

Werkzeuge zur Geodaten-Schema-Abbildung

Informatik-Seminar Sommersemester 2002

Version: 1.3 vom 9. Juli 2002

Autor: Patrick Ulrich, 199b

Betreuer: Professor Stefan F. Keller

Inhaltsverzeichnis

1	Hintergrund	2
2	Problemstellung und Ziele	2
3	Modellzuordnung: Was ist das?	2
3.1	Theorie	2
3.2	Allgemeines Szenario	3
3.3	Szenario 1: Konzeptionelles zu internes Schema	3
3.4	Szenario 2: View auf einen Datensatz.....	4
3.5	Szenario 3: Datenkonversion mit Konstanten	4
3.6	Szenario 4: Klassen- Attributenamen umbenennen	4
3.7	Szenario 5: Datentypen konvertieren	4
4	Schema Mapping-Notationen (Sprachen)	4
4.1	Grafische Sprachen	4
4.2	Textuelle Sprachen	5
5	Vorhandene Softwarewerkzeuge	8
5.1	FME Workbench	8
5.2	InfoGrips INTERLIS-Tools	9
5.3	BizTalk	10
5.4	Rational Rose	11
5.5	Weitere.....	11
6	Vergleich	11
6.1	Tools	11
6.2	Sprachen.....	12
7	Fazit und Ausblick	13
7.1	Fazit: Allgemeine Beurteilung der Problematik.....	13
7.2	Ausblick.....	13
8	Quellenangaben	13
Anhang A		14
Anhang B		15

1 Hintergrund

Das Thema dieser Arbeit ist aus einem Problem entstanden, das nicht nur im Bereich der Geo-Informationssysteme (GIS) grosse Bedeutung hat. Das sogenannte "Schema Mapping" ist überall dort notwendig, wo das Quellschema nicht dem Zielschema entspricht und wo Daten ausgetauscht werden. "Schema Mapping" bedeutet, dass ein bestehendes Schema in ein neues Schema umgewandelt wird; "Mapping" (englisch) im Sinne von logischer Zuordnung zweier Schemas, bzw. Modellierungselemente zweier Schemas (z.B. alle Objekte der Klasse M1_A zu Objekten der Klasse M2_A).

In der Praxis kann dieser Fall zum Beispiel eintreten, wenn die Geodaten einer bestimmten Region zwischen zwei Vermessungsbüros ausgetauscht werden müssen. Oder wenn z.B. zwei Personen ihre Adressen austauschen wollen und im Quellschema Name und Vorname zusammen in einem Attribut verwaltet werden und im Zielschema Name und Vorname je als eigenes Attribut erwarten. Das Problem tritt auch auf, wenn man ein konzeptionelles Schema (typischerweise als UML-Diagramm beschrieben) in ein internes Schema (ein Datenbankmodell) umwandeln muss. In diesem Falle müssen z.B. assoziierte Klassen in Tabellen mit Identifikatoren und Primär-/Fremdschlüssel abgebildet werden. Es kann aber auch sein, dass eine Tabelle in mehrere aufgeteilt werden muss oder umgekehrt. Damit diese und andere Abläufe verständlich dokumentiert werden und vereinfacht werden können, benötigt man eine Sprache, mit der Daten(-Schemas) systematisch in ein anderes Datenschema umgewandelt werden können. Da man nicht erwarten kann, dass Personen, die mit diesem Problem konfrontiert werden, programmieren können, sollte das Ganze mit einem geeigneten Tool durchgeführt werden können.

2 Problemstellung und Ziele

Die Aufgabe dieses Seminarbeitrags ist es, sich mit der Problematik des Schema Mappings auseinanderzusetzen und einige Beispiele von textuellen Sprachen zu diesem Thema anzusehen und zu versuchen diese zu verstehen. Natürlich sollten die Leser dieses Berichtes danach ein gewisses Verständnis für die Problematik haben und einige Tools kennen. Ein weiteres Ergebnis des Berichtes sollte ein Vorschlag sein für die Herstellung eines solchen Tools, beziehungsweise einer solchen Sprache.

3 Modellzuordnung: Was ist das?

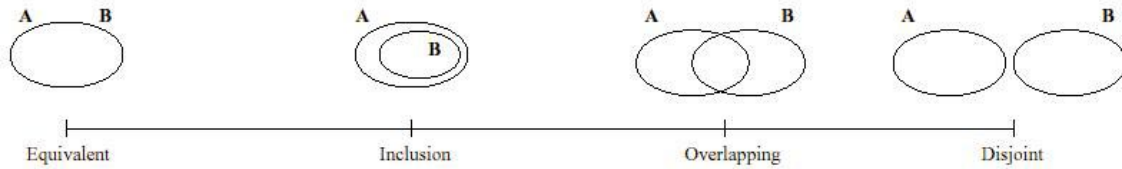
Bei näherer Betrachtung des Problems merkt man, dass es einige Szenarien gibt mit denen ein grosser Teil den in der Praxis auftretenden Problemen abgedeckt sind.

3.1 Theorie

Werden zwei Datensätze miteinander verglichen, können vier verschiedene Möglichkeiten eintreten (vgl. auch Figur 1).

1. Die zwei Datensätze sind völlig identisch (engl. Equivalent).
2. Der Datensatz B ist eine Teilmenge vom Datensatz A (engl. Inclusion).

3. Die beiden Datensätze haben nur eine gemeinsame Schnittmenge (engl. Overlapping).
4. Die Datensätze haben keine Gemeinsamkeiten (engl. Disjoint).



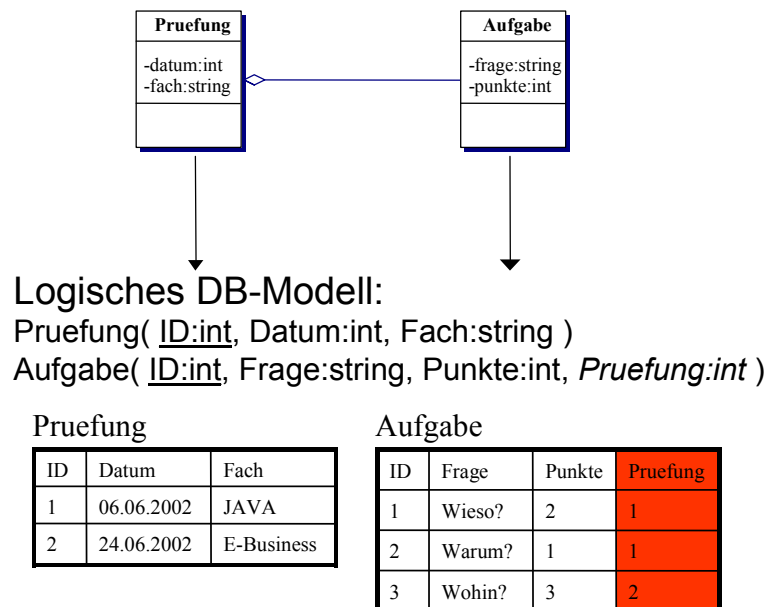
Figur 1 Verhältnisse zwischen zwei Datensätzen

3.2 Allgemeines Szenario

Ein User A bekommt von seinem Kollegen der Partnerfirma Daten. Diese Daten soll der User A nun in die firmeninterne Datenbank einspeisen. Das Problem ist nun, dass die Datenbankschemas der beiden Firmen nicht miteinander übereinstimmen. Das hat zur Folge, dass die Daten angepasst werden müssen. Das wird mit Hilfe von Daten-Schema Mapping gemacht. Dazu entwickelt der User A mit Hilfe einer Sprache oder eines Tools die notwendigen Mapping-Scripts. Mit Hilfe dieser Scripts werden die Quelldaten so angepasst, dass sie ins Zielschema passen. Jetzt kann der User A die Daten in die eigene Datenbank transferieren.

3.3 Szenario 1: Konzeptionelles zu internes Schema

Ein vorgegebenes konzeptionelles Modell muss in ein Datenbank-Schema umgewandelt werden.



Figur 1 - Schema Mapping: Konzeptionelles in internes Datenschema.

3.4 Szenario 2: View auf einen Datensatz

Von einem bestimmten Datensatz will man nur einen Teil der Daten sehen oder transferieren. Zum Beispiel will man von einem Geodatensatz nur die Strassen und Wege; der Rest der Daten ist nicht interessant für den Anwender. Mit anderen Worten: die ursprünglichen Daten werden gefiltert.

3.5 Szenario 3: Datenkonversion mit Konstanten

Die Daten einer Adressdatenbank (gemäss Quellschema) werden in eine neue Adressdatenbank (gemäss Zielschema) übernommen. Im Zielschema soll immer "Schweiz" ins Zielattribut "Land" eingetragen werden. Zusätzlich soll bei allen Adressen ein neues Attribut Vorwahl hinzukommen das mit "041" aufgefüllt wird.

Bei diesem Beispiel sind zwei neue Attribute mit den konstanten Werten "Schweiz" und "041" eingetragen worden.

3.6 Szenario 4: Klassen- Attributenamen umbenennen

Ein weiterer Fall, der in der Praxis auftaucht, ist das Umbenennen von Klassen- und/oder Attributnamen. Es kann zum Beispiel sein, dass im Quellschema eine Tabelle Haus heisst, im Zielschema aber die gleiche Tabelle Gebäude genannt wird. Dasselbe kann in mehrsprachigen Gegenden vorkommen, z.B. Haus und Maison. Die gleiche Problematik gibt es auch mit den Attributen, zum Beispiel ID in Oid umwandeln.

3.7 Szenario 5: Datentypen konvertieren

Die Konversion von Datentypen ist ein weit verbreitetes Problem. Einfacher ist es mit Fließkommazahlen-Datentypen wie float oder double. Kompliziert wird es bei Aufzähl-Typen. So kann zum Beispiel im Quellschema ein Baum vom Typ 1 oder 2 sein. Im Zielschema ist es aber möglich, dass die Werte 1 und 2 als "Laubbaum" und "Tannenbaum" erwartet werden.

4 Schema Mapping-Notationen (Sprachen)

Man unterscheidet grafische und textuelle (nicht-grafische) Notationen.

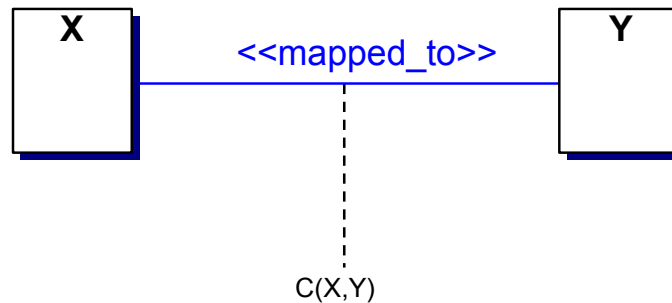
4.1 Grafische Sprachen

4.1.1 UML

Auch mit UML lässt sich Schema Mapping beschreiben [UML]. Laut [Akehurst], ist es möglich, mit Hilfe von Constraints (OCL, [OCL-Tutorial]) und Stereotypes (vgl. die Syntaxelemente mit "<< ... >>") Schema Mapping zu beschreiben.

Beispiel: Die Daten vom Objekt X sollen in das Objekt Y geschrieben werden. X hat die Attribute xCoord und yCoord, Y hat hingegen die Attribute x und y; in beiden Fällen repräsentieren die Attribute die Koordinaten im XY-Koordinatenfeld. Das Constraint C(a, b) für dieses Beispiel könnte etwa so aussehen:

X.allInstances->forAll(X, Y | Y.y = X.xCoord and Y.x = X.yCoord)



Figur 2 - Schema Mapping mit UML und OCL.

4.1.2 Weitere grafische Sprachen

Es wurden keine weiteren grafischen Sprachen untersucht, da der Schwerpunkt nicht bei den Sprachen liegt, sondern bei den Tools.

4.2 Textuelle Sprachen

4.2.1 iG/Script

InfoGrips hat für ihre Tools eine eigene Scriptsprache entwickelt, mit der es möglich ist, Datenschemas zu "mappen". Hier ein kleiner Einstieg in die Scriptsprache:

Am Anfang des Scripts sind einige Konfigurationsparameter angegeben, die vom Tool selber generiert werden. Danach gibt es die sogenannten MAP's, hier sind die Attribute, beziehungsweise Symbole die man zu den Symbolen hinzufügen kann (im DXF sind das DXF_POINT, DXF_BLOCK, DXF_SHAPE, DXF_CIRCLE, DXF_TEXT, DXF_LINE/ARC, DXF_POLYLINE, DXF_PROCEDURE).

Bsp.:

```

MAP TEXT_SYMBOLGY
TS_1 => BRUNNEN, DASHDOT, 8, 1, STANDARD,,0,0,0 !TS_1 ist der Name
END_MAP ! des neuen Attributes

```

Danach wird die Quelle (Input Source) definiert.

```

MAP INPUT_SOURCES
I1 => ILTOPO, OPT.input
END_MAP

```

Jetzt werden den Objekten der Quelle, die jetzt den Namen I1 hat, die neuen Attribute beziehungsweise die Veränderungen der Objekte beschrieben. In diesem Beispiel heisst das, dass jedem Punkt ein TEXT_SYMBOLGY, mit dem Wert "Point" angehängt wird.

```

MAP INOUT
I1 => IN.TOPIC, IN.TABLE
I1,DXF_ENTITIES,DXF_POINT => T_1, TS_1
I1,DXF_ENTITIES,DXF_POLYLINE => OFF
usw.
END_MAP

```

Am Schluss werden die Makros definiert, die die geografische Lage des neuen Symbols beschreiben.

```

MAP MACRO
T_1 => DXFOUT_WRITE_TEXT6, Point, IN.GEOM, 0,,
END_MAP

```

Ganz am Schluss werden noch die Includes angegeben; das sind Basis-Module, die von der Firma vorgegeben sind.

4.2.2 EXPRESS-X

STEP (STandard for the Exchange of Product Data) ist ein internationaler Standard zur Beschreibung physikalischer und funktionaler Merkmale von Produktdaten [EXPRESS_X]. EXPRESS ist vor allem in der Automobilindustrie verbreitet. EXPRESS-X ist ein Teil von STEP und ist eine Sprache mit der ein bestehendes Quellmodell in ein neues Zielmodell umgewandelt werden kann. Das Quell- sowie das Zielmodell müssen in der EXPRESS-Sprache definiert sein.

Im folgenden Beispiel, wird das Schema source in das Schema target umgewandelt. Im Quell-Schema gibt es die ENTITY's, **person**, **man**, **woman**, **child**. Eine **person** hat - je nach dem ob es ein Mann oder eine Frau ist - zusätzliche Attribute. Eine Frau z.B. enthält eine Liste mit **child**'s. Im Zielschema werden nur die Daten der Männer benötigt. Zusätzlich werden die Attribute **social_security_number** und **car**, in **id** bzw. **vehicle** umgewandelt.

```
(*EXPRESS Notation*)
SCHEMA source;
TYPE m_or_f = SELECT (man, woman);
END_TYPE;

ENTITY person;
    social_security_number : STRING (8) fixed
    name : STRING;
    age : REAL;
    data : m_or_f;
END_ENTITY;
ENTITY man;
    car : STRING;
END_ENTITY;
ENTITY woman;
    children : LIST [0:?] of child;
END_ENTITY;
ENTITY child;
    (*Attrinutes of a child*)
END_ENTITY;

SCHEMA target;
ENTITY male;
    id : STRING
    age : REAL;
    vehicle : STRING
END_ENTITY;
```

(*EXPRESS-X Notation*)

```
SCHEMA_MAP Mapping_Schema;
GLOBAL
    DECLARE sdb INSTANCE OF source;
    DECLARE tdb INSTANCE OF target;
END_GLOBAL;
VIEW v1 : tdb::male;
FROM( m : sdb::man ) WHEN TRUE;
BEGIN_VIEW
```

```
id:= m.social_security_number;  
vehicle := m.car  
END_VIEW;  
END_SCHEMA_MAP;
```

4.2.3 XSLT (Extensible Stylesheet Language Transformations)

XSLT ist eine Sprache die auf XML basiert und darauf ausgerichtet ist, XML-Dokumente in ein anderes Textdokument umzuwandeln [XSLT]. Dazu gehört natürlich auch Schema Mapping, das Eingabeformat ist mit XML fest. Das Ausgabeformat ist relativ frei wählbar, das heisst meistens ist es wieder XML oder HTML (XHTML).

Beispiel: Die Instanzen der Klasse **division** haben eine **id** und die Attribute **revenue** und **bonus**. Diese Daten werden jetzt mit Hilfe von XSLT ins HTML-Format umgewandelt. Das heisst es wird eine Tabelle generiert, in der die Daten hineingeschrieben werden.

(*XML Daten*)

```
<sales>  
<division id="North">  
<revenue>10</revenue>  
<bonus>7</bonus>  
</division>  
<division id="South">  
<revenue>4</revenue>  
<bonus>4</bonus>  
</division>  
</sales>
```

(*XSLT Mapping Code*)

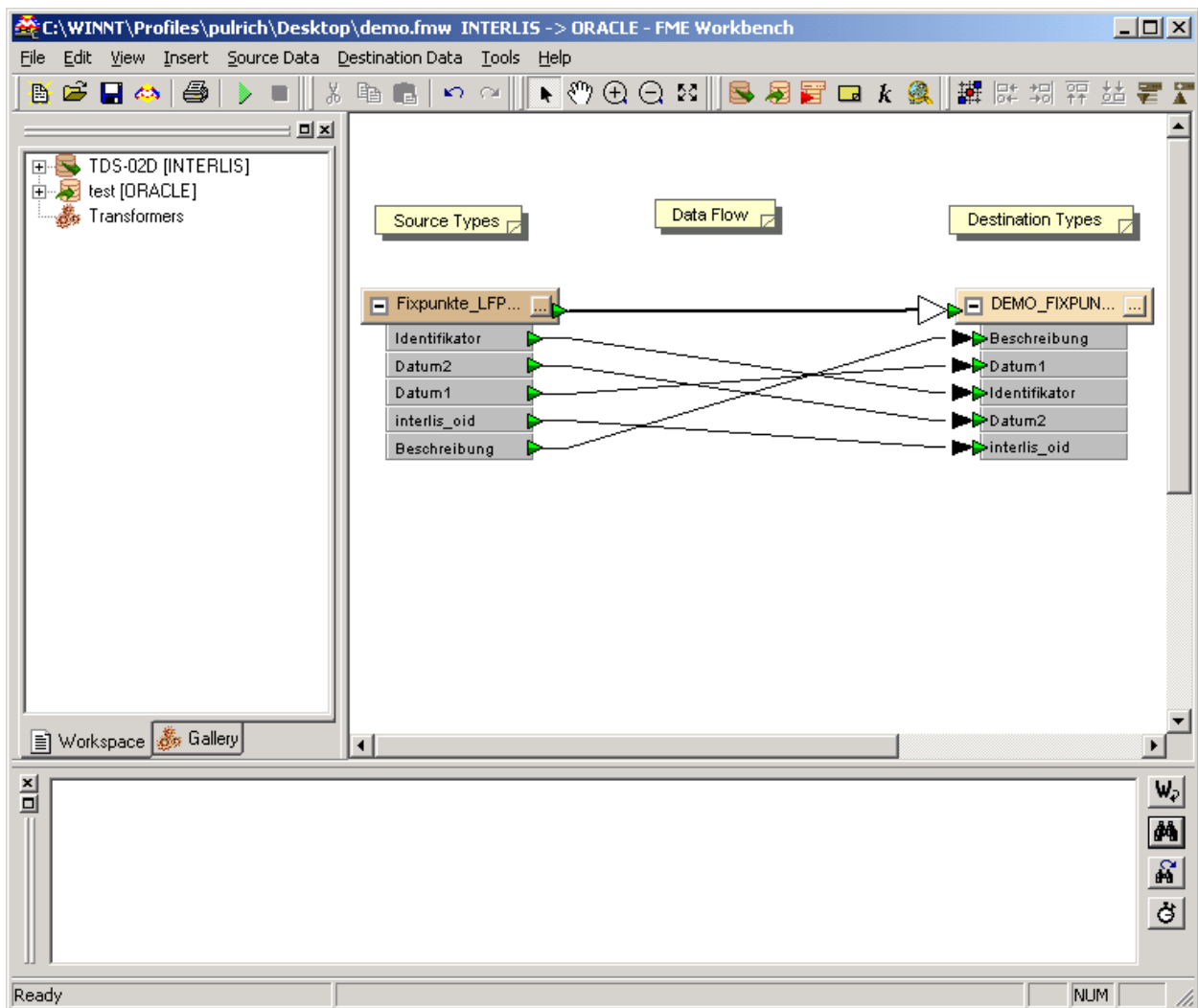
```
<xsl:for-each select="sales/division">  
  <!-- order the result by revenue -->  
  <xsl:sort select="revenue"  
    data-type="number"  
    order="descending"/>  
  <tr>  
    <td>  
      <xsl:value-of select="@id"/>  
    </td>  
    <td>  
      <xsl:value-of select="revenue"/>  
    </td>  
    <td>  
      <xsl:value-of select="bonus"/>  
    </td>  
  </tr>  
</xsl:for-each>
```

5 Vorhandene Softwarewerkzeuge

5.1 FME Workbench

Mit dem FME Workbench [FME Workbench] ist es möglich, die Schemas anhand eines Datensatzes zu lesen und dann in ein anderes beliebiges Format um zu wandeln. Das Ganze ist auf einem grafischen User Interface aufgebaut, mit dem man gut arbeiten kann.

Auf dem Bild unten sieht man die Source- und die Destinationstabelle, die Attribute bleiben die gleichen, ihre Reihenfolge werden aber geändert, was zum Beispiel in einer Datenbank wesentlich sein kann. Es ist aber auch möglich, die Werte eines Attributes zu ändern, so könnte man zum Beispiel beim Attribut Datum2 das aktuelle Datum einfügen. Man kann mit dem FME Workbench die Daten, aus denen das Quellschema erstellt wurde, direkt umwandeln. Wenn es erwünscht ist, kann man auch ein Script erstellen mit welchem dann Daten umgewandelt werden können. Dazu braucht es noch den FME Translator, der im FME Workbench-Paket dabei ist.



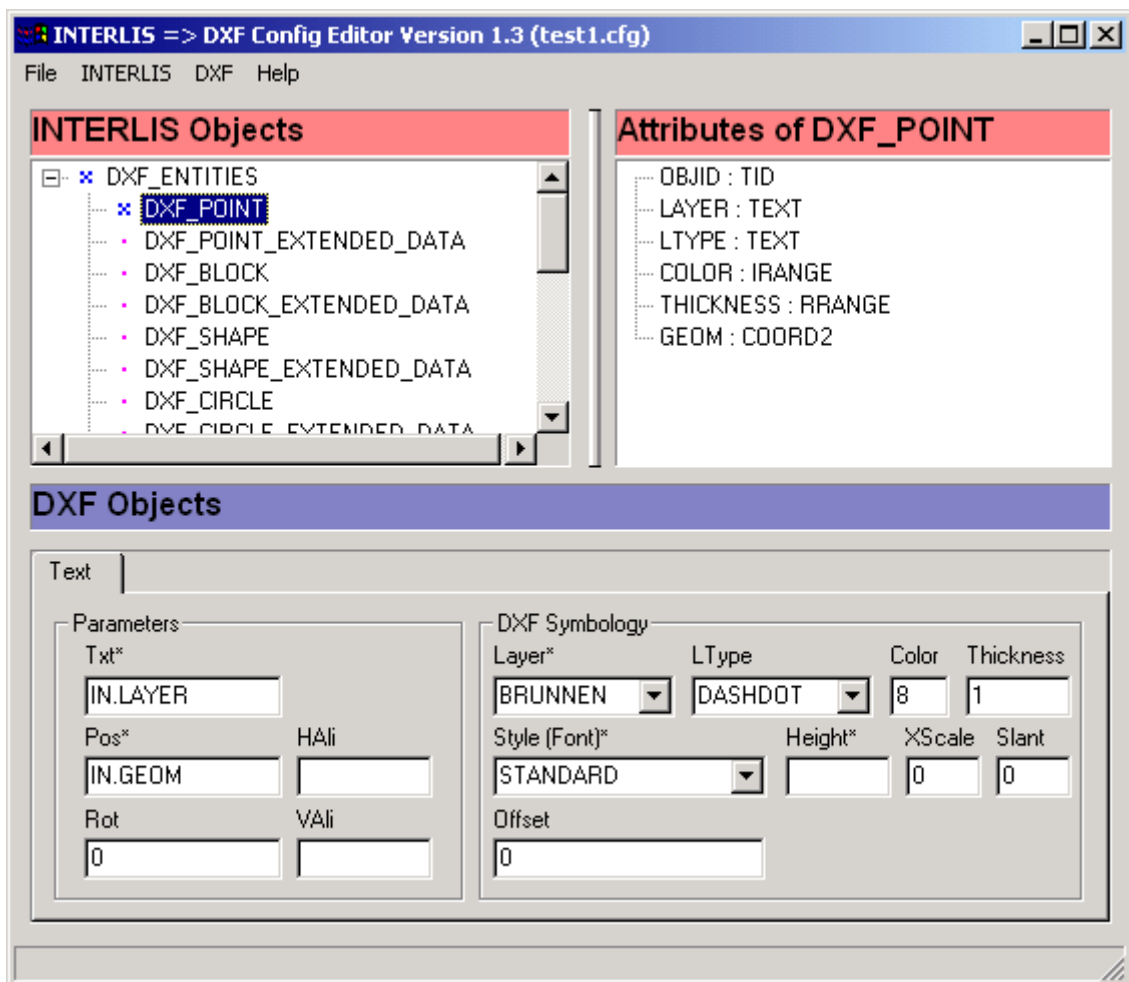
Figur 3 - FME Workbench.

5.2 InfoGrips INTERLIS-Tools

Aus Sicht des Benutzers steht der *Config Editor* im Zentrum, mit Hilfe dessen das Mapping-Schema erstellt wird. Von diesem Tool existieren drei Varianten der INTERLIS => DXF Editor, INTERLIS => SHAPE Editor und der INTERLIS => SDL Editor. Das andere wichtige Tool ist das *ICS for Windows* - mit diesem Programm werden die Daten wirklich konvertiert ("gemappt").

Zuerst muss man den geeigneten Editor wählen. In unserem Beispiel nehmen wir den INTERLIS => DXF Editor. Danach wird ein neues Config-File erstellt (*.cfg-Datei). Dazu muss man die INTERLIS-Quelldatei angeben, dann zeigt der Editor alle Tabellen mit ihren Attributen an.

Wenn man aus dieser Situation eine Config-Datei erstellt, werden allerdings noch keine Daten konvertiert. Das heisst, man muss jedes DXF-Symbol, das man im Zielschema erhalten will, definieren. Dazu kann man im Menüpunkt DXF auf NEW gehen und das passende DXF-Symbol anwählen. Danach werden unter DXF Objects die notwendigen Attribute angezeigt. Um die richtigen Werte aus den INTERLIS-Daten mit Drag&Drop ins richtige Textfeld im Bereich Parameters. Danach kann die Config-Datei gespeichert werden



Figur 4 - Config Editor für INTERLIS nach DXF-Konversionen von InfoGrips.

Der nächste Schritt besteht jetzt darin, mit dem *ICS for Windows* die Daten zu konvertieren. Dazu kann man den Button Script drücken und das dazugehörige Script (Config File, test1.cfg) auswählen. Sobald das Tool die Config-Datei geöffnet hat wird sie automatisch kompiliert. Jetzt kann man die Daten konvertieren, indem man Run drückt, zuerst muss aber natürlich noch die Quell- und die Ziel-Datei

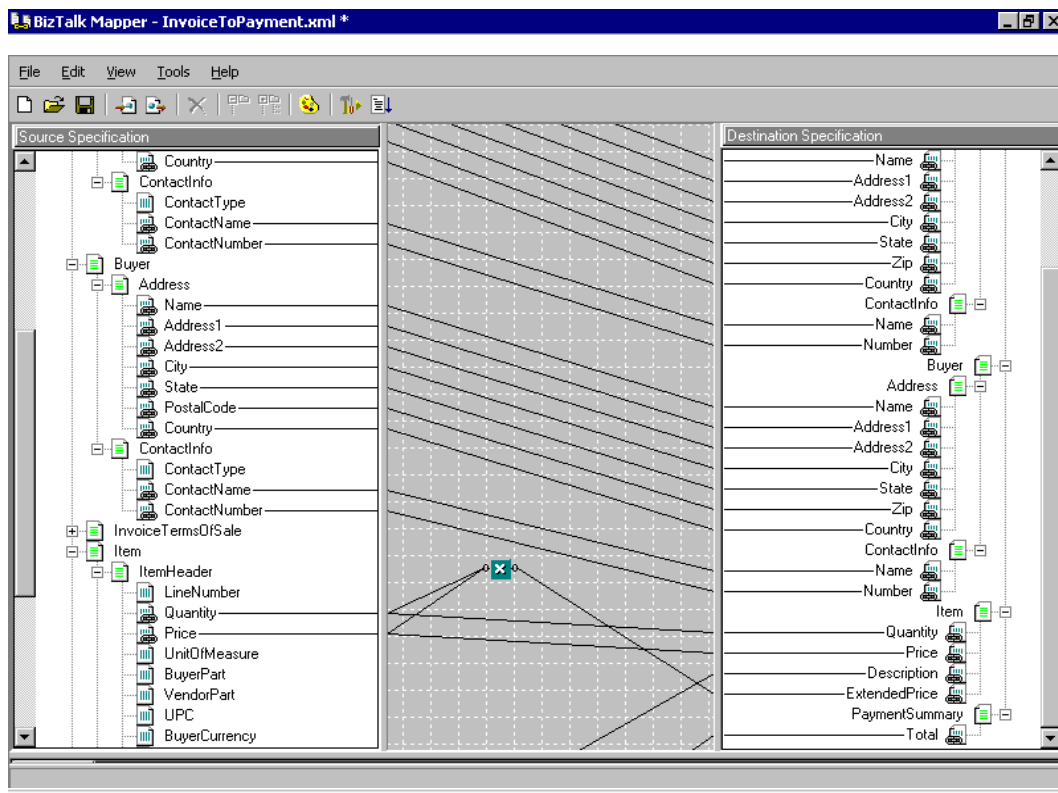
angegeben werden. Im Feld Status Information wird die Anzahl der In- und Outputobjekte angegeben, man sieht dort auch, ob es Fehler gegeben hat. Mit dem Button Show Log kann man das Logfile der Konvertierung ansehen was vielleicht bei Fehlern eine Hilfe sein kann.

5.3 BizTalk

BizTalk ist ein Produkt von Microsoft. Es ist ein Editor, mit dem man Schema-Mappings (oder eher Format-Mappings) definiert und dann XSLT-Skripts erzeugt, die dann von XSLT-Prozessoren interpretiert werden und XML umwandeln. Die Datenformate, die zur Zeit unterstützt werden, sind, XML, EDI (siehe Quellenangaben) und FlatFiles. BizTalk ist aus dem Gedanken entsprungen, dass Daten zwischen verschiedenen Firmen schnell und problemlos ausgetauscht werden können und wie wir mittlerweile wissen, ist dazu fast immer ein Schema Mapping notwendig.

Um Daten in XML umzuwandeln, sind zwei Schritte notwendig:

1. Die Daten im BizTalk Editor spezifizieren. Das heisst, die Datenelemente von, z.B. einer Rechnung oder Offerte, beschreiben.
2. Die Datenbeschreibungen, die mit dem Editor gemacht wurden, können nun im BizTalk Mapper bearbeitet werden. Der Mapper ist ein grafisches Tool (siehe Figur 2). Links sieht man die sogenannte Source Specification und rechts die Destination Specification. In der Mitte sind die Verbindung die zwischen den beiden Spezifikationen gemacht werden. Es ist auch möglich die Daten die in das andere Schema schreiben werden noch anzupassen. So kann man zum Beispiel mehrere Attribute aufsummieren oder die Fläche einer bestimmten Form ausrechnen.



Figur 5 - BizTalk Mapper.

5.4 Rational Rose

Mit dem Rational Rose Editor ist es möglich, das konzeptionelle Modell in das entsprechende Datenbankschema umzuwandeln. Der Begriff "Datenbankschema" ist etwas übertrieben, da das Ergebnis des Umwandelns "nur" SQL Befehle sind, die dann in die entsprechende Datenbank eingespielt werden können.

5.5 Weitere

Des Weiteren wurde InterlisStudio und ERWIN noch angeschaut.

- InterStudio (<http://www.geocom.ch/>) ist vergleichbar mit FME Workbench, verwaltet jedoch seine Konfigurationsdaten in binärem, eigenem Format und bietet auch kein API.
- ERWIN ([ERWIN](#)) ist ein Werkzeug, mit dem man Datenbankmodelle erstellen kann. ERWIN soll zur Zeit keine Möglichkeiten bieten, Schemas zu Mappen. Daher wird in dieser Arbeit nicht weiter auf ERWIN eingegangen.

6 Vergleich

Nachdem wir einige Notationen, grafische sowie textuelle, angeschaut haben, ist es Zeit eine kleine Auswertung zu machen.

Kriterium \ Werkzeuge	textuell oder grafisch	User Interface (1 bis 6, 6 = max)	geeignet für	Preis	Bemerkung
FME Workbench	grafisch	5	Geo- bzw. Grafische-Daten	Support pro Jahr 1'500 Fr.	gutes Interface, kennt viele Dateiformate
InfoGrips INTERLIS Tools	textuell	2	Geodaten	5'000 Fr.	Interface ungenügend und der Datenfluss ist fast nicht ersichtlich
BizTalk	grafisch	6	Businessdaten	\$500 – \$25'000	gutes Interface, vielversprechender Ansatz mit XSLT, dadurch aber auf XML beschränkt
Rational Rose	grafisch	2	UML in Datenschema	\$1794 bis \$5995 pro Lizenz	generiert nur SQL-Code

6.1 Tools

FME Workbench: Dieses Werkzeug ist speziell für Geodaten und grafische Daten erstellt; es kennt auch sehr viele Datenformate (wobei zum Teil auch völlig irrelevante darunter sind). Das User Interface ist gut zu gebrauchen, das

Script, das erstellt wird, ist allerdings nicht einfach zu lesen. Das Programm hat einige Defizite bei der Performance. Schade war allerdings, dass die Demoversion die ich heruntergeladen habe, nicht funktioniert hat; das heisst, es war nicht möglich mit dem erstellten Mapping Script, Daten anzupassen.

- InfoGrips INTERLIS Tools: Ist speziell auf Geodaten ausgerichtet. Das User Interface des Werkzeugs ist nicht sehr intuitiv, aber das Script ist dafür lesbar. Ist aber eher für INTERLIS-Experten zu empfehlen, da es nur die drei oben erwähnten Editoren gibt.
- BizTalk Das Werkzeug, das mir vom User Interface her am Besten gefallen hat. Leider habe ich keine Demoversion bekommen, darum kann ich keine Bewertung abgeben.
- Rational Rose Rational Rose ist konzipiert, um UML-Diagramme zu zeichnen; als Feature wird angeboten, das UML-Schema in ein Datenbankschema umzuwandeln. Das heisst es wird SQL-Code generiert, mit dem die Datenbankstruktur erstellt werden kann. Hat aber keine spezifische Erweiterung für Geodaten; ist nur für UML anwendbar.

6.2 Sprachen

- EXPRESS-X EXPRESS-X hat mir persönlich als Sprache am besten gefallen und zwar weil es sehr leserlich war und mit den drei Schemas (Source, Destination und MAP) klar ersichtlich ist, was wozu gebraucht wird. Der grosse Vorteil von EXPRESS-X ist, dass sich nahtlos in die EXPRESS-Syntax einfügt, mit der man die Schemas selber beschreibt.
- iG/Script Das iG/Script ist mit den Zuordnungs-Pfeilen ("=>") eine Alternative zu EXPRESS-X, ist aber nicht so gut leserlich. Der Nachteil von iG/Script ist, dass es "proprietär" ist, d.h. dass es speziell auf die InfoGrips Tools ausgerichtet ist.
- UML Es ist also möglich, auch mit UML Schema Mapping zu beschreiben, für mich war das aber nicht sehr gut lesbar. Noch schwieriger scheint es allerdings, die Constraints zu schreiben. Dazu muss man sich in die OCL (Object Constraint Language) einarbeiten. So glaube ich, dass es nicht sehr empfehlenswert ist, Schema Mapping mit UML beziehungsweise OCL zu beschreiben.
- XSLT XSLT hat den Nachteil, dass es nur mit XML Daten gefüttert werden kann. Das heisst, man ist an das XML-Format gebunden. Aber da BizTalk damit arbeitet, könnte ich mir vorstellen, dass sich XSLT im Bereich Schema Mapping etablieren kann.

7 Fazit und Ausblick

7.1 Fazit: Allgemeine Beurteilung der Problematik

Nachdem ich mich eingehend mit dem Thema befasst habe, bin ich zu dem Entschluss gekommen, dass diesem Problem in der Zukunft mehr Gewicht zukommen wird. Da Schema Mapping ist nicht "nur" im GIS Bereich ein grosses Problem. Denn überall wo Daten und auch Informationen ausgetauscht werden, müssen die Datenschemas total übereinstimmen oder es muss ein Schema Mapping vorgenommen werden, was in vielen Fällen der Fall sein wird. Da auch Microsoft und IBM das Problem anscheinend erkannt haben und Produkte anbieten oder entwickeln die im Bereich Schema Mapping tätig sind, denke ich, dass in Zukunft viel mehr Tools angeboten werden als bis jetzt.

7.2 Ausblick

Wenn es nur darum geht, eine eigenes Tool zu erstellen, dann glaube ich, dass ich in der Lage bin einen Vorschlag zu machen, mit dem man in der Praxis bestehen könnte.

Mein Vorschlag besteht darin, dass man aus BizTalk und InfoGrips INTERLIS Tools je einen Teil nimmt. Das User Interface hat mir von BizTalk am besten gefallen. Das Interface ist in drei Teile unterteilt, genau so wie das der User im Kof hat: links das Quellschema, in der Mitte der Datenfluss und rechts das Zielschema. Man sieht gut, dass die Daten von links nach rechts fließen. Des Weiteren ist die Baumstruktur für ein solches Tool eine gute Variante, da es die Kapselung der Attribute gut aufzeigt.

Des Weiteren würde ich die Idee von den INTERLIS Tools von Infogrips übernehmen. Nämlich, dass man aus dem Grafik-Tool ein Script erstellt wird mit Hilfe von dem die Daten umgewandelt werden. Die Scriptsprache, die ich nehmen würde, ist EXPRESS-X oder es wird eine selber definiert, die stark an EXPRESS-X angelehnt ist. Diese Sprache ist meiner Meinung nach gut lesbar, da es an High-Level Sprachen angelehnt ist.

8 Quellenangaben

Internet-Adressen der Software und Manuals

Download FME Workbench	http://www.tydac.ch/
BizTalk	von Microsoft http://www.biztalk.org/
InfoGrips INTRLIS Tools	von InfoGrips http://www.infogrips.ch/
RationalRose	von RationalRose http://www.rational.com/rose/
EDI Beschreibung	http://www.glossar.de/glossar

Referenzen

[Akehurst]	David H. Akehurst, Model Translation: A UML-based specification technique and active implementation approach, Doktorarbeit, 2001.
[EXPRESS-X]	http://www.nist.gov/sc4/step/parts/expressx/n002/
[FME-Workbench]	http://www.geotask.ch/
[OCL-Tutorial]	http://neptune.irit.fr/Biblio/ocl_1_4.shtml
[UML]	UML 1.3, OMG, http://www.omg.org/
[XSLT]	http://www.w3.org/TR/xslt

Anhang A

Demonstration von FME Workbench:

1. FME Workbench starten, im Menu File→ NEW, danach Source Format und Dataset sowie Destination Format und Workspacename. Das Quellschema wird aus den Quelldatenerstellt, zusätzlich wird ein Vorschlag für das Zielschema dargestellt.
2. Jetzt können die Tabellen modifiziert werden.
3. Das fertige Schema kann gespeichert werden und mit dem Translate Button(grüner Pfeil) können die Inputdaten in ein Outputfile(muss noch angegeben werden) geschrieben werden.

Anhang B

Demonstration von infoGrips INTERLIS Tools:

1. Geeigneten Editor öffnen (INTERLIS=>DXF Editor), im Menu File→ New INTERLIS Datei und DXF Template angeben.
2. Links werden jetzt alle INTERLIS Objekte angezeigt, wenn die Objekte angezeigt werden, erscheinen rechts die Attribute des jeweiligen Objektes.
3. Jetzt kann im Menu DXF→ New ein Attribute angegeben werden, das im Output-Schema hinzugefügt werden soll. Mit Drag&Drop können die Werte der Attribute des Quellschemas in die Attribute des Zielschemas gezogen werden.
4. Am Schluss, wenn das Config File gespeichert worden ist, öffnet man den ICS for Windows geöffnet werden. In diesem Tool kann man das Config File das wir mit dem Editor erstellt haben angeben.
5. Danach wird der Run Button gedrückt, damit die Daten umgewandelt werden können, müssen zuerst noch die In- und Outputdateien angegeben werden.