaries =
    DEPENDS ON AdministrativeUnits_V1.Countries;

    STRUCTURE CountryName EXTENDS Entry =
        Code: MANDATORY CountryCode_ISO3166_1;
    END CountryName;

    CLASS CountryNamesTranslation EXTENDS Dictionary =
        Entries(EXTENDED): LIST OF CountryName;
    UNIQUE Entries->Code;
    EXISTENCE CONSTRAINT
        Entries->Code REQUIRED IN AdministrativeUnits_V1.Countries.Country: Code;
    END CountryNamesTranslation;

END CountryNames;

TOPIC Agencies (ABSTRACT) =
    DEPENDS ON AdministrativeUnits_V1.AdministrativeUnits;

    CLASS Agency (ABSTRACT) =
    END Agency;

    ASSOCIATION Authority =
        Supervisor (EXTERNAL) -<> {1..1} Agency OR
        AdministrativeUnits_V1.AdministrativeUnits.AdministrativeElement;
    Agency -- Agency;
    END Authority;

    ASSOCIATION Organisation =
        Orderer (EXTERNAL) -- Agency OR
        AdministrativeUnits_V1.AdministrativeUnits.AdministrativeElement;
    Executor -- Agency;
    END Organisation;

END Agencies;

END AdministrativeUnits_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL AdministrativeUnitsCH_V1 (en)
    AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

    IMPORTS UNQUALIFIED INTERLIS;
    IMPORTS UNQUALIFIED CHAdminCodes_V1;
    IMPORTS UNQUALIFIED InternationalCodes_V1;
    IMPORTS LocalisationCH_V1;
    IMPORTS AdministrativeUnits_V1;

    TOPIC CHCantons EXTENDS AdministrativeUnits_V1.AdministrativeUnits =
        DEPENDS ON AdministrativeUnits_V1.Countries;

        CLASS CHCanton EXTENDS AdministrativeUnit =
            Code: MANDATORY CHCantonCode;
            Name: LocalisationCH_V1.MultilingualText;
            Web: URI;

```

```

UNIQUE Code;
END CHCanton;

ASSOCIATION Hierarchy (EXTENDED) =
  UpperLevelUnit (EXTENDED, EXTERNAL) -<> {1..1}
    AdministrativeUnits_V1.Countries.Country;
  LowerLevelUnit (EXTENDED) -- CHCanton;
MANDATORY CONSTRAINT
  UpperLevelUnit->Code == "CHE";
END Hierarchy;

END CHCantons;

TOPIC CHDistricts EXTENDS AdministrativeUnits_V1.AdministrativeUnits =
  DEPENDS ON AdministrativeUnitsCH_V1.CHCantons;

CLASS CHDistrict EXTENDS AdministrativeUnit =
  ShortName: MANDATORY TEXT*20;
  Name: LocalisationCH_V1.MultilingualText;
  Web: MANDATORY URI;
END CHDistrict;

ASSOCIATION Hierarchy (EXTENDED) =
  UpperLevelUnit (EXTENDED, EXTERNAL) -<> {1..1}
    AdministrativeUnitsCH_V1.CHCantons.CHCanton;
  LowerLevelUnit (EXTENDED) -- CHDistrict;
UNIQUE UpperLevelUnit->Code, LowerLevelUnit->ShortName;
END Hierarchy;

END CHDistricts;

TOPIC CHMunicipalities EXTENDS AdministrativeUnits_V1.AdministrativeUnits =
  DEPENDS ON AdministrativeUnits_V1.CHCantons;
  DEPENDS ON AdministrativeUnits_V1.CHDistricts;

CLASS CHMunicipality EXTENDS AdministrativeUnit =
  Code: MANDATORY CHMunicipalityCode;
  Name: LocalisationCH_V1.MultilingualText;
  Web: URI;
UNIQUE Code;
END CHMunicipality;

ASSOCIATION Hierarchy (EXTENDED) =
  UpperLevelUnit (EXTENDED, EXTERNAL) -<> {1..1}
    AdministrativeUnitsCH_V1.CHCantons.CHCanton OR
    AdministrativeUnitsCH_V1.CHDistricts.CHDistrict;
  LowerLevelUnit (EXTENDED) -- CHMunicipality;
END Hierarchy;

END CHMunicipalities;

TOPIC CHAdministrativeUnions EXTENDS AdministrativeUnits_V1.AdministrativeUnits =
  DEPENDS ON AdministrativeUnits_V1.Countries;
  DEPENDS ON AdministrativeUnitsCH_V1.CHCantons;
  DEPENDS ON AdministrativeUnitsCH_V1.CHDistricts;
  DEPENDS ON AdministrativeUnitsCH_V1.CHMunicipalities;

CLASS AdministrativeUnion (EXTENDED) =
  OID AS UUIDOID;
  Name: LocalisationCH_V1.MultilingualText;
  Web: URI;
  Description: LocalisationCH_V1.MultilingualMText;
END AdministrativeUnion;

```

```

END CHAdministrativeUnions;

TOPIC CHAgencies EXTENDS AdministrativeUnits_V1.Agencies =
  DEPENDS ON AdministrativeUnits_V1.Countries;
  DEPENDS ON AdministrativeUnitsCH_V1.CHCantons;
  DEPENDS ON AdministrativeUnitsCH_V1.CHDistricts;
  DEPENDS ON AdministrativeUnitsCH_V1.CHMunicipalities;

CLASS Agency (EXTENDED) =
  OID AS UUIDOID;
  Name: LocalisationCH_V1.MultilingualText;
  Web: URI;
  Description: LocalisationCH_V1.MultilingualMText;
END Agency;

END CHAgencies;

END AdministrativeUnitsCH_V1.

!! #####

```

Teil 5: Nachführungsproblematik

```

#####
PART V -- MODIFICATION INFORMATION
- Package WithOneState
- Package WithLatestModification
- Package WithModificationObjects
*/

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL WithOneState_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  STRUCTURE ModInfo =
  END ModInfo;

END WithOneState_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL WithLatestModification_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;

  STRUCTURE ModInfo =
    ValidFrom: XMLDateTime;
    ValidUntil: XMLDateTime;
    LatestModification: MANDATORY XMLDateTime;
  MANDATORY CONSTRAINT
    DEFINED(ValidUntil) AND LatestModification < ValidUntil;
  END ModInfo;

END WithLatestModification_V1.

!! #####

```

```

!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL WithModificationObjects_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS WithLatestModification_V1;

  TOPIC Modifications (ABSTRACT) =

    CLASS Modification (ABSTRACT) =
      ModificationTime: MANDATORY XMLDateTime;
    END Modification;

  END Modifications;

  STRUCTURE ModificationReference =
    Reference: MANDATORY REFERENCE TO (EXTERNAL)
      WithModificationObjects_V1.Modifications.Modification;
  END ModificationReference;

  STRUCTURE ModInfo EXTENDS WithLatestModification_V1.ModInfo =
    Modifications: LIST OF ModificationReference;
    /* Order: descending ModificationTime of referenced Modifications */
  END ModInfo;

END WithModificationObjects_V1.

!! #####

```

Teil 6: Grafikhinweise

```

#####
PART VI -- GRAPHIC ANNOTATIONS
- Package GraphicCHLV03
- Package GraphicCHLV95
*/

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL GraphicCHLV03_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS UNQUALIFIED GeometryCHLV03_V1;

  STRUCTURE SymbolGraphic =
    Orientation: Orientation;
  END SymbolGraphic;

  STRUCTURE TextGraphic =
    Position: MANDATORY Coord2;
    Orientation: Orientation;
    HAl: MANDATORY HALIGNMENT;
    VAl: MANDATORY VALIGNMENT;
  END TextGraphic;

END GraphicCHLV03_V1.

!! #####

```



```

!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
TYPE MODEL GraphicCHLV95_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS UNQUALIFIED GeometryCHLV95_V1;

  STRUCTURE SymbolGraphic =
    Orientation: Orientation;
  END SymbolGraphic;

  STRUCTURE TextGraphic =
    Position: MANDATORY Coord2;
    Orientation: Orientation;
    HAl: MANDATORY HALIGNMENT;
    VAl: MANDATORY VALIGNMENT;
  END TextGraphic;

END GraphicCHLV95_V1.

!! #####

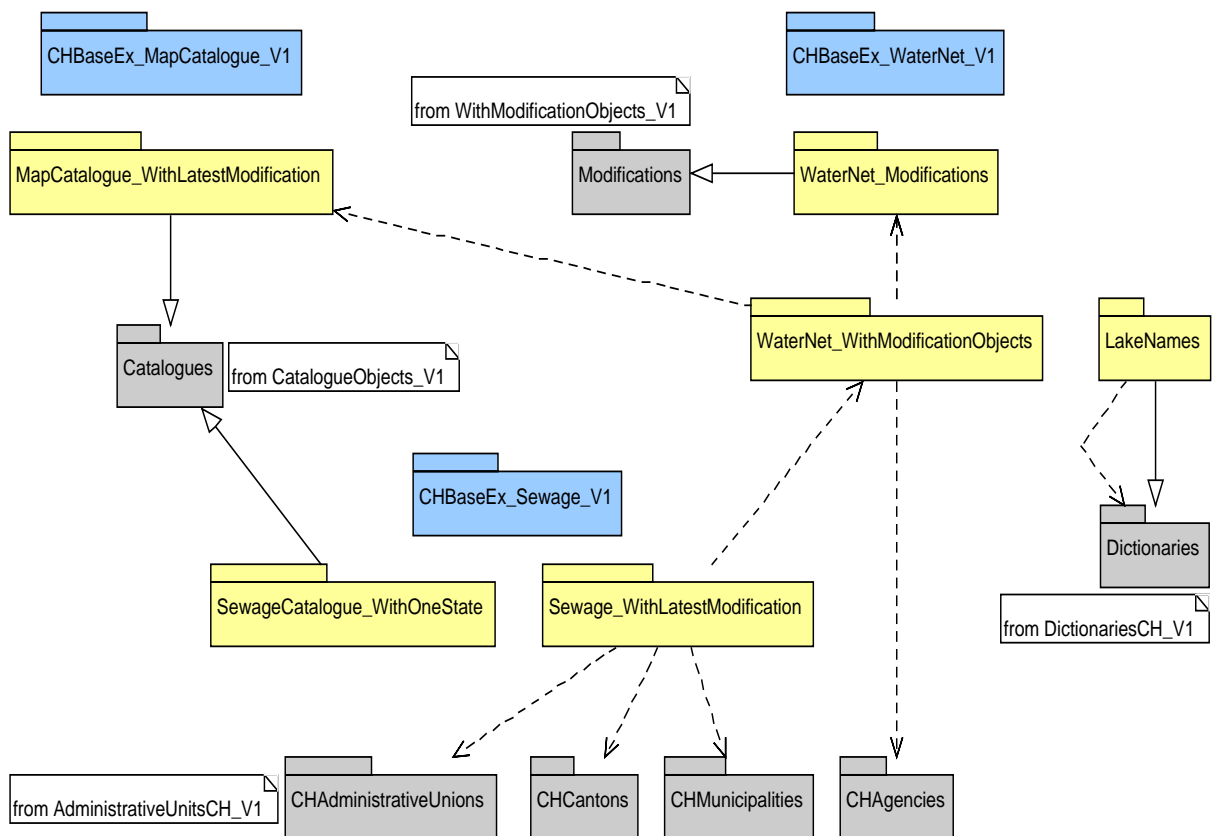
```

B. Komplettes Modellbeispiel

Konzeptionelles Datenmodell: UML-Klassendiagramme

Nachfolgend wird das Modellbeispiel im Überblick als UML-Klassendiagramme dargestellt. Für die Darstellung der UML-Elemente sei auf die Lesehilfe auf Seite 39 verwiesen. **Grau** werden die «externen» Definitionen dargestellt, welche im entsprechenden Modell Abhängigkeiten erzeugen, geerbt/erweitert oder referenziert werden.

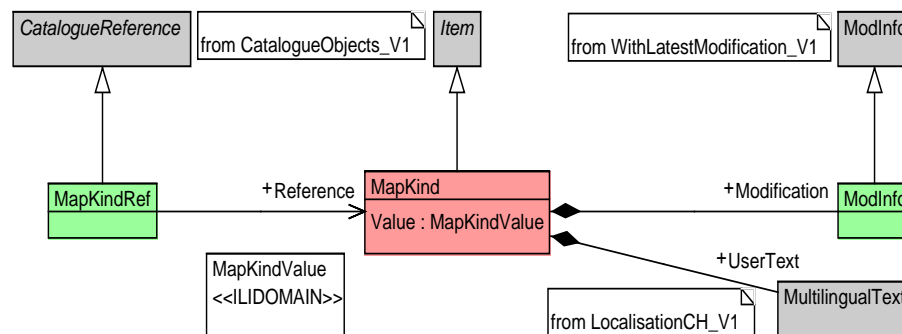
Gesamtsicht zum Modellbeispiel; Vererbungen und Abhängigkeiten der Topics:



Modellpaket	Erläuterung
CHBaseEx_MapCatalogue_V1	Modell der Kartenkataloge
MapCatalogue_WithLatestModification	Katalog der verwendeten Kartentypen. Dieser Katalog ist eine Spezialisierung der <u>Katalogobjekte</u> von CHBase (<u>Catalogues</u>)
CHBaseEx_WaterNet_V1	Modell des Gewässernetzes und dessen Aufsichtsbehörde
WaterNet_Modifications	Definition der Modifikations- oder Mutationsinformationen für die Gewässernetzobjekte. Dieses Thema ist eine Spezialisierung aus dem CHBase-Modul <u>WithModificationObjects</u> (<u>Modifications</u>).
WaterNet_WithModificationObjects	Gewässernetzinformationen: Gewässerobjekte (Quelle, Fluss, See) sowie die Aufsichtsbehörde, wodurch eine Abhängigkeit zum CHBase-Modul <u>AdministrativeUnitsCH</u> entsteht (<u>CHAgencies</u>).
LakeNames	Übersetzungsliste der Seennamen. Diese Liste ist eine Spezialisierung der <u>Übersetzungsliste</u> von CHBase (<u>Dictionaries</u>).
CHBaseEx_Sewage_V1	Modell der Kläranlagen und ihrer Verwaltung
SewageCatalogue_WithOneState	Katalog der vorkommenden Typen von Kläranlagen. Dieser Katalog ist eine Spezialisierung der <u>Katalogobjekte</u> von CHBase (<u>Catalogues</u>)
Sewage_WithLatestModification	Kläranlagen, typisiert durch den Katalog der Kläranlagen-Typen; hydrologische Zusammenhänge (Vorfluter) sowie die Verwaltung der Kläranlagen. Durch die Verwaltung entstehen die Abhängigkeiten zum CHBase-Modul <u>AdministrativeUnitsCH</u> (<u>CHAdministrativeUnions</u> , <u>CHCantons</u> , <u>CHMunicipalities</u>).

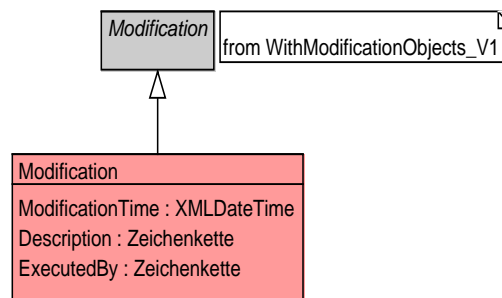
Modell **CHBaseEx_MapCatalogue_V1**, Topic **MapCatalogue_WithLatestModification**:

Die Klasse **MapKind** definiert den Katalog der Kartentypen inklusive einer mehrsprachigen Bezeichnung (UserText); die Struktur **MapKindRef** stellt die Referenz zum Kartenkatalog her. Die Struktur **ModInfo** stellt Nachführungsinformationen zur Verfügung.



Modell **CHBaseEx_WaterNet_V1**, Topic **WaterNet_Modifications**:

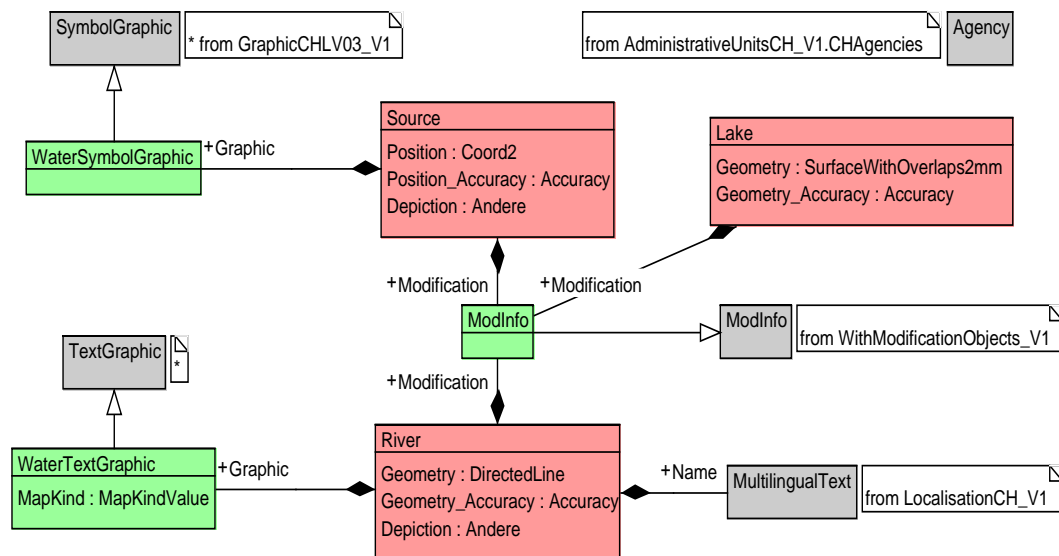
Die Klasse **Modification** spezifiziert die Objektmutations-Informationen.



Modell **CHBaseEx_WaterNet_V1**, Topic **WaterNet_WithModificationObjects**:

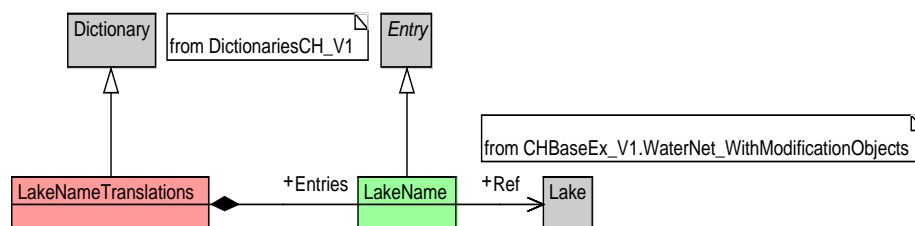
Die drei Objektklassen **Source**, **River** und **Lake** definieren die entsprechenden Objekte inklusive Geometrie. **Source** und **River** werden mittels des Strukturattributs **Graphic** jeweils durch ein Symbol (Struktur **WaterSymbolGraphic**) resp. einen Text (Struktur **WaterTextGraphic**) dargestellt. Ein **River** erhält zudem einen mehrsprachigen Namen. Alle drei Gewässerobjekte erhalten über die Struktur **ModInfo** Zugang zu den Objektmutations-Informationen. Über eine Referenz, welche im Basismodul **WithModificationObjects** vordefiniert ist, wird der Zusammenhang zur «Mutationstabelle» (**Modification**; vgl. Diagramm oben) hergestellt.

Zwei Assoziationen im Modell können nicht angemessen dargestellt werden: Water_Net (Source OR River OR Lake → River OR Lake) und QualityControlAuthority (Source OR River OR Lake → Agency). Zur genauen Definition sei auf den INTERLIS-Code ab Seite 63 verwiesen.



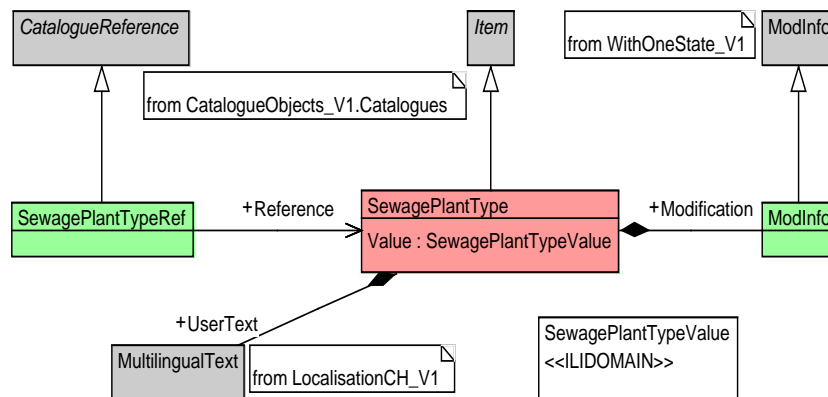
Modell **CHBaseEx_WaterNet_V1**, Topic **LakeNames**:

Die Übersetzungen der Seennamen (**LakeNameTranslations**) werden als Übersetzungsliste definiert. Jeder Eintrag der Übersetzungsliste (Strukturattribut Entries) ist ein **LakeName**. Über ein Referenzattribut Ref wird der Bezug zu einem konkreten See (**Lake**) hergestellt.



Modell **CHBaseEx_Sewage_V1**, Topic **SewageCatalogue_WithOneState**:

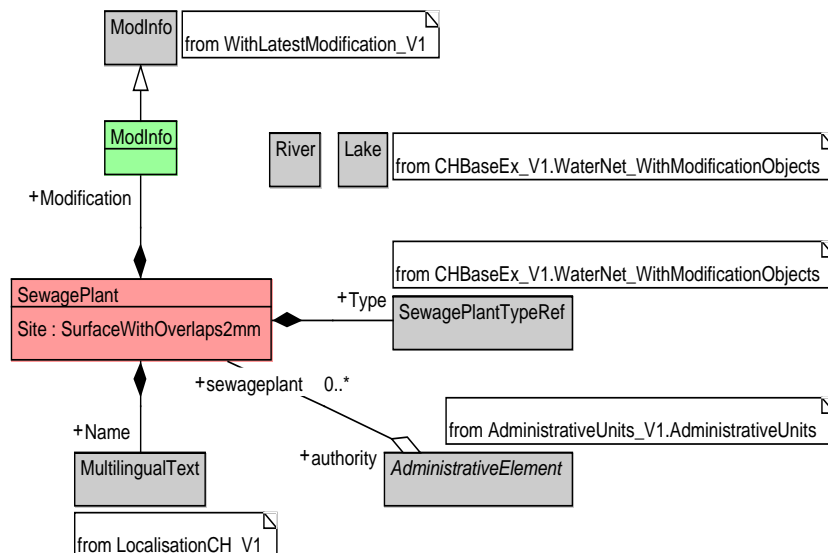
Die Klasse **SewagePlantType** definiert den Katalog der Kläranlagentypen inklusive einer mehrsprachigen Bezeichnung; die Struktur **SewagePlantTypeRef** stellt die Referenz zum Kläranlagentypen-Katalog her. . Die Struktur **ModInfo** stellt Nachführungsinformationen zur Verfügung.



Modell **CHBaseEx_Sewage_V1**, Topic **Sewage_WithLatestModification**:

Die Kläranlage (**SewagePlant**) erhält aus der Struktur **ModInfo** Nachführungsinformationen; einen mehrsprachigen Namen und über das Strukturattribut Type einen Typ aus dem oben beschriebenen Katalog zugewiesen. Die Verwaltung(sbehörde) wird assoziiert.

Eine Assoziation im Modell kann nicht angemessen dargestellt werden: Recipient (River OR Lake → SewagePlant). Zur genauen Definition sei auf den INTERLIS-Code ab Seite 63 verwiesen.



Konzeptionelles Datenmodell: INTERLIS-Code

INTERLIS-Modelldatei-Name: CHBaseEx_V1.ili

```
/* #####
Example Data Models for CHBase Application
#####
*/
INTERLIS 2.3;

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL CHBaseEx_MapCatalogue_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS LocalisationCH_V1;           !! Part II of CHBase
  IMPORTS CatalogueObjects_V1;       !! Part III of CHBase
  IMPORTS WithLatestModification_V1;  !! Part V of CHBase
  IMPORTS UNQUALIFIED GraphicCHLV03_V1; !! Part VI of CHBase

  !! =====
  TOPIC MapCatalogue_WithLatestModification
  EXTENDS CatalogueObjects_V1.Catalogues =

    STRUCTURE ModInfo EXTENDS WithLatestModification_V1.ModInfo =
    END ModInfo;

    DOMAIN
      MapKindValue = MANDATORY TEXT;

    CLASS MapKind
    EXTENDS CatalogueObjects_V1.Catalogues.Item =
      Modification: ModInfo;
      Value: MapKindValue;
      UserText: LocalisationCH_V1.MultilingualText;
    UNIQUE Value;
    END MapKind;

    STRUCTURE MapKindRef
    EXTENDS CatalogueObjects_V1.Catalogues.CatalogueReference =
      Reference(EXTENDED): MANDATORY REFERENCE TO (EXTERNAL) MapKind;
    END MapKindRef;

  END MapCatalogue_WithLatestModification;

END CHBaseEx_MapCatalogue_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL CHBaseEx_WaterNet_V1 (en)
  AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

  IMPORTS UNQUALIFIED INTERLIS;
  IMPORTS UNQUALIFIED GeometryCHLV03_V1; !! Part I of CHBase
  IMPORTS LocalisationCH_V1;           !! Part II of CHBase
  IMPORTS Dictionaries_V1;
  IMPORTS DictionariesCH_V1;
  IMPORTS CatalogueObjects_V1;       !! Part III of CHBase
  IMPORTS AdministrativeUnits_V1;    !! Part IV of CHBase
  IMPORTS AdministrativeUnitsCH_V1;
```

→ XML-Datei mit dem konkreten Katalog der Kartentypen

```

IMPORTS WithOneState_V1;           !! Part   V of CHBase
IMPORTS WithModificationObjects_V1;
IMPORTS UNQUALIFIED GraphicCHLV03_V1;  !! Part  VI of CHBase
IMPORTS CHBaseEx_MapCatalogue_V1;      !! Model example

!! =====
TOPIC WaterNet_Modifications
EXTENDS WithModificationObjects_V1.Modifications =

    CLASS Modification (EXTENDED) =
        ModificationTime(EXTENDED): MANDATORY XMLDateTime;
        Description: MANDATORY TEXT;
        ExecutedBy: MANDATORY TEXT;
    END Modification;

END WaterNet_Modifications;

!! =====
TOPIC WaterNet_WithModificationObjects =
    OID AS UUIDOID;
    DEPENDS ON AdministrativeUnitsCH_V1.CHAgencies;
    DEPENDS ON CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification;
    DEPENDS ON CHBaseEx_WaterNet_V1.WaterNet_Modifications;

    STRUCTURE ModInfo EXTENDS WithModificationObjects_V1.ModInfo =
        END ModInfo;

    STRUCTURE WaterSymbolGraphic
    EXTENDS SymbolGraphic =
        MapKind: CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKindRef;
    END WaterSymbolGraphic;

    STRUCTURE WaterTextGraphic
    EXTENDS TextGraphic =
        MapKind: CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKindValue;
    EXISTENCE CONSTRAINT
        MapKind REQUIRED IN CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind: Value;
    END WaterTextGraphic;

    CLASS Source =
        Modification: ModInfo;
        Position: Coord2;
        Position_Accuracy: Accuracy;
        Graphic: WaterSymbolGraphic;
        Depiction(TRANSIENT): ATTRIBUTE := >>Position;
    END Source;

    CLASS River =
        Modification: ModInfo;
        Name: LocalisationCH_V1.MultilingualText;
        Geometry: DirectedLine;
        Geometry_Accuracy: Accuracy;
        Graphic: BAG OF WaterTextGraphic;
        Depiction(TRANSIENT): ATTRIBUTE := >>Name;
    END River;

    CLASS Lake =
        Modification: ModInfo;
        Geometry: SurfaceWithOverlaps2mm;
        Geometry_Accuracy: Accuracy;
    END Lake;

```



```

ASSOCIATION Water_Net =
    connect -- Source OR River OR Lake;
    with    -- River OR Lake;
    Modification: ModInfo;
END Water_Net;

ASSOCIATION QualityControlAuthority =
    watersubject -- Source OR River OR Lake;
    controller (EXTERNAL) -- {0..1} AdministrativeUnitsCH_V1.CHAgencies.Agency;
END QualityControlAuthority;

END WaterNet_WithModificationObjects;

!! =====
TOPIC LakeNames EXTENDS DictionariesCH_V1.Dictionaries =
    DEPENDS ON DictionariesCH_V1.Dictionaries;

STRUCTURE LakeName EXTENDS Entry =
    Ref: MANDATORY REFERENCE TO (EXTERNAL)
        CHBaseEx_WaterNet_V1.WaterNet_WithModificationObjects.Lake;
END LakeName;

CLASS LakeNameTranslations EXTENDS Dictionary =
    Entries (EXTENDED): LIST OF LakeName;
END LakeNameTranslations;

END LakeNames;

END CHBaseEx_WaterNet_V1.

!! #####
!!@technicalContact=models@geo.admin.ch
!!@furtherInformation=http://www.geo.admin.ch/internet/geoportal/de/home/topics/geobasedata/models.html
MODEL CHBaseEx_Sewage_V1 (en)
AT "http://www.geo.admin.ch" VERSION "2011-08-30" =

IMPORTS UNQUALIFIED INTERLIS;
IMPORTS UNQUALIFIED GeometryCHLV03_V1;  !! Part I of CHBase
IMPORTS LocalisationCH_V1;              !! Part II of CHBase
IMPORTS CatalogueObjects_V1;            !! Part III of CHBase
IMPORTS AdministrativeUnits_V1;          !! Part IV of CHBase
IMPORTS AdministrativeUnitsCH_V1;
IMPORTS WithOneState_V1;                 !! Part V of CHBase
IMPORTS WithLatestModification_V1;
IMPORTS UNQUALIFIED GraphicCHLV03_V1;    !! Part VI of CHBase
IMPORTS CHBaseEx_WaterNet_V1;

!! =====
TOPIC SewageCatalogue_WithOneState EXTENDS CatalogueObjects_V1.Catalogues =

DOMAIN
    SewagePlantTypeValue = MANDATORY TEXT;

STRUCTURE ModInfo EXTENDS WithOneState_V1.ModInfo =
END ModInfo;

CLASS SewagePlantType
EXTENDS CatalogueObjects_V1.Catalogues.Item =
    Modification: ModInfo;
    Value: SewagePlantTypeValue;
    UserText: LocalisationCH_V1.MultilingualText;

```

→ XML-Datei mit der konkreten Übersetzungsliste der Seennamen

→ XML-Datei mit dem konkreten Katalog der Kläranlagentypen

```

    UNIQUE Value;
    END SewagePlantType;

    STRUCTURE SewagePlantTypeRef
    EXTENDS CatalogueObjects_V1.Catalogues.CatalogueReference =
        Reference(EXTENDED): MANDATORY REFERENCE TO (EXTERNAL) SewagePlantType;
    END SewagePlantTypeRef;

    END SewageCatalogue_WithOneState;

!! =====
    TOPIC Sewage_WithLatestModification =
        OID AS UUIDOID;
        DEPENDS ON AdministrativeUnitsCH_V1.CHCantons;
        DEPENDS ON AdministrativeUnitsCH_V1.CHMunicipalities;
        DEPENDS ON AdministrativeUnitsCH_V1.CHAdministrativeUnions;
        DEPENDS ON CHBaseEx_WaterNet_V1.WaterNet_WithModificationObjects;

        STRUCTURE ModInfo EXTENDS WithLatestModification_V1.ModInfo =
            END ModInfo;

        CLASS SewagePlant =
            Modification: ModInfo;
            Name: LocalisationCH_V1.MultilingualText;
            Type: CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantTypeRef;
            Site: SurfaceWithOverlaps2mm;
        END SewagePlant;

        ASSOCIATION Recipient = !! Vorfluter / cours d'eau récepteur
            sewageplant -- SewagePlant;
            recipient (EXTERNAL) -- {1..1} CHBaseEx_WaterNet_V1.WaterNet_WithModificationObjects.River OR
            CHBaseEx_WaterNet_V1.WaterNet_WithModificationObjects.Lake;
        END Recipient;

        ASSOCIATION SewageManagementAuthority =
            authority (EXTERNAL) -<> {1..1} AdministrativeUnits_V1.AdministrativeUnits.AdministrativeElement;
            sewageplant -- SewagePlant;
        END SewageManagementAuthority;

    END Sewage_WithLatestModification;

END CHBaseEx_Sewage_V1.

!! #####

```

Folgende XML-Daten gehören zum Modell:

1. Katalog der Kartentypen *CHBaseEx_V1_MapCatalogue.xml*

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- File CHBaseEx_V1_MapCatalogue.xml 2011-08-30 (http://www.geo.admin.ch) -->
<TRANSFER xmlns="http://www.interlis.ch/INTERLIS2.3" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://models.geo.admin.ch/CHBase/CHBaseEx_V1.xsd">
  <HEADERSECTION VERSION="2.3" SENDER=" http://www.geo.admin.ch ">
    <MODELS>
      <MODEL NAME="LocalisationCH_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
      <MODEL NAME="CatalogueObjects_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
      <MODEL NAME="WithLatestModification_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
      <MODEL NAME="CHBaseEx_MapCatalogue_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
    </MODELS>
  </HEADERSECTION>
  <DATASECTION>
    <CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification BID="b001">
      <CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind TID="001001">
        <Value>OverviewMap</Value>
        <UserText>
          <LocalisationCH_V1.MultiLingualText>
            <LocalisationCH_V1.LocalisedText>
              <Language>de</Language>
              <Text>Übersichtsplan</Text>
            </LocalisationCH_V1.LocalisedText>
            <LocalisationCH_V1.LocalisedText>
              <Language>fr</Language>
              <Text>Plan d'ensemble</Text>
            </LocalisationCH_V1.LocalisedText>
            <LocalisationCH_V1.LocalisedText>
              <Language>en</Language>
              <Text>Overview map</Text>
            </LocalisationCH_V1.LocalisedText>
          </LocalisationCH_V1.MultiLingualText>
        </UserText>
      </CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind>
      <CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind TID="001002">
        <Value>HydroMap</Value>
        <UserText>
          <LocalisationCH_V1.MultiLingualText>
            <LocalisationCH_V1.LocalisedText>
              <Language>de</Language>
              <Text>Hydrologischer Plan</Text>
            </LocalisationCH_V1.LocalisedText>
            <LocalisationCH_V1.LocalisedText>
              <Language>fr</Language>
              <Text>Plan hydrologique</Text>
            </LocalisationCH_V1.LocalisedText>
            <LocalisationCH_V1.LocalisedText>
              <Language>en</Language>
              <Text>Hydrology map</Text>
            </LocalisationCH_V1.LocalisedText>
          </LocalisationCH_V1.MultiLingualText>
        </UserText>
      </CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind>
      <CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind TID="001003">
        <Value>WaterQualMap</Value>
        <UserText>
          <LocalisationCH_V1.MultiLingualText>
            <LocalisationCH_V1.LocalisedText>
              <Language>de</Language>
              <Text>Wasserqualitäts-Karte</Text>
            </LocalisationCH_V1.LocalisedText>
            <LocalisationCH_V1.LocalisedText>
              <Language>fr</Language>
              <Text>Plan de qualité de l'eau</Text>
            </LocalisationCH_V1.LocalisedText>
            <LocalisationCH_V1.LocalisedText>
              <Language>en</Language>
              <Text>Water quality map</Text>
            </LocalisationCH_V1.LocalisedText>
          </LocalisationCH_V1.MultiLingualText>
        </UserText>
      </CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind>
    </CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification>
  </DATASECTION>
</TRANSFER>
```

```

        </LocalisationCH_V1.MultiLingualText>
    </UserText>
    </CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification.MapKind>
    </CHBaseEx_MapCatalogue_V1.MapCatalogue_WithLatestModification>
</DATASECTION>
</TRANSFER>

```

2. Katalog der Kläranlagen-Typen *CHBaseEx_V1_SewageCatalogue.xml*

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- File CHBaseEx_V1_SewageCatalogue.xml 2011-08-30 (http://www.geo.admin.ch) -->
<TRANSFER xmlns="http://www.interlis.ch/INTERLIS2.3" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://models.geo.admin.ch/CHBase/CHBaseEx_V1.xsd">
    <HEADERSECTION VERSION="2.3" SENDER="http://www.geo.admin.ch">
        <MODELS>
            <MODEL NAME="LocalisationCH_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
            <MODEL NAME="CatalogueObjects_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
            <MODEL NAME="WithOneState_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
            <MODEL NAME="CHBaseEx_SewageCatalogue_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
        </MODELS>
    </HEADERSECTION>
    <DATASECTION>
        <CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState BID="b001">
            <CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantType TID="001001">
                <Value>Mechanical</Value>
                <UserText>
                    <LocalisationCH_V1.MultiLingualText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>de</Language>
                            <Text>Mechanische Klärung</Text>
                        </LocalisationCH_V1.LocalisedText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>fr</Language>
                            <Text>Épuration mécanique</Text>
                        </LocalisationCH_V1.LocalisedText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>en</Language>
                            <Text>Mechanical treatment</Text>
                        </LocalisationCH_V1.LocalisedText>
                    </LocalisationCH_V1.MultiLingualText>
                </UserText>
            </CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantType>
            <CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantType TID="001002">
                <Value>Biological</Value>
                <UserText>
                    <LocalisationCH_V1.MultiLingualText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>de</Language>
                            <Text>Biologische Klärung</Text>
                        </LocalisationCH_V1.LocalisedText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>fr</Language>
                            <Text>Épuration biologique</Text>
                        </LocalisationCH_V1.LocalisedText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>en</Language>
                            <Text>Biological treatment</Text>
                        </LocalisationCH_V1.LocalisedText>
                    </LocalisationCH_V1.MultiLingualText>
                </UserText>
            </CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantType>
            <CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantType TID="001003">
                <Value>MembraneFiltration</Value>
                <UserText>
                    <LocalisationCH_V1.MultiLingualText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>de</Language>
                            <Text>Membran-Filterung</Text>
                        </LocalisationCH_V1.LocalisedText>
                        <LocalisationCH_V1.LocalisedText>
                            <Language>fr</Language>
                            <Text>Filtration par membrane</Text>

```

```

        </LocalisationCH_V1.LocalisedText>
        <LocalisationCH_V1.LocalisedText>
            <Language>en</Language>
            <Text>Membrane filtration</Text>
        </LocalisationCH_V1.LocalisedText>
        </LocalisationCH_V1.MultiLingualText>
    </UserText>
    </CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState.SewagePlantType>
    </CHBaseEx_Sewage_V1.SewageCatalogue_WithOneState>
</DATASECTION>
</TRANSFER>

```

3. Übersetzungsliste mit den Seennamen *CHBaseEx_V1_LakeNames.xml*

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- File CHBaseEx_V1_LakeNames.xml 2011-08-30 (http://www.geo.admin.ch) -->
<TRANSFER xmlns="http://www.interlis.ch/INTERLIS2.3" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://models.geo.admin.ch/swisstopo/CHBaseEx_V1.xsd">
    <HEADERSECTION VERSION="2.3" SENDER=" http://www.geo.admin.ch ">
        <MODELS>
            <MODEL NAME="DictionariesCH_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
            <MODEL NAME="CHBaseEx_WaterNet_V1" URI="http://models.geo.admin.ch/CHBase" VERSION="2011-08-30"/>
        </MODELS>
    </HEADERSECTION>
    <DATASECTION>
        <CHBaseEx_WaterNet_V1.LakeNames BID="b001">
            <CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations TID="001001">
                <Language>de</Language>
                <Entries>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Guggitalersee</Text>
                        <Ref REF="123401"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Qualensee</Text>
                        <Ref REF="123402"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Foowilersee</Text>
                        <Ref REF="123403"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                </Entries>
            </CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations>
            <CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations TID="001002">
                <Language>fr</Language>
                <Entries>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Lac du Guggital</Text>
                        <Ref REF="123401"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Lac du Tourment</Text>
                        <Ref REF="123402"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Lac de Foo-Villars</Text>
                        <Ref REF="123403"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                </Entries>
            </CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations>
            <CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations TID="001003">
                <Language>en</Language>
                <Entries>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Lake Guggital</Text>
                        <Ref REF="123401"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                    <CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                        <Text>Lake Sorrow</Text>
                        <Ref REF="123402"></Ref>
                    </CHBaseEx_WaterNet_V1.LakeNames.LakeName>
                </Entries>
            </CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations>
        </CHBaseEx_WaterNet_V1.LakeNames>
    </DATASECTION>
</TRANSFER>

```

Diese Referenzen zeigen auf konkrete «Lake»-Objekte und sind im Beispiel fiktiv

```
<CHBaseEx_WaterNet_V1.LakeNames.LakeName>
  <Text>Lake Footown</Text>
  <Ref REF="123403"></Ref>
</CHBaseEx_WaterNet_V1.LakeNames.LakeName>
</Entries>
</CHBaseEx_WaterNet_V1.LakeNames.LakeNameTranslations>
</CHBaseEx_WaterNet_V1.LakeNames>
</DATASECTION>
</TRANSFER>
```

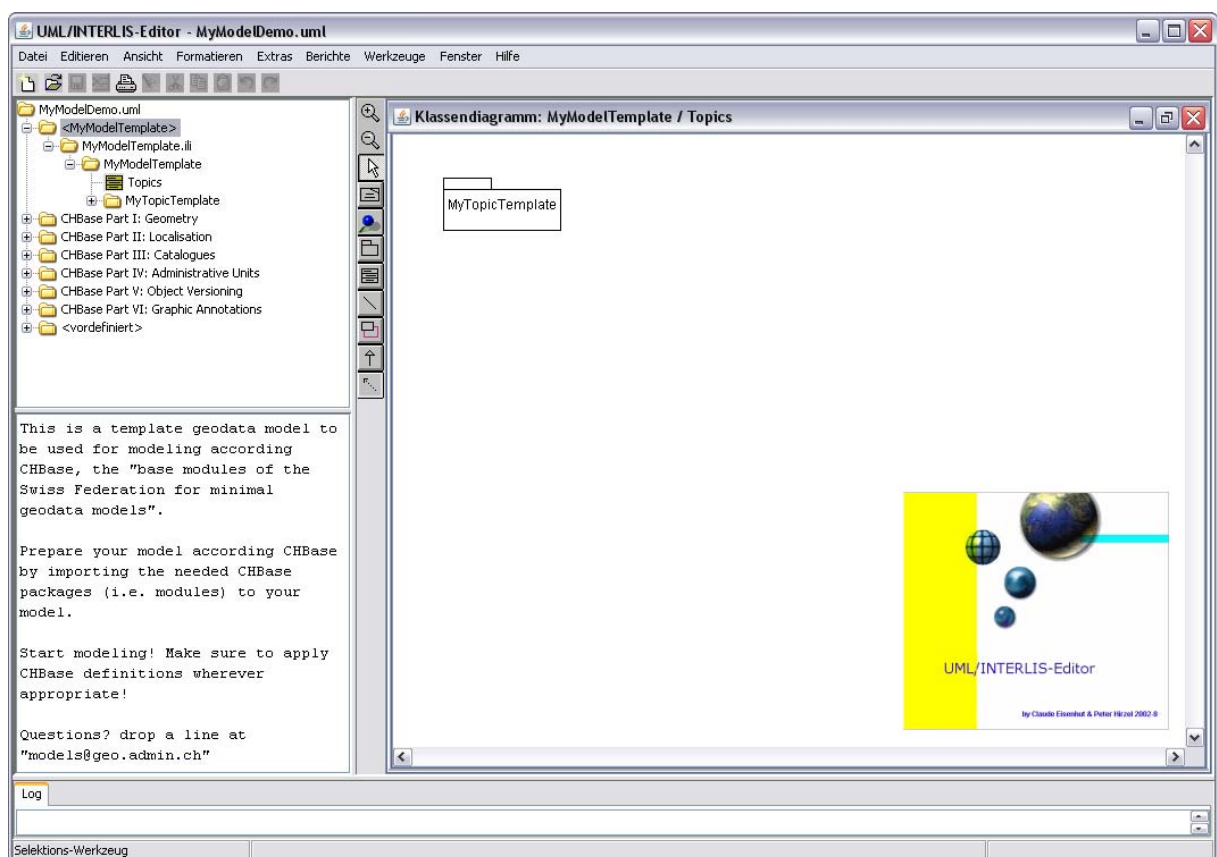
C. MGDM modellieren mit der UML-Vorlage

Für die praktische Arbeit an MGDM basierend auf CHBase wurde eine UML-Vorlage erstellt. Diese Vorlage ist für den UML/INTERLIS-Editor [1, Anhang] konfektioniert und kann online bezogen werden. Die UML-Vorlage beinhaltet sämtliche CHBase-Module sowie ein Muster für ein eigenes Modell *MyModelTemplate*.

Nachfolgend wird die Modellierung mit der UML-Vorlage Schritt für Schritt erläutert.

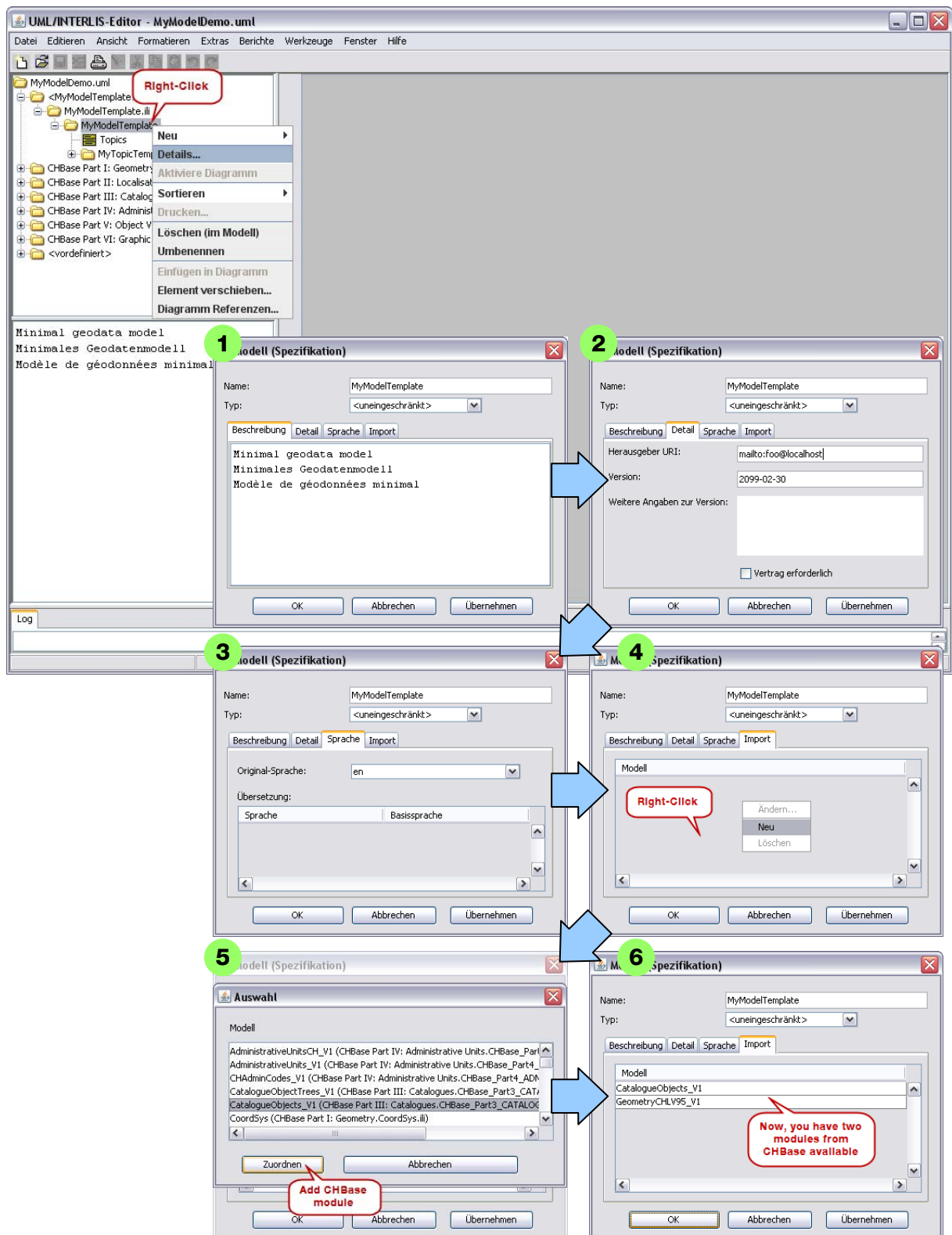
Datenmodell, Modellimport

Die Gesamtansicht zeigt zunächst die Modell-Vorlage, gefolgt von den Basismodulen.



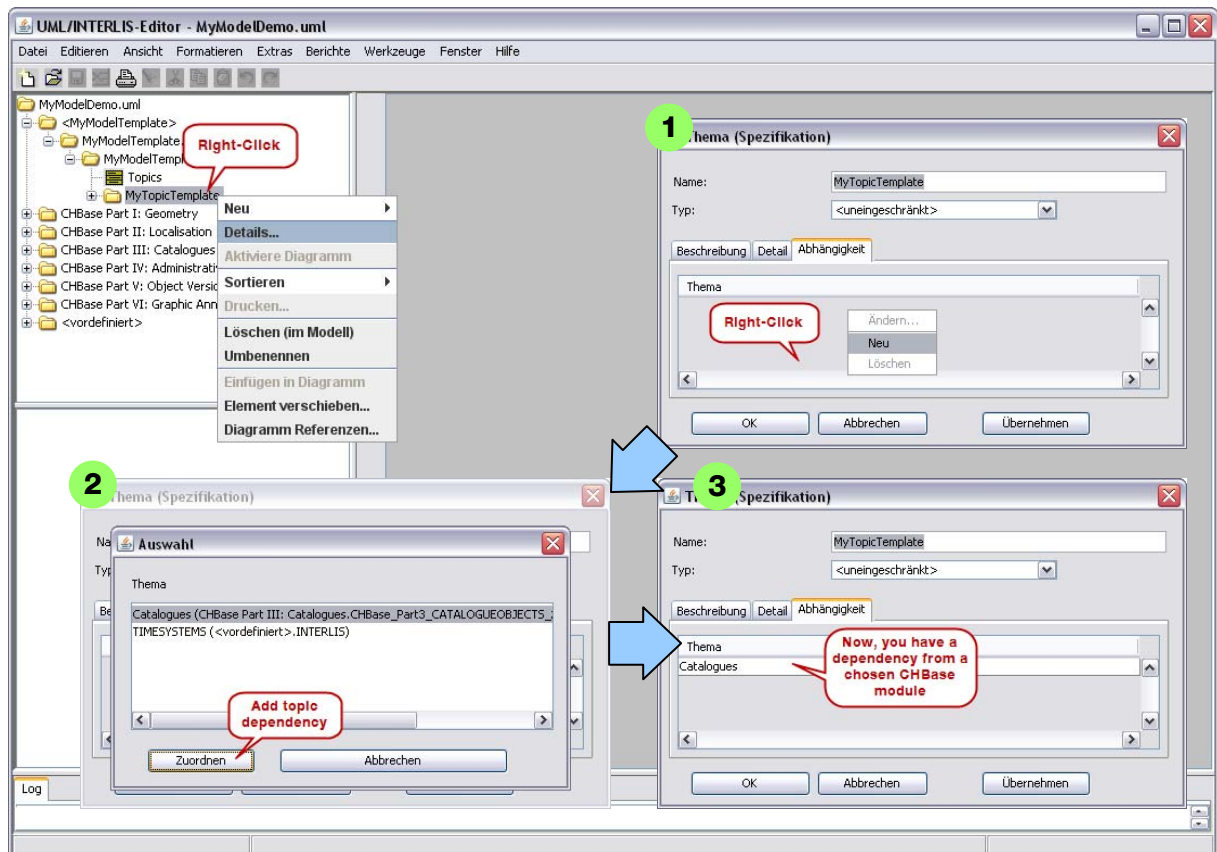
Das Erläuterungsfeld unten links zeigt ein paar grundlegende Erläuterungen zum Vorgehen der Modellierung mit CHBase sowie eine Kontaktmöglichkeit bei Fragen.

Um mit dem Modellieren zu beginnen, werden als erstes die gewünschten Basismodule in das Datenmodell *importiert*. Dadurch stehen die entsprechenden Definitionen für die Anwendung im MGDM zur Verfügung. Mit den «Details» des MODELS können u.a. Modelle importiert werden:

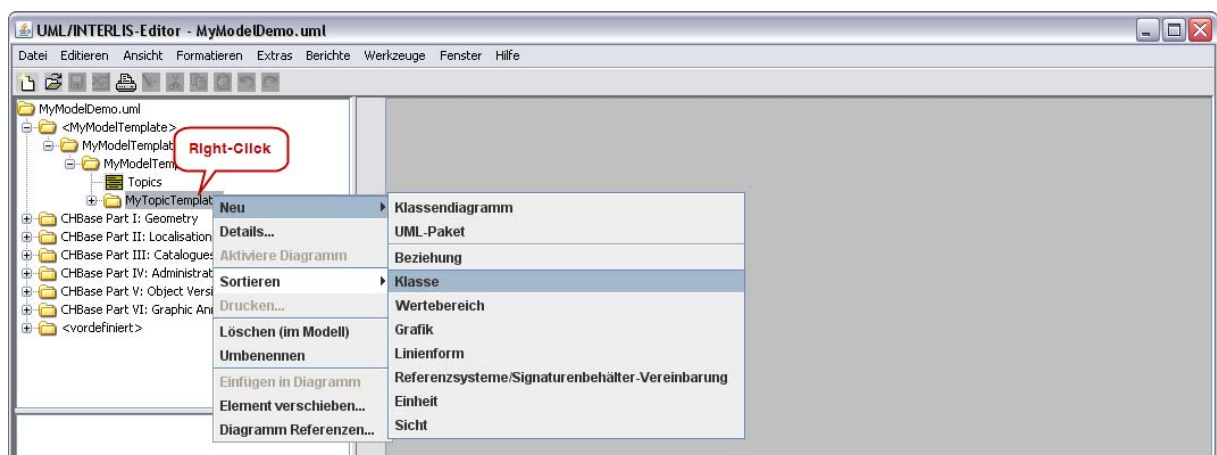


Datenthema, Klassen erzeugen

Falls im MGDM Basiskonstrukte angewendet werden (insbes. bei Behälter-übergreifenden Beziehungen), so bewirkt dies u.U. eine Abhängigkeit des Themas (Topic). Diese Abhängigkeit muss deshalb im Topic des MGDM deklariert werden. Vgl. dazu [6], Kap. 2.5.2.

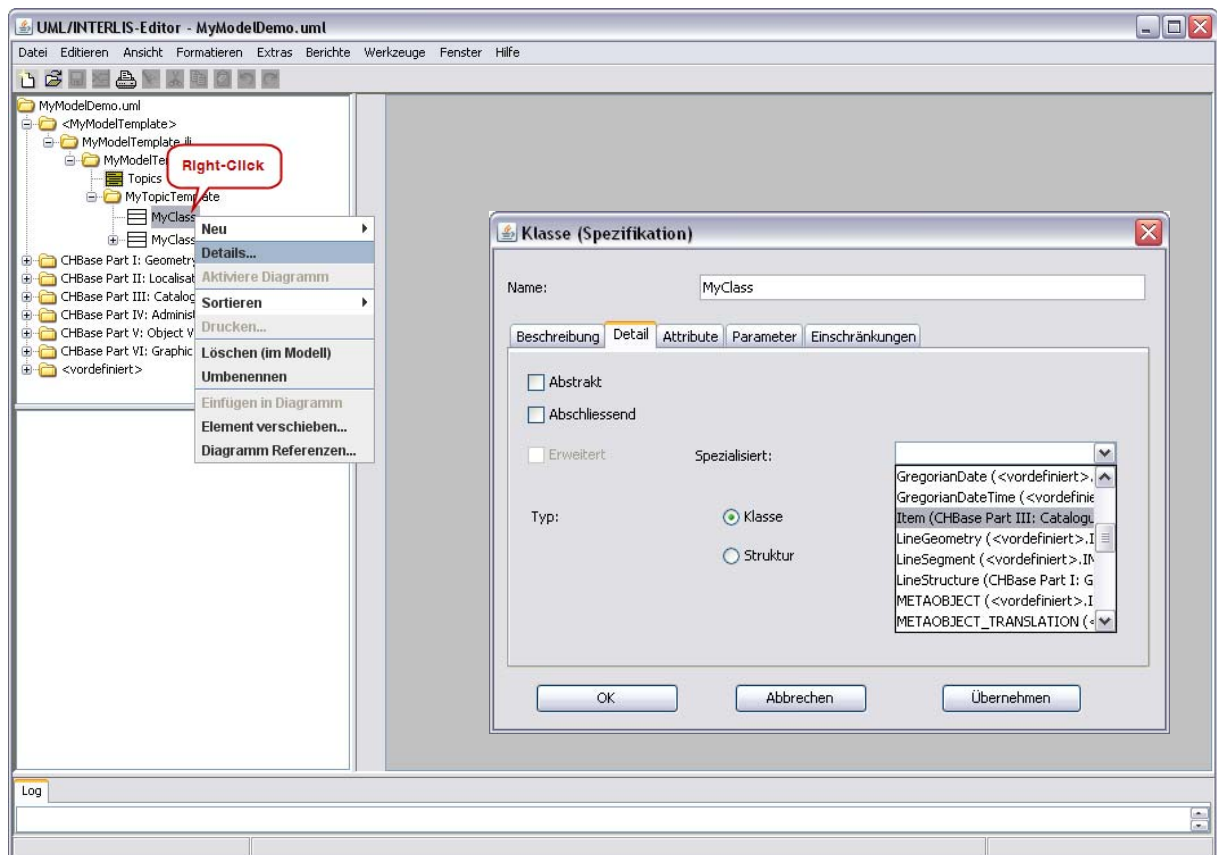


Innerhalb eines Topics werden die Modellklassen (oder -strukturen etc.) erzeugt.

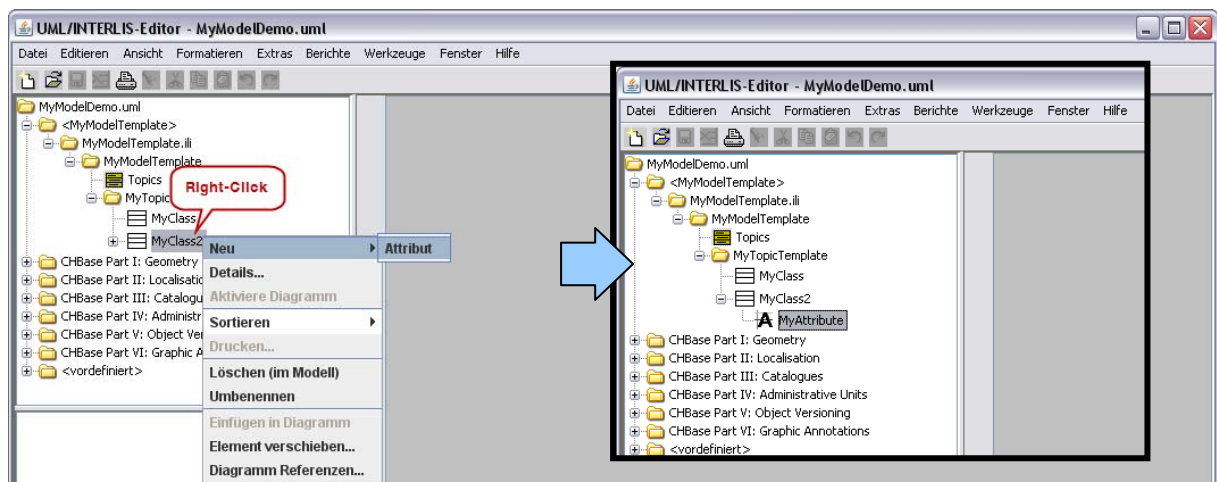


Klassen, Vererbung von CHBase-Konstrukten, Attribute erzeugen

Wenn eine Klasse durch Vererbung einer Basismodul-Klasse entsteht, so wird dies bei den «Details» definiert. Im Dialog «Detail» wird bei der Rubrik «Spezialisiert» die entsprechende Klasse aus dem Basismodul ausgewählt. Durch den vorgängigen Modulimport und die Deklaration der Abhängigkeit des Topics werden die entsprechenden Konstrukte zugänglich.

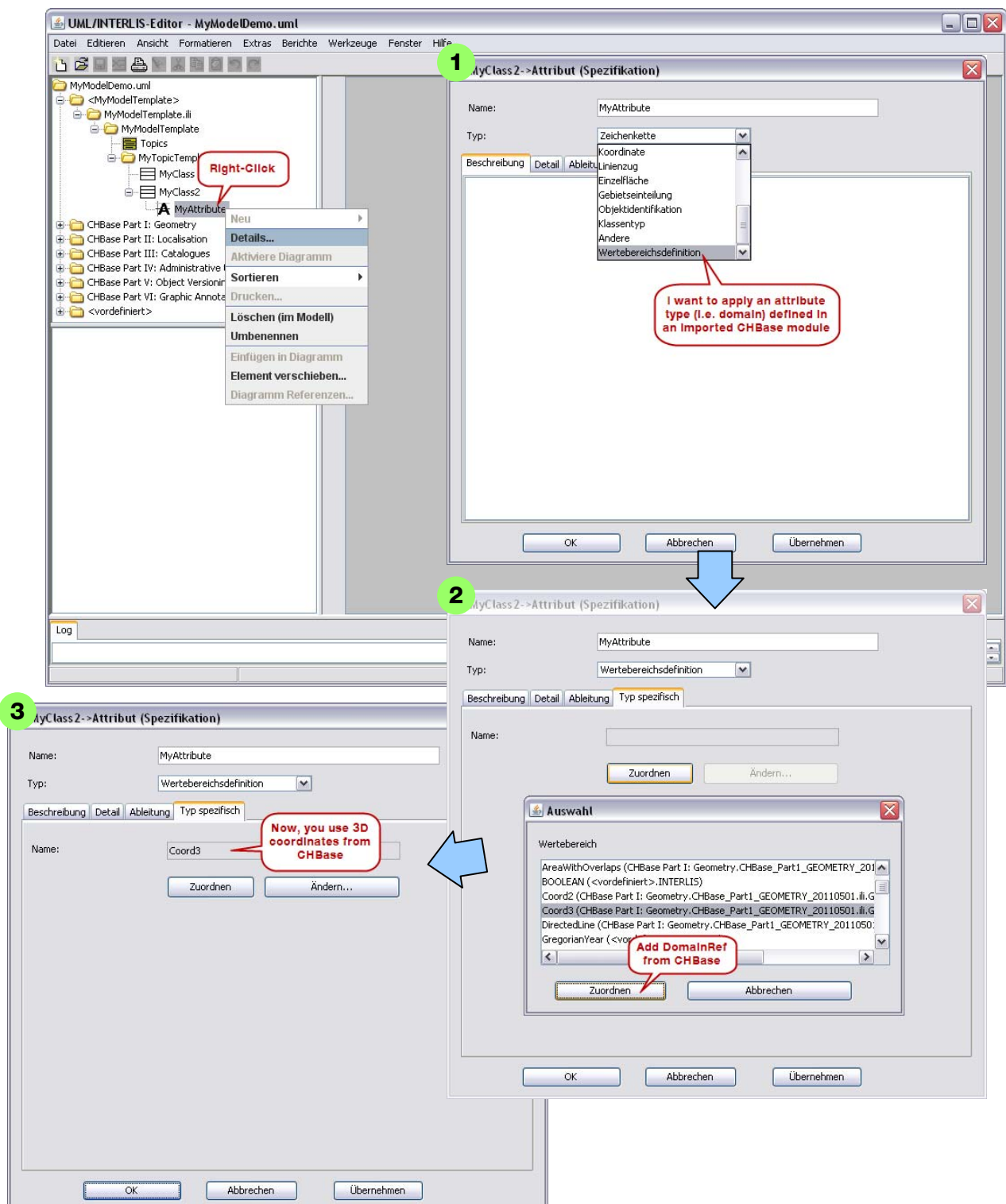


Innerhalb der Klassen (oder Strukturen) werden Attribute erzeugt.



Attribut-Typen aus Modulen verwenden

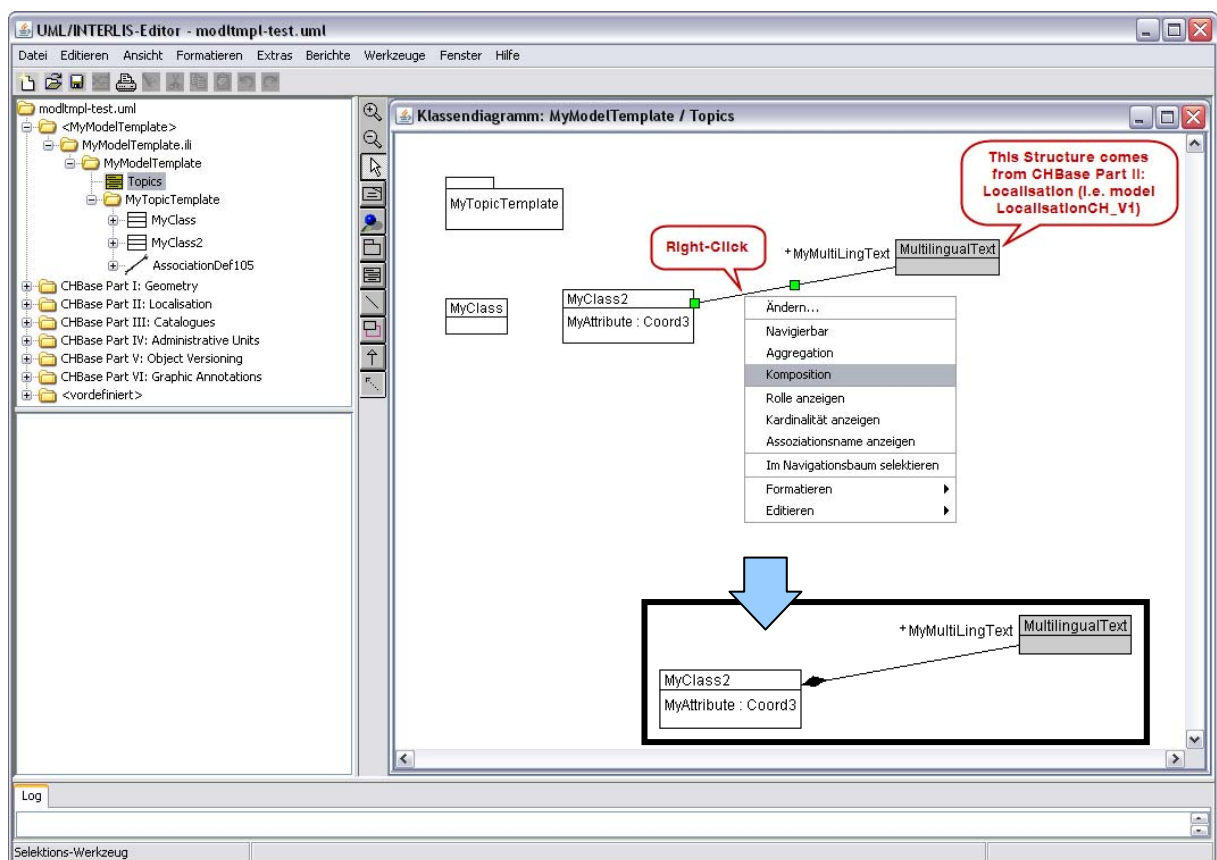
Wenn Wertebereichsdefinitionen als Attributtypen in einem MGDM verwendet werden möchten, so kann durch *Zuordnung* in der Attribut-Spezifikation eine entsprechende Definition aus einem Basismodul ausgewählt werden. Hier wird das Beispiel der 3D-Koordinaten Coord3 aus dem Basismodul GeometryCHLV95 dargestellt.



Strukturattribute aus Modulen verwenden

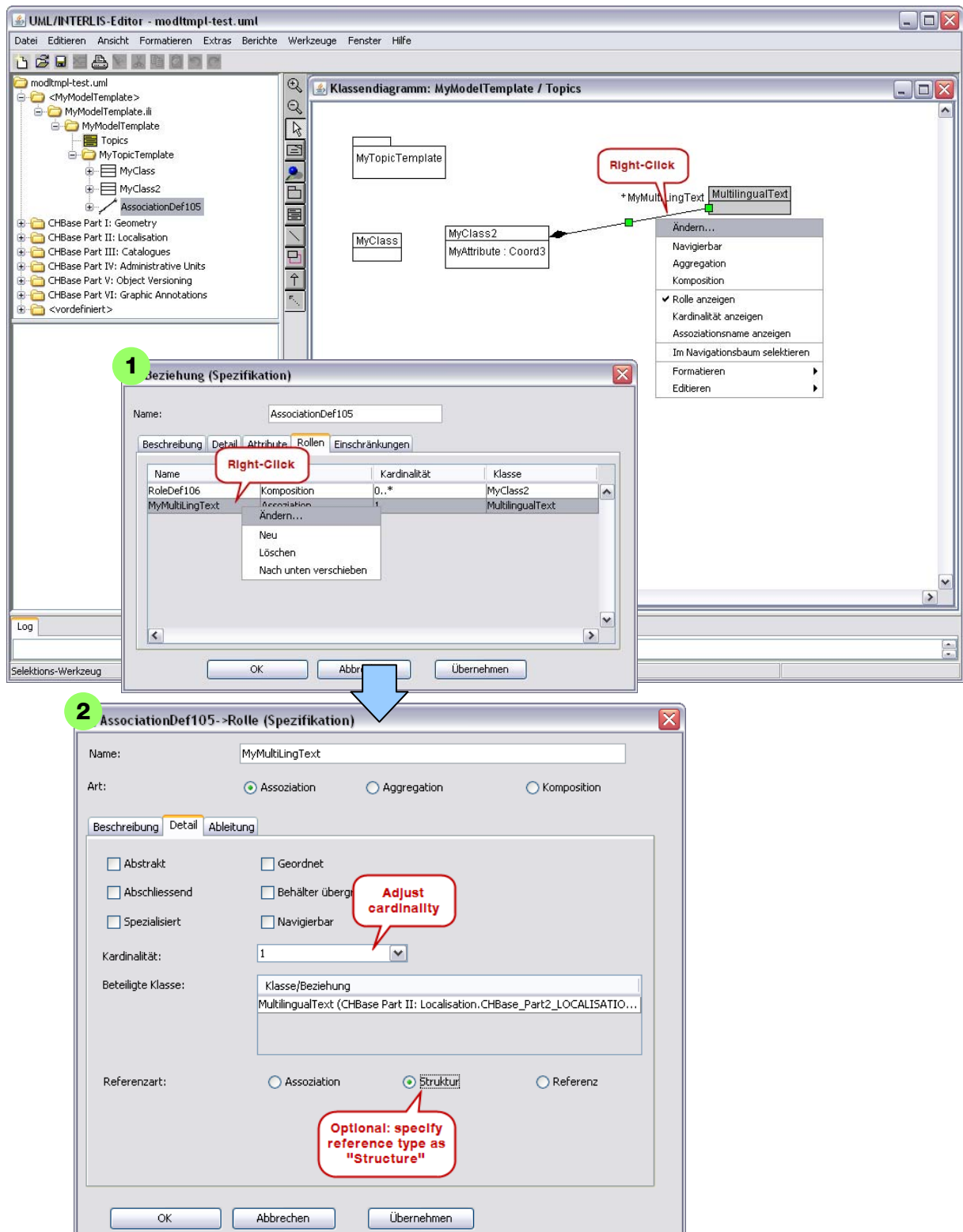
Wenn Strukturattribute als Attributstypen in einem MGDM modelliert werden sollen, dann ist dies im INTERLIS-Code relativ einfach zu erreichen, indem die entsprechende Struktur (oder ggf. auch Klasse) als Attributstyp verwendet wird (`MyAttr: MyStructure;`). Im UML/INTERLIS-Editor müssen Strukturattribute als *Kompositionen* modelliert werden. Dabei ist die Struktur *quasi* existenzabhängig von der Klasse, zu der sie «gehört». Weil die Komposition von einer Struktur ausgeht und nicht von einer Klasse, wird beim Export des INTERLIS-Codes korrekt ein Strukturattribut in der Klasse kodiert. Im Anwendungsbeispiel soll das neue Attribut *MyMultiLingText* der Klasse *MyClass2* ein mehrsprachiger Text sein. Zur besseren Anschaulichkeit wird die Struktur *MultilingualText* aus dem Basismodul *LocalisationCH* in das Diagramm eingefügt und grau markiert (den Modellimport von *LocalisationCH* nicht vergessen!).

Dann wird eine Assoziation gezeichnet, wobei jene Rolle, die zur Klasse *MyClass2* zeigt, als Komposition spezifiziert wird.



Die Rolle der Komposition, welche auf die Struktur *MultilingualText* zeigt, erhält den gewünschten Namen des Strukturattributs im MGDM; im Beispiel «MyMultiLingText».

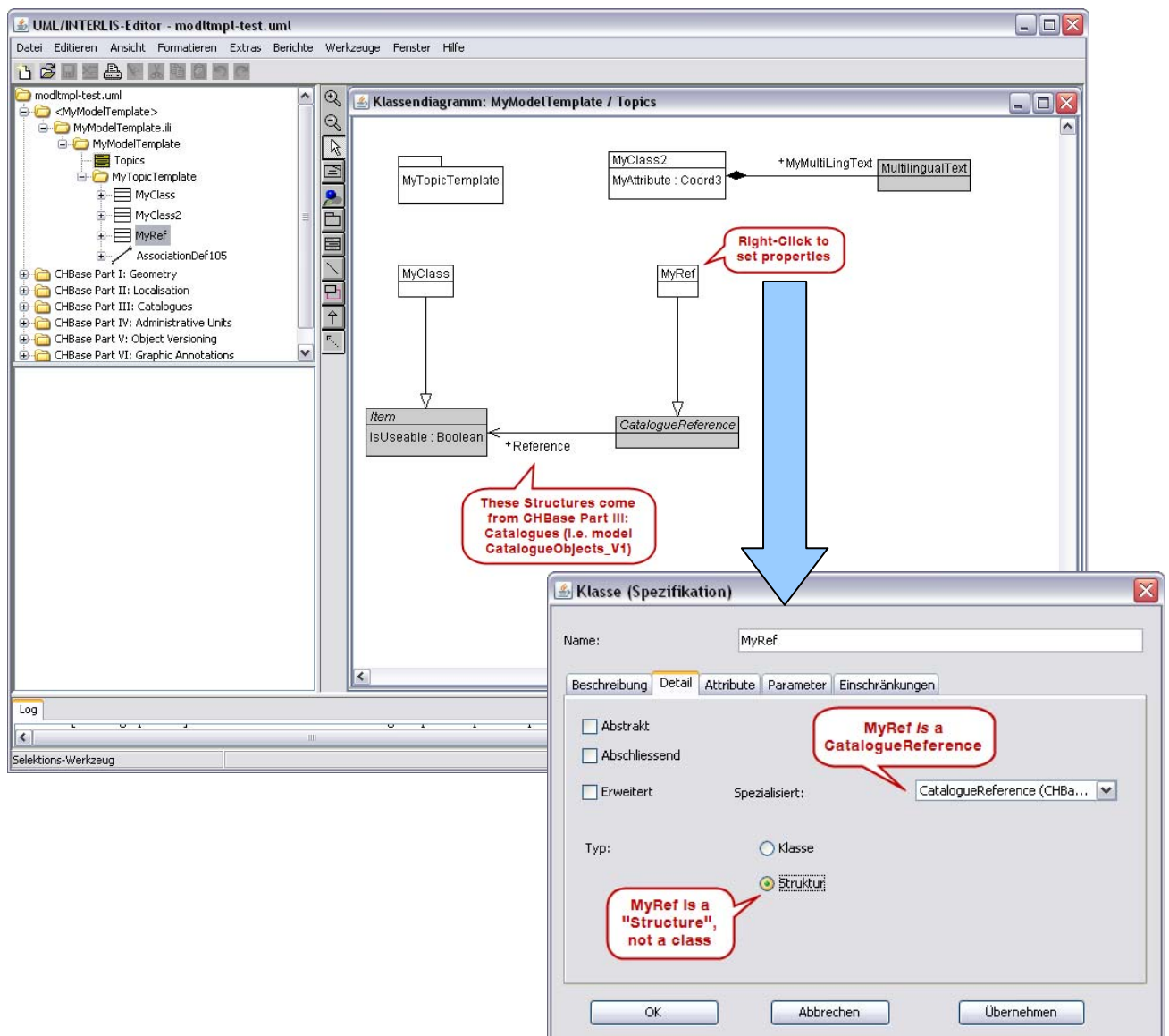
Nun wird noch die Kardinalität des Strukturattributs spezifiziert (normalerweise 1), und es kann (muss aber nicht) angegeben werden, dass diese Rolle eine Struktur referenziert. Im INTERLIS-Export wird aber das Strukturattribut auch dann korrekt kodiert, wenn diese Wahl nicht explizit getroffen wird.

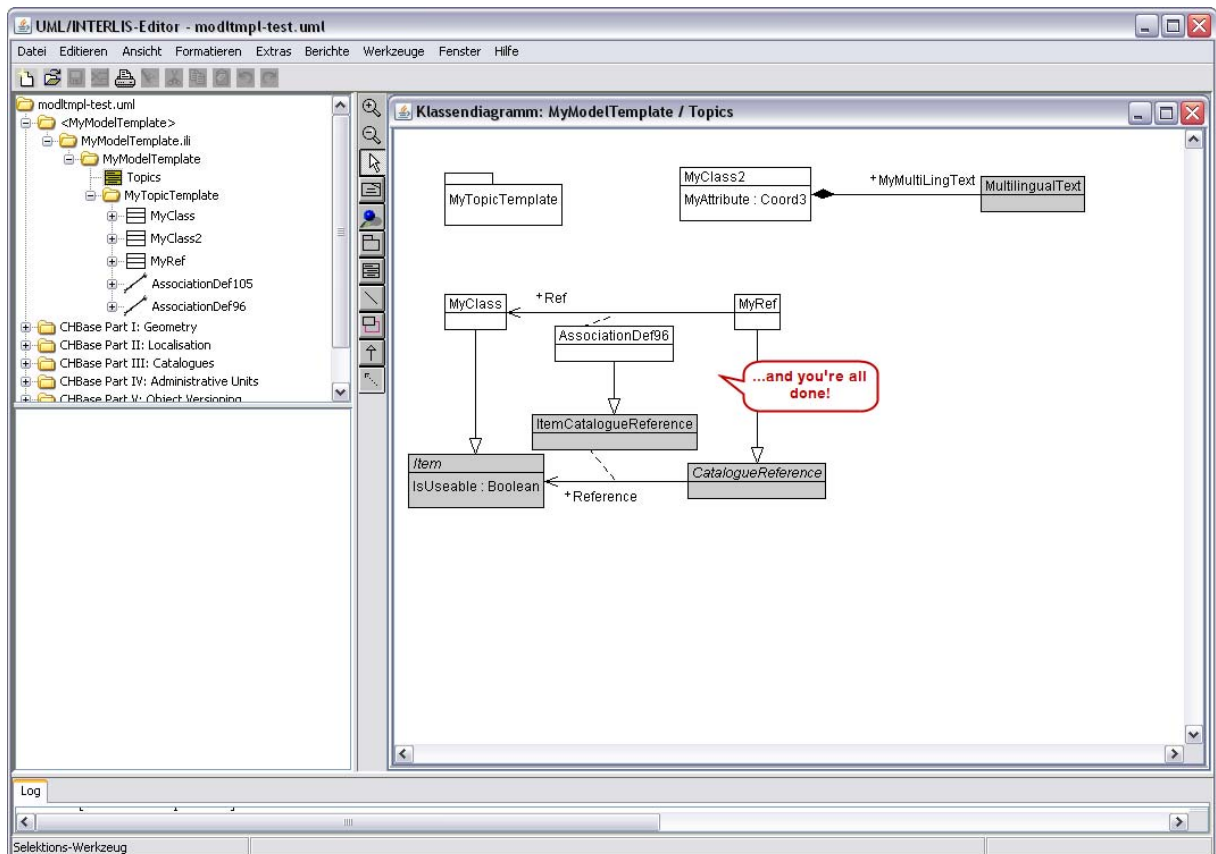
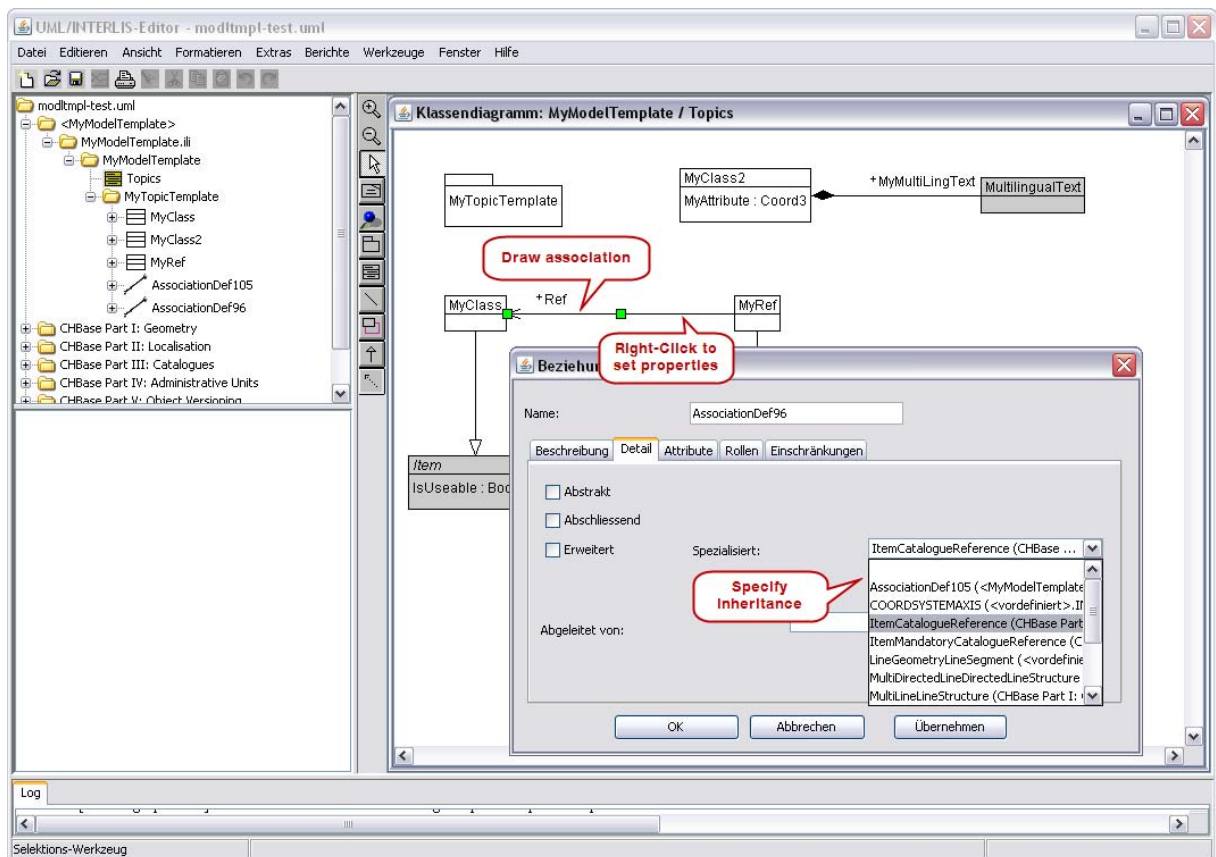


Modellieren von Referenzattributen

Die lose Koppelung von Katalogen und ggf. Übersetzungslisten wird durch *Referenzattribute* hergestellt. Ein Referenzattribut kann aber nur in einer Struktur modelliert werden [6]. Solche Strukturen werden dann als Strukturattribute – wie oben beschrieben – in einer konkreten Modellklasse modelliert. Referenzattribute werden im Editor als gewöhnliche Assoziationen modelliert respektive spezialisiert. Dadurch, dass hier die eine Rolle auf eine Struktur zeigt, wird wiederum beim INTERLIS-Export alles korrekt kodiert, nämlich als «REFERENCE TO *MyClass*».

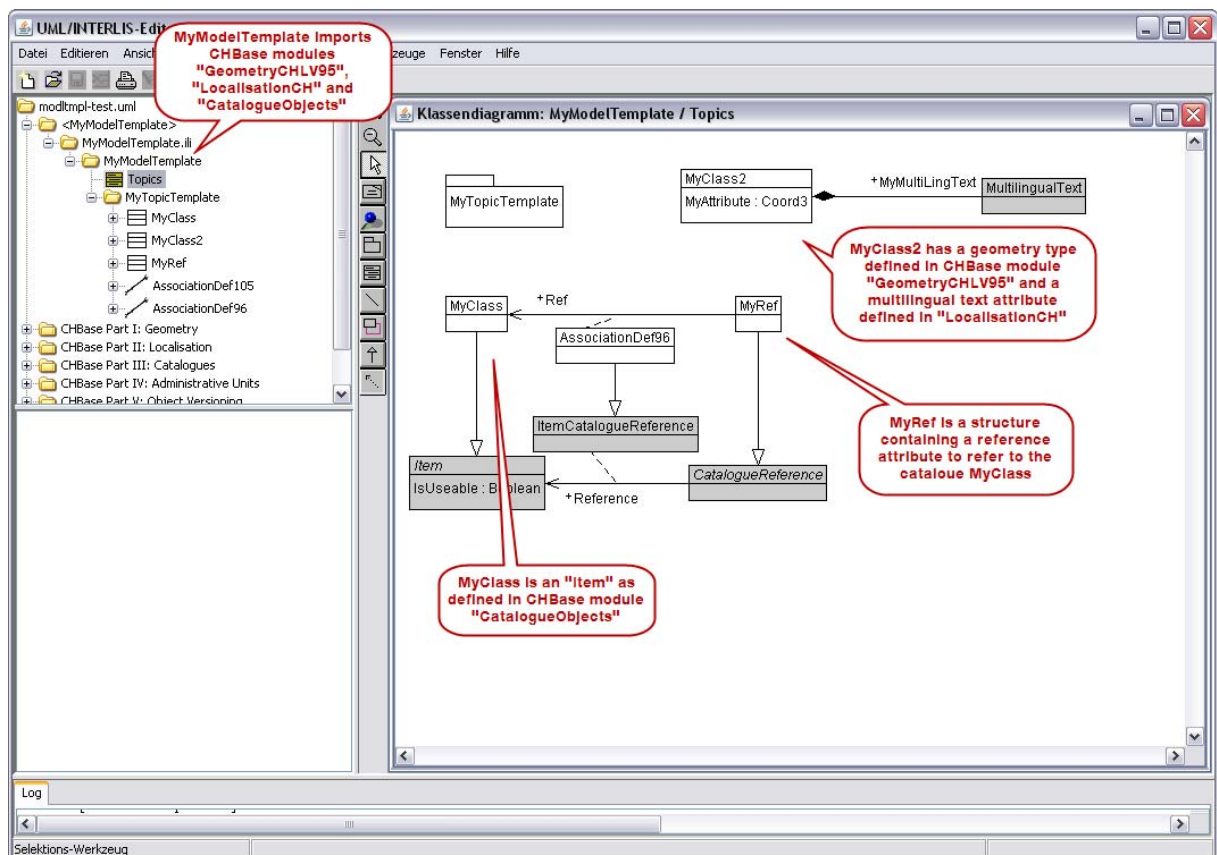
Die Klasse *MyClass* ist eine Spezialisierung der CHBase-Klasse *Item*. Um nun einen vollständigen Katalog zu definieren, muss auch noch die Struktur *CatalogueReference* spezialisiert werden, damit die Referenzierung über das Referenzattribut möglich wird (Struktur *MyRef*):





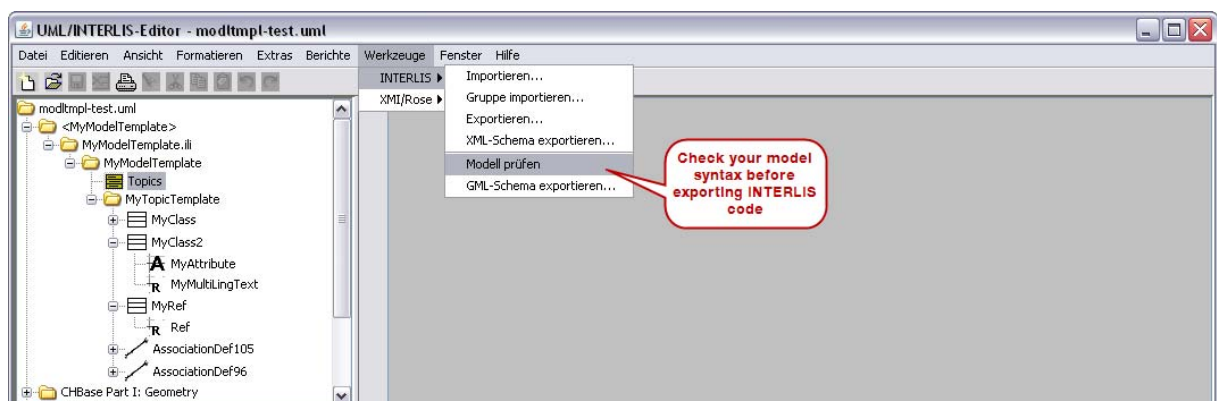
Übersicht

Die Übersicht zeigt noch einmal die Anwendung der oben beschriebenen Mechanismen Modellimport, Klassenvererbung, Wertebereichsdefinition als Attributstyp, Strukturattribut und Referenzattribut.

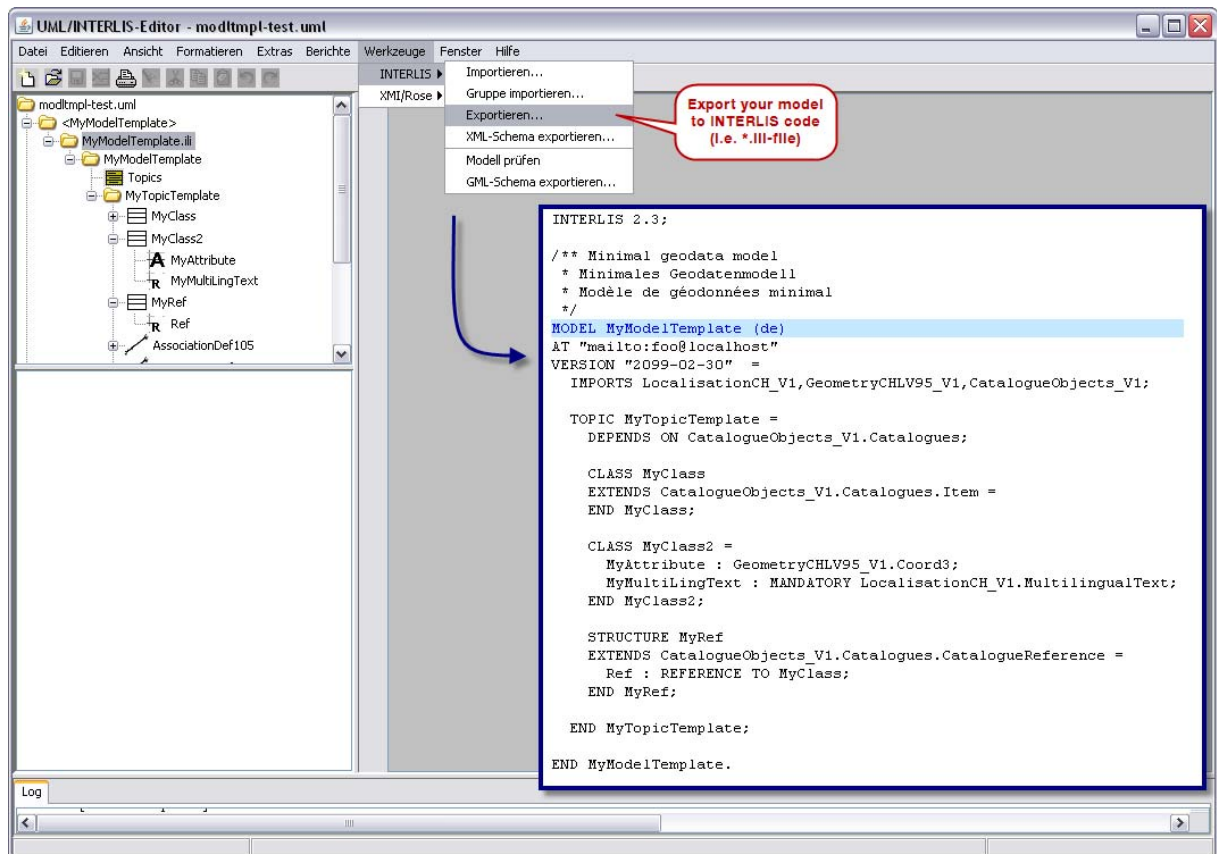


Modell prüfen und als INTERLIS-Code exportieren

Bevor aus dem Modell INTERLIS-Code erzeugt wird, kann die Syntax geprüft werden. Der eingebaute INTERLIS-Compiler wird hierzu verwendet.



Schliesslich wird aus dem UML-Modell die textuelle Form des konzeptionellen Datenmodells generiert, der INTERLIS-Code. Dieser gehört ebenso zur Modelldokumentation wie die grafische Darstellung des MGDM in der Form von UML-Klassendiagrammen.



Hinweis: der UML/INTERLIS-Editor speichert die exportierte *.ili-Datei automatisch am gleichen Ort ab wie die UML-Datei. Dies ist bei bestehenden Dateien zu beachten, da solche ohne Rückfrage überschrieben werden.