Bundesamt für Raumplanung
Office fédéral de l'aménagement du territoire
Ufficio federale della pianificazione del territorio
Uffizi federal da planisaziun dal territori

Eidgenössische Vermessungsdirektion
Direction fédérale des mensurations cadastrales
Direzione federale delle misurazioni catastali
Direcziun federala da mesiraziun

Eidg. Justiz- und Polizeidepartement
Département fédéral de justice et police
Dipartimento federale di giustizia e polizia
Departament federal da giustia e polizia

Kompetenzzentrum INTERLIS/AVS
Centre de compétence INTERLIS/IMO
Centro di competenza INTERLIS/IMU
Center da cumpetenza INTERLIS/IMU

INTERLIS

Ein Datenaustausch-Mechanismus für Land-Informationssysteme

Version 1 Revision 2, März 1999

Inhalt

Abbildungsverzeichnis

1. Überblick

Die Grundidee von INTERLIS besteht darin, dass ein Austausch der Informationen eines Landinformationssystems nur möglich ist, wenn die am Austausch beteiligten Stellen eine genaue und einheitliche Vorstellung über die Art der auszutauschenden Daten haben. INTERLIS befasst sich deshalb zunächst mit der genauen Beschreibung des Datenmodells und erst in einem zweiten Schritt mit der Festlegung des Austauschformates.

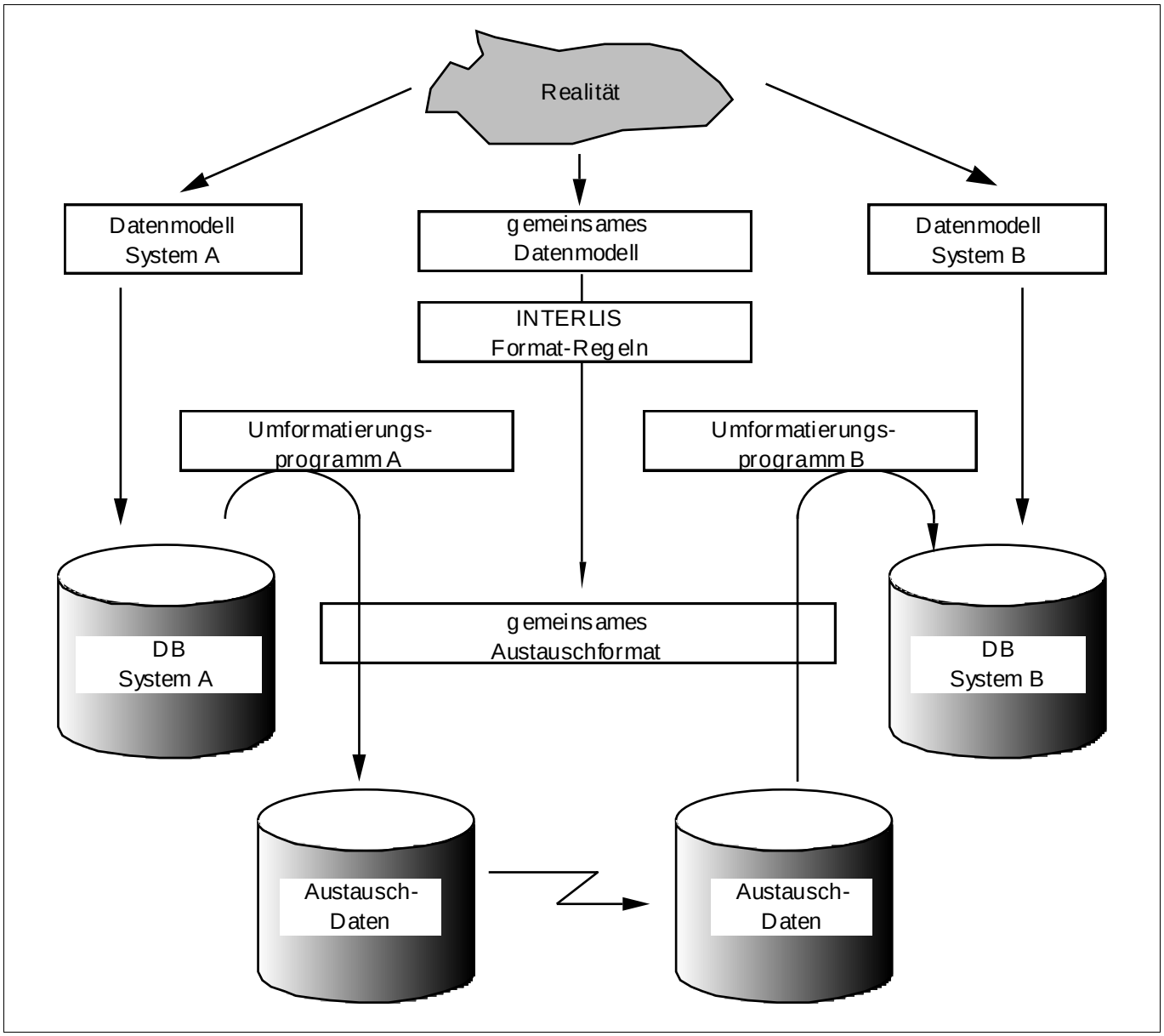

Abbildung 1: Datenaustausch über gemeinsames Datenmodell und gemeinsame Datenbeschreibungssprache zwischen verschiedenen Datenbanken

Bei der Beschreibung des Datenmodells wird nicht von einer bestimmten Anwendung ausgegangen. Vielmehr kann das Modell einer konkreten Anwendung mittels einer Beschreibungssprache definiert werden (vgl. Kapitel 2). Damit steht INTERLIS für die ganze Menge der Anwendungen zur Verfügung, die mit den vorgesehenen Grundelementen beschrieben werden können.

Vom Beschreibungsprinzip her lehnt sich INTERLIS an das relationale Datenmodell an, erweitert dieses aber um Elemente, die typisch für Landinformationssysteme sind. Die Syntax der Sprache folgt den Ideen moderner Programmiersprachen wie PASCAL, MODULA2.

Um einen konkreten Datenaustausch zu ermöglichen, muss aber über das Datenmodell hinaus noch ein Transferprotokoll festgelegt werden. Damit die Flexibilität der Modellbeschreibung nicht verloren geht, wird das Transferprotokoll so formuliert, dass es sich der konkreten Datendefinition anpasst. Zur Zeit ist der Datenaustausch auf Stufe Text-Datei definiert (vgl. Kapitel 3).

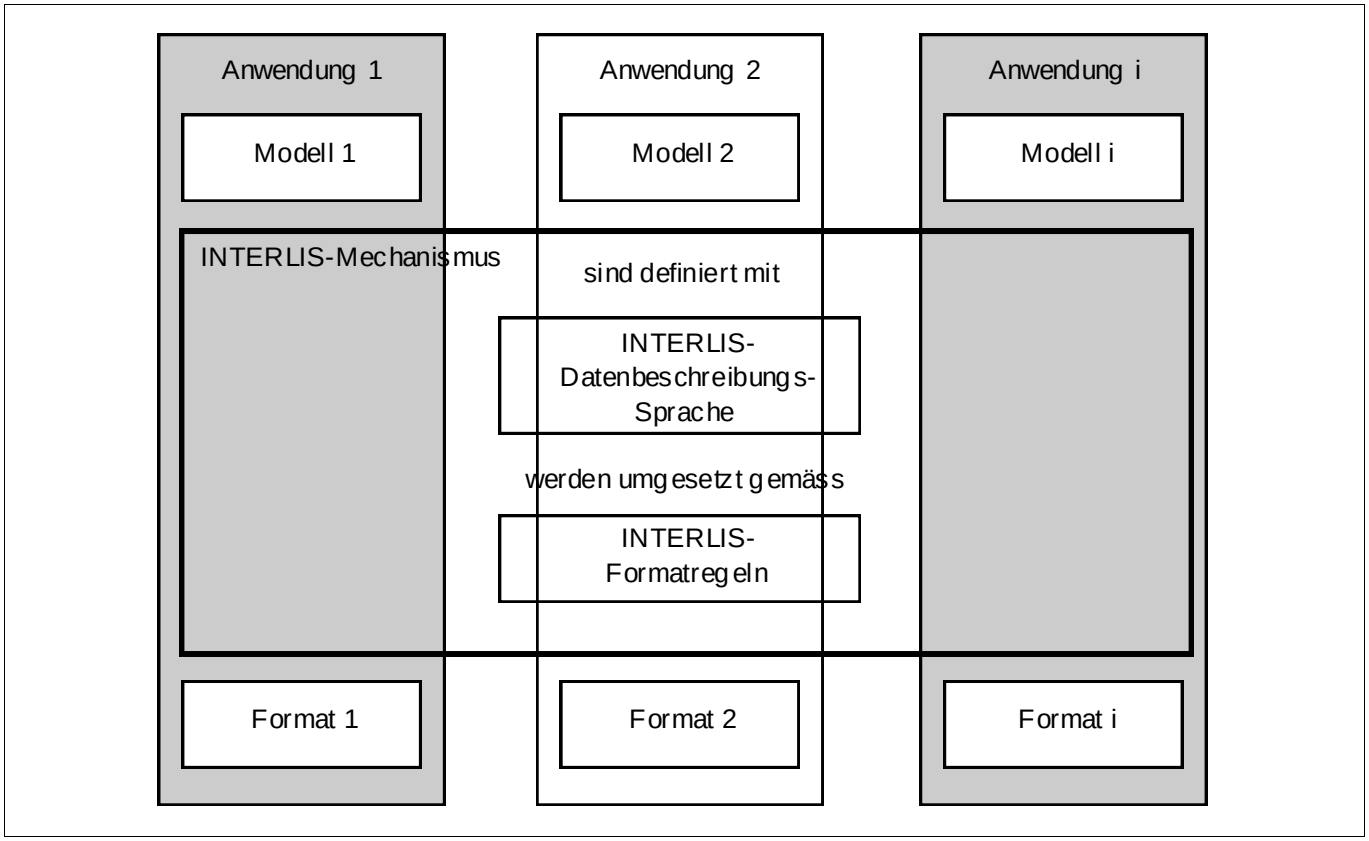

Abbildung 2: INTERLIS-Datenbeschreibungssprache und INTERLIS-Formatregeln

Um kontrollieren zu können, ob ein mit INTERLIS beschriebenes Modell formal korrekt ist, steht ein Compiler zur Verfügung. Er erstellt zudem eine Beschreibung des dem Modell entsprechenden Transferformates (vgl. Kapitel 4).

Kapitel 5 enthält ein einfaches Beispiel. Im Kapitel 6 sind einige Hinweise zu speziellen Fragen zusammengestellt.

2. Beschreibungssprache

2.1. Verwendete Syntax

Zur Festlegung des Datenmodells und der Transferparameter eines Datentransfers wird in den folgenden Kapiteln eine formale Sprache definiert. Diese Sprache ist selbst formal definiert. Syntax-Regeln beschreiben dabei die zulässige Reihenfolge von Symbolen.

Die Beschreibung folgt damit der Art und Weise, die bei der Beschreibung moderner Programmiersprachen üblich ist. Hier wird deshalb nur eine knappe, für das Verständnis nötige Darstellung angegeben. Für Einzelheiten wird auf die Literatur verwiesen. Eine kurze Einführung befindet sich z.B. in "Programmieren in Modula-2" von Niklaus Wirth.

Eine Formel wird im Sinne der erweiterten Backus-Naur-Notation (EBNF) wie folgt aufgebaut:

```
Formel-Name = Formel-Ausdruck.
```

Der Formel-Ausdruck ist eine Kombination von:

- fixen Wörtern (inkl. Spezialzeichen) der Sprache. Sie werden in Apostrophe eingeschlossen, z.B. 'BEGIN'.
- Referenzen von anderen Formeln durch Angabe des Formelnamens.

Als Kombination kommen in Frage:

Aneinanderreihung

```
a b c           zuerst a, dann b, dann c.
```

Obligatorische Auswahl

```
(a|b|c)         a, b, oder c.
```

Fakultative Auswahl

```
[a|b|c]         a, b oder c oder nichts.
```

Fakultative Wiederholung

```
{a|b|c}         Beliebige Folgen von a, b oder c.
```

Beispiele: aaaaaa, abbc, accccab. Es ist auch zulässig, wenn keine Folge von a, b oder c angegeben ist.

Obligatorische Wiederholung (als Zusatz zur EBNF)

```
(*a|b|c*)
```

Beliebige Folge von a, b oder c.

Mindestens einmaliges Vorkommen ist verlangt.

Es ist also unzulässig, wenn nichts angegeben ist. (*a*) ist also gleichbedeutend mit a {a}, erleichtert aber die Schreibweise.

Häufig möchte man eine syntaktisch gleiche Formel in verschiedenen Zusammenhängen, für verschiedene Zwecke verwenden. Um diesen Zusammenhang auch ausdrücken zu können, müsste man eine zusätzliche Formel schreiben:

```
Beispiel = 'TABLE' Tabellenname '=' Tabellendef.
Tabellenname = Name.
```

Damit diese Indirektion vermieden werden kann, wird folgende abgekürzte Schreibweise angewendet:

```
Beispiel = 'TABLE' Tabellen-Name '=' Tabellendef.
```

Die Formel Tabellen-Name wird nicht definiert. Syntaktisch kommt direkt die Regel Name zur Anwendung. Von der Bedeutung her ist der Name jedoch ein Tabellenname. "Tabellen" wird damit quasi zu einem Kommentar.

2.2. Definition der Beschreibungssprache

2.2.1. Grundsymbole der Sprache

Die Beschreibungssprache weist die folgenden Symbol-Klassen auf: Namen, Zahlen, Erläuterungen, Sonderzeichen und reservierte Worte, Kommentare.

2.2.1.1. Name

```
Name = Buchstabe {Buchstabe | Ziffer | '_'}.
Buchstabe = ('A'| .. | 'Z' | 'a' | .. | 'z').
Ziffer = ('0' | '1' | .. | '9').
```

Ein Name ist definiert als eine Folge von Buchstaben, Ziffern und Unterstrichen, wobei das erste Zeichen ein Buchstabe sein muss. Umlaute und Akzente sind in den Namen nicht erlaubt!

2.2.1.2. Zahlen

Zahlen kommen im Zusammenhang mit der Definition von Wertebereichen und bei der Festlegung des Codes eines bestimmten Zeichens vor. Im zweiten Fall ist es angenehm, wenn dafür auch die hexadezimale Darstellung zur Verfügung steht.

```
PosZahl = (* Ziffer *).
Zahl = ['+'|'-']PosZahl.
Dez = Zahl['.'PosZahl] [Skalierung].
Skalierung = 'S'Zahl.
Code = (PosZahl | Hexzahl).
Hexziffer = (Ziffer |'A'|'B'|'C'|'D'|'E'|'F'|'a'|'b'|'c'|'d'|'e'|'f').
Hexzahl = '0' ('x'| 'X') (* Hexziffer *).
```

Beispiele:

```
PosZahl: 5134523         1          23
Zahl:        123      -435       +5769
Dez:         123.456   123.456S4   123.456S-2
Code:       1234      0XA2
```

Mit der Skalierung wird bei Dez erreicht, dass die Wertangabe in einem sinnigen Bereich, ohne überflüssige Nullen erfolgen kann. 123.456S4 bedeutet z.B. 123.456 *10**4 also 1234560.

2.2.1.3. Erläuterungen

Erläuterungen werden dort verlangt, wo ein Sachverhalt näher beschrieben werden muss. Aus der Sicht des Standard-Mechanismus wird die Erläuterung nicht weiter interpretiert, also als Kommentar aufgefasst. Es ist aber durchaus zulässig, Erläuterungen detaillierter zu formalisieren

und damit einer weitergehenden maschinellen Verarbeitung zugänglich zu machen. Als Anfang und Abschluss einer Erläuterung wurde deshalb // gewählt. Innerhalb der Erläuterung dürfen zwei aufeinanderfolgende Schrägstriche nie vorkommen.

```
Erläuterung = '//' beliebig_ausser_// '//'.
```

2.2.1.4. Sonderzeichen und reservierte Worte

Sonderzeichen und reservierte Worte sind in den Syntax-Regeln der Sprache (nicht aber bei einer Datenbeschreibung) immer zwischen Apostrophen geschrieben, z.B. ',' oder 'MODEL'. Die reservierten Worte werden grundsätzlich mit Grossbuchstaben geschrieben. Konflikte zwischen Namen und reservierten Worten können vermieden werden, wenn in Namen nicht ausschliesslich Grossbuchstaben verwendet werden.

Es werden folgende reservierte Worte verwendet:

ANY	ARCS	AREA	BASE	BLANK
CODE	CONTINUE	CONTOUR	COORD2	COORD3
DATE	DEFAULT	DEGREES	DERIVATIVES	DIM1
DIM2	DOMAIN	END	FIX	FONT
FORMAT	FREE	GRADS	HALIGNMENT	I16
I32	IDENT	LINEATTR	LINESIZE	MODEL
NO	OPTIONAL	OVERLAPS	PERIPHERY	POLYLINE
RADIANS	STRAIGHTS	SURFACE	TABLE	TEXT
TID	TIDSIZE	TOPIC	TRANSFER	UNDEFINED
VALIGNMENT	VERTEX	VERTEXINFO	VIEW	WITH
WITHOUT				

Tabelle 1: Reservierte Worte

2.2.1.5. Kommentar

```
!! bis Zeilenende
```

Zwei Ausrufezeichen, die sich unmittelbar folgen, eröffnen einen Kommentar. Der Kommentar wird durch das Zeilenende abgeschlossen.

2.2.1.6. Abtrennung der einzelnen Symbole

Überlicherweise trennt man aufeinanderfolgende Symbole durch einen Zwischenraum (ein oder mehrere Leerzeichen, Tabulatoren oder Zeilenende). Dies ist jedoch nur dann zwingend nötig, wenn durch das Fehlen eines Zwischenraums aus zwei Symbolen ein einziges entstünde.

2.2.2. Ein kleines Beispiel als Einstieg

Als Beispiel wird eine einfache Beschreibung der Art der Erdoberfläche (die amtliche Vermessung der Schweiz ‚AV93' braucht dazu die Bezeichnung Bodenbedeckung) gewählt. Im Kapitel 5 wird das selbe Beispiel näher erläutert. Zudem ist dort das entsprechende Datenaustauschformat und ein konkreter Datensatz aufgeführt.

```
TRANSFER Beispiel;

MODEL Beispiel

  DOMAIN
    LKoord = COORD2 480000.00   60000.00
                    850000.00  320000.00;
```

```
  TOPIC Bodenbedeckung =

    TABLE BoFlaechen =
      Art : (Gebaeude, befestigt, humusiert, Gewaesser,
             bestockt, vegetationslos);
      Form : AREA WITH (STRAIGHTS, ARCS) VERTEX LKoord
             WITHOUT OVERLAPS > 0.10;
    NO IDENT
      !! Suche ueber Form oder Gebaeude
    END BoFlaechen;

    TABLE Gebaeude =
      AssNr : TEXT*6;
      Flaeche : -> BoFlaechen // Art = Gebaeude //;
    IDENT
      AssNr;   !! Annahme AssNr sei eindeutig
      Flaeche; !! Dem Gebaeude ist genau eine Flaeche zugeordnet
    END Gebaeude;

  END Bodenbedeckung.
END Beispiel.

FORMAT FREE;

CODE
  BLANK = DEFAULT, UNDEFINED = DEFAULT, CONTINUE = DEFAULT;
  TID = ANY;
END.
```

2.2.3. Hauptstruktur der Sprache

2.2.3.1. Die Hauptteile

```
TransferDef = 'TRANSFER' Transfer-Name';'
              [Globale-Wertebereichdef]
              DatenModell [Auswertungen]
              {Sicht} Format Kodierung.
```

Eine Transfer-Definition besteht aus der Definition von Wertebereichen, die für die ganze Definition gültig sind (fakultativ), der Beschreibung des Daten-Modells und der Angabe von Transferparametern (Format, Kodierung). Mit dem Daten-Modell wird die Art und Struktur der auszutauschenden Daten präzis beschrieben. Mit den Transferparametern werden noch zusätzliche Angaben für den Transfer beigefügt.

2.2.3.2. Wertebereichdefinition

```
Wertebereichdef = 'DOMAIN'
                    (* Wertebereich-Name '=' Attributtyp ';' *).
```

Mit einer Wertebereichdefinition wird ein bestimmter Datentyp mit allen seinen Eigenschaften festgelegt. Er kann später für die Definition der Attribut-Eigenschaften verwendet werden. Die Wertebereichdefinitionen sind für die Attributdefinitionen wie folgt verfügbar:

- in der ganzen Definition, wenn die Wertebereichdefinition im Rahmen der Transfer-Def erfolgt. Der Name bleibt in der ganzen Definition reserviert.
- innerhalb des Modells oder einer Auswertung, wenn die Wertebereichdefinition im Rahmen der Modelldefinition erfolgt. Der Name ist dann in diesem Rahmen reserviert.
- innerhalb eines Themas, wenn die Wertebereichdefinition im Rahmen der Themadefinition erfolgt. Der Name ist dann nur bis zum Abschluss des Themas reserviert.

2.2.3.3. Das Datenmodell

```
DatenModell = 'MODEL' Modell-Name
              [ Modell-Wertebereichdef ]
              (* Thema *)
              'END' Modell-Name '.'.
```

Das Datenmodell wird beschrieben durch

- einen Modellnamen
- Wertebereichdefinitionen, die im ganzen Modell Gültigkeit haben
- den Definitionen der verschiedenen Themen.

2.2.3.4. Das Thema

```
Thema = 'TOPIC' Thema-Name '='
          (* Tabelle | Lokale-Wertebereichdef *)
        'END' Thema-Name '.'.
```

Ein Thema ist primär eine Sammlung von Tabellen. Vom Standpunkt des Datentransfers aus gesehen, sind die Themen vollständig unabhängig voneinander. Ein System, das Daten entgegennimmt, muss in der Lage sein, die Daten auch zu übernehmen, wenn nicht alle Themen geliefert werden. Bei der Übernahme eines Objektes eines bestimmten Themas dürfen keine Voraussetzungen über das Vorhandensein von Objekten anderer Themen gemacht werden.

Durch diese Entkopplung der Daten wird erreicht, dass die Übertragung eines Objektes nicht in einem Kaskadeneffekt den Transfer einer ganzen Reihe weiterer Objekte nach sich zieht. Übertragungen von partiellen Datenbeständen sind damit möglich. Damit ist keine Aussage darüber gemacht, welche Massnahmen auf dem empfangenden System ergriffen werden müssen, um allfällige Inkonsistenzen zwischen den Themen im Griff zu behalten und zu bereinigen.

2.2.3.5. Die Tabelle

```
Tabelle = ['OPTIONAL'] 'TABLE' Tabellen-Name '='
                          Attribute Identifikationen
                        'END' Tabellen-Name ';'.
Identifikationen = ('NO' 'IDENT' | 'IDENT' (* Identdef *) ).
Identdef = Attribut-Name { ',' Attribut-Name } ';'.
```

Eine Tabelle ist eine Sammlung von Datenobjekten mit gleichen Eigenschaften. Diese werden mittels der Attribute und der Identifikation beschrieben. Die Attribute beschreiben den genauen Aufbau der Daten der Objekte. Mit jeder Definition einer Identifikation wird ein Attribut oder eine Attributkombination definiert, die ein Objekt eindeutig identifiziert.

Beispiel:

```
TABLE Objekt =
  a: TEXT*8;
  b: TEXT*8;
  c: TEXT*8;
IDENT
  a;
  b,c;
END Objekt;
```

Mit der ersten Zeile wird ausgesagt, dass das Attribut a das Objekt identifiziert, mit der zweiten, dass die Kombination von b und c identifizierend ist (b oder c allein müssen also nicht eindeutig sein).

Mit 'OPTIONAL' wird angezeigt, dass die Tabelle nicht zwingend vorhanden sein muss.

2.2.3.6. Das Attribut

```
Attribute = (* Attribut-Name ':' ['OPTIONAL']
               (LokalAttribut | BezAttribut)
               [Konsistenzbedingung] ';' *).
Konsistenzbedingung = Erläuterung.
```

Jedes Attribut wird durch seinen Namen, seine Struktur und allenfalls Angaben über spezielle Konsistenzbedingungen beschrieben. Zudem kann mit 'OPTIONAL' angegeben werden, dass es zulässig ist, das Attribut nicht zu definieren. Fakultativ kann eine Attributdefinition durch Konsistenzangaben abgeschlossen werden. Diese sind jedoch nicht formalisiert, sondern als Erläuterungen angegeben. Von der Struktur her können die Attribute zunächst in zwei Gruppen gegliedert werden.

Lokale Attribute

Ein lokales Attribut hat nur innerhalb der Tabelle eine Bedeutung. Zur Beschreibung der Eigenschaften eines Attributes stehen verschiedene Attributtypen zur Verfügung. Sie können grob in Basistypen und verschiedene Typen für Geometrien gegliedert werden. Die genauen Spezifikationen sind in den folgenden Kapiteln angegeben.

```
LokalAttribut = Wertebereich.
Wertebereich = (Wertebereich-Name | Attributtyp).
Attributtyp = (Basistyp | Linientyp | Flächentyp).
```

Beziehungs-Attribute

Mit dem Mittel der Beziehungs-Attribute werden Beziehungen zwischen Objekten geschaffen. Man muss sich jedoch nicht darum kümmern, wie die Beziehung geschaffen wird. Es wird nur angegeben, zu welcher Tabelle und damit zu welchen Objekten die Beziehung besteht. Damit aber die Unabhängigkeit der Themen gewahrt bleibt, sind Beziehungs-Attribute nur für Beziehungen innerhalb des gleichen Themas zulässig.

```
BezAttribut = '->' Tabellen-Name.
```

2.2.4. Basistypen

```
Basistyp = ( Koord2
```

```
| Koord3
| Länge
| Flächenmass
| Winkel
| Bereich
| Text
| Datum
| Aufzählung
| HorizAlignment
| VertAlignment).
```

Koordinaten, Längen und Flächenmasse bauen alle auf dem Längenmass auf. Damit über die Bedeutung der transferierten Daten Klarheit herrscht, soll für diese Typen eine einheitliche Grundeinheit (in der Regel der Meter) festgelegt werden. Alle Angaben über Längen, Flächen und Koordinaten sollen dann in dieser Einheit erfolgen.

Die Angabe von Minimum und Maximum bei den verschiedenen Typen definiert nebst den Grenzen auch die Genauigkeit. Dezimalstellen und Skalierung müssen bei Minimum und Maximum übereinstimmen. Will man z.B. einen Längentyp mit einer Obergrenze von einem Kilometer definieren und den Millimeter dabei darstellen können, schreibt man:

```
DIM1 0.000 1000.000
```

2.2.4.1. Koordinaten

```
Koord2 = 'COORD2' Emin-Dez Nmin-Dez Emax-Dez Nmax-Dez.
Koord3 = 'COORD3' Emin-Dez Nmin-Dez Hmin-Dez Emax-Dez Nmax-Dez Hmax-Dez.
```

Bei Koordinaten wird für jede Komponente der zulässige Wertebereich festgelegt. Ohne weitere Angaben (als Kommentar) ist die erste Komponente der Ostwert (Easting), die zweite Komponente der Nordwert (Northing), und die dritte Komponente der Höhenwert (Height).

2.2.4.2. Länge und Flächenmass

```
Länge = 'DIM1' min-Dez max-Dez.
Flächenmass = 'DIM2' min-Dez max-Dez.
```

2.2.4.3. Winkel

```
Winkel = ('RADIANS' | 'GRADS' | 'DEGREES') min-Dez max-Dez.
```

Ohne weiteren Angaben (als Kommentar) wird angenommen, dass GRADS und DEGREES im vermessungstechnischen Koordinatensystem, RADIANS im mathematischen System definiert sind.

2.2.4.4. Bereich

```
Bereich = '[' min-Dez '..'max-Dez']'.
```

Es wird ein Zahlenbereich definiert, ohne dass die genaue Bedeutung durch den Typ festgelegt wird. Sie soll sich entweder aus dem jeweiligen Attribut-Namen oder einem zusätzlichen Kommentar ergeben.

2.2.4.5. Text

```
Text = 'TEXT' '*' MaxLänge-PosZahl.
```

Unter einem Text wird eine beliebige Folge von Zeichen verstanden, die die angegebene maximale Zeichenzahl nicht überschreiten darf. Um Probleme bei der Datenübertragung zu vermeiden, müssen allerdings die Regeln und Empfehlungen gemäss Kap. 3.4.5.5. eingehalten werden.

2.2.4.6. Datum

```
Datum = 'DATE'.
```

Damit eine einheitliche Behandlung des Datums möglich ist, wird das Datum nicht einfach als Text eingeführt. Das Datum identifiziert den Tag. Es besteht also aus Jahr, Monat und Tag.

2.2.4.7. Aufzählung

```
Aufzählung = '(' Element {','Element} ')'.
Element = Element-Name [Unter-Aufzählung].
```

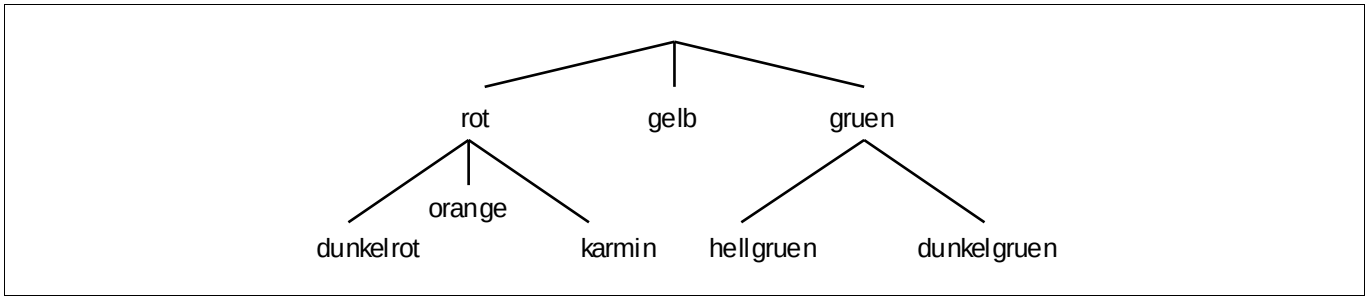

Abbildung 3: Beispiel einer Aufzählung

Mit einer Aufzählung werden die für diesen Typ zulässigen Werte festgelegt. Die Aufzählung ist jedoch nicht einfach linear, sondern im Sinne eines Baumes strukturiert. Beispiel:

```
(rot (dunkelrot, karmin, orange), gelb, gruen (hellgruen, dunkelgruen))
```

ergibt folgende zulässigen Werte:

```
rot-dunkelrot rot-karmin rot-orange gelb gruen-hellgruen gruen-dunkelgruen
```

Eine Schachtelung wird in runden Klammern angegeben. Die Schachtelungstiefe ist frei wählbar.

2.2.4.8. Textausrichtung

```
HorizAlignment = 'HALIGNMENT'.
VertAlignment = 'VALIGNMENT'.
```

Für die Aufbereitung von Plänen müssen die Positionen von Texten festgehalten werden. Dabei muss festgelegt werden, welcher Stelle des Textes die Position entspricht. Mit dem horizontalen Alignment wird festgelegt, ob die Position auf dem linken oder rechten Rand des Textes oder in der Textmitte liegt. Das vertikale Alignment legt die Position in Richtung der Texthöhe fest.

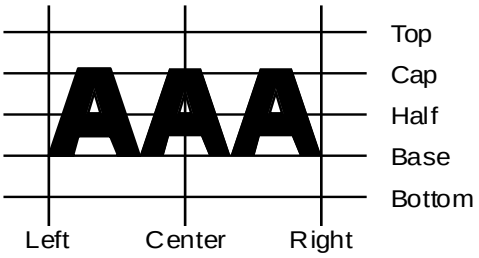

Abbildung 4: Textausrichtung horizontal und vertikal

Der Abstand Cap-Base entspricht der Höhe der Grossbuchstaben. Unterlängen befinden sich im Bereich von Base-Bottom.

Horizontales und vertikales Alignment können als vordefinierte Aufzählung verstanden werden. Die Wirkung ist, wie wenn man in der globalen Wertebereichsdefinition folgendes definiert hätte:

```
DOMAIN
  HALIGNMENT = (Left, Center, Right);
  VALIGNMENT = (Top, Cap, Half, Base, Bottom);
```

2.2.5. Linientyp

Unter dem Begriff Linie wird eine ununterbrochene Folge von Linienstücken, ein einziger Linienzug verstanden. Nähere Beschreibungen werden für die Form des Linienzuges (welche Verbindungsgeometrien sind zulässig; dürfen sich Linien schneiden?) und Angaben über die Eigenschaften der Stützpunkte verlangt.

```
Linientyp = 'POLYLINE' Form Stützpunkte [Schnittdef].
Form = 'WITH' '('Formtyp { ',' Formtyp} ')'.
Formtyp = ('STRAIGHTS' | 'ARCS' | Erläuterung).
```

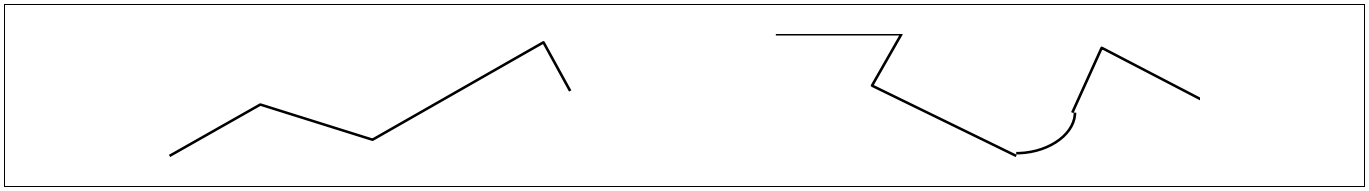

Abbildung 5: Verschiedene Linienzügen

Es können also Kombinationen von Geraden, Kreisbogen und speziellen Formen als zulässige Verbindungsgeometrien erklärt werden.

Die Kreisbogen werden auf dem Transferfile mittels einem Zwischenpunkt in Bogenmitte dargestellt. Bogen mit sehr grossem Zentriwinkel (nahezu Vollkreise) werden dabei numerisch instabil. Sie sind also zu vermeiden. Der Zwischenpunkt in Bogenmitte hat nicht die gleiche Bedeutung wie ein Stützpunkt. Er dient nur der Definition der Form des Bogens. Für den gleichen Bogen können bei verschiedenen Transfers durchaus verschiedene Zwischenpunkte verwendet werden. Sie sollen aber immer in der Nähe der Bogenmitte liegen. Damit ist gewährleistet, dass die am Transfer beteiligten Systeme in der internen Bogenbeschreibung frei sind.

Kompliziertere Verbindungsgeometrien werden durch INTERLIS nicht fixiert. Damit aber dennoch die Möglichkeit zur Beschreibung und Übertragung besteht, können Spezialfälle als Erläuterung

angegeben werden (z.B. für Interpolationskurven). Die Art der Beschreibung und der Codierung auf dem Transferfile ist dabei Sache der am Transfer Beteiligten.

```
Schnittdef = 'WITHOUT' 'OVERLAPS' '>' Dez .
```

Es kann verlangt werden, dass sich eine Linie weder mit sich selbst noch mit einer Linie des gleichen Attributes eines anderen Objektes schneidet. Überschneidungen, die innerhalb der gegebenen Toleranz liegen, sind im Zusammenhang mit Kreisbogen dann erlaubt, wenn sich der Kreisbogen ausgehend von einem Linienstützpunkt zunächst auf der einen Seite der anderen Linie befindet und diese dann schneidet, ohne dass zwischen dem gemeinsamen Stützpunkt und dem Schnittpunkt auf der einen oder anderen Linie weitere Stützpunkte liegen. Diese Regelung erfolgt aus zwei Gründen: Einserseits sind bei Kreisbogen kleine Überschneidungen aus numerischen Gründen in gewissen Fällen (tangentiale Kreisbogen) nicht vermeidbar. Andererseits sind bei der Übernahme von Daten, die ursprünglich grafisch erfasst wurden, auch grössere Überlappungen (z.B. einige Zentimeter) zu tolerieren, will man nicht einen enormen Nachbearbeitungsaufwand in Kauf nehmen. Die Angabe der Toleranz muss in den gleichen Einheiten, wie die der Stützpunktkoordinaten erfolgen.

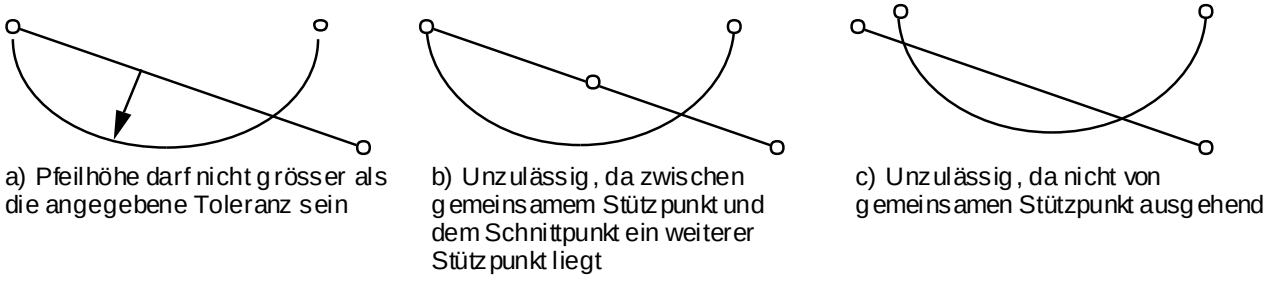

Abbildung 6: Toleranz (a) und unzulässige Überschneidungen (b+c) von Linienzügen

```
Stützpunkte = 'VERTEX' (Koord2 | Koord3 | Koordtyp-Name) ['BASE' Erläuterung].
```

Als Stützpunkte der Linie werden diejenigen Punkte bezeichnet, die eine spezielle Bedeutung haben (vor allem Anfangs- und Endpunkte sowie Knickpunkte).

Primär wird der Wertebereich der Koordinaten definiert. Mittels einer Erläuterung besteht die Möglichkeit, zusätzliche Bedingungen zu beschreiben. So kann z.B. gefordert werden, dass die Koordinaten nicht beliebig sein dürfen, sondern denjenigen der Punkte einer bestimmten Tabelle entsprechen müssen.

Die Zwischenpunkte von Kreisbogen gelten nicht als Stützpunkte. Ihre Koordinaten haben aber den gleichen Wertebereich wie die Stützpunkte.

2.2.6. Flächentypen

Fläche heisst ein durch Linienzüge berandeter massiv zusammenhängender ebener Bereich. Eine Fläche hat immer einen zusammenhängenden äusseren Rand. Gehört nicht der ganze Bereich innerhalb dieses äusseren Randes zur Fläche, dann gibt es einen oder mehrere weitere zusammenhängende Ränder, sogenannte innere Ränder der Fläche. Die dadurch ausgesparten Bereiche heissen Enklaven.

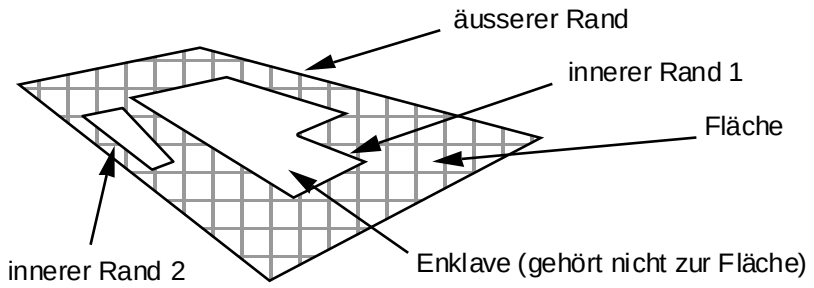

Abbildung 7: Fläche mit Rand und Enklaven

Teilflächen gehören nur dann zur gleichen Fläche, wenn sie massiv zusammenhängen. Stossen sie nur in einem Punkt aneinander, bilden sie zwei Flächen.

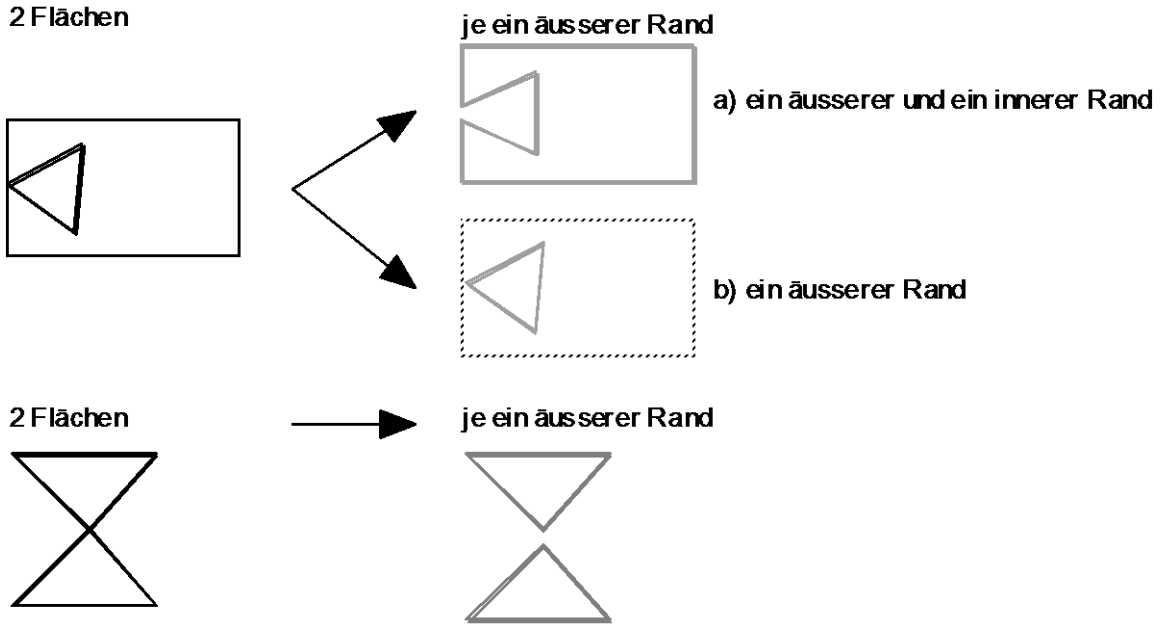

Abbildung 8: Beispiele von Teilflächen. INTERLIS erlaubt Variante a und b.

Mit INTERLIS kann beschrieben werden, ob sich die Flächen verschiedener Objekte der gleichen Tabelle überlappen dürfen oder nicht.

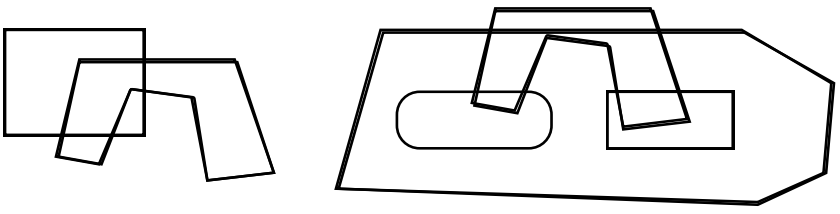

Abbildung 9: Einzelflächen (SURFACE)

Einzelfläche

Für Flächen, die sich ganz oder teilweise überlappen dürfen, d.h. die nicht nur Randpunkte gemeinsam haben dürfen, steht der geometrische Attributtyp SURFACE zur Verfügung (siehe Abbildung 9 mit Beispielen).

Gebietseinteilung

Gebietseinteilung heisst eine Sammlung von Flächen, welche die Ebene lückenlos und überlappungsfrei überdecken.

Die Gebietseinteilung besteht aus einer Restfläche mit beliebig vielen Enklaven. Alle anderen Flächen überdecken die Enklaven der Restfläche vollständig. Für Gebietseinteilungen steht der geometrische Attributtyp AREA zur Verfügung.

Einem Gebietsobjekt ist genau eine Fläche der Gebietseinteilung zugeordnet. Einer Fläche der Gebietseinteilung ist höchstens ein Gebietsobjekt zugeordnet. Es ist nicht zulässig, dass zwei Flächen der Gebietseinteilung mit einem gemeinsamen Rand je kein Gebietsobjekt entspricht.

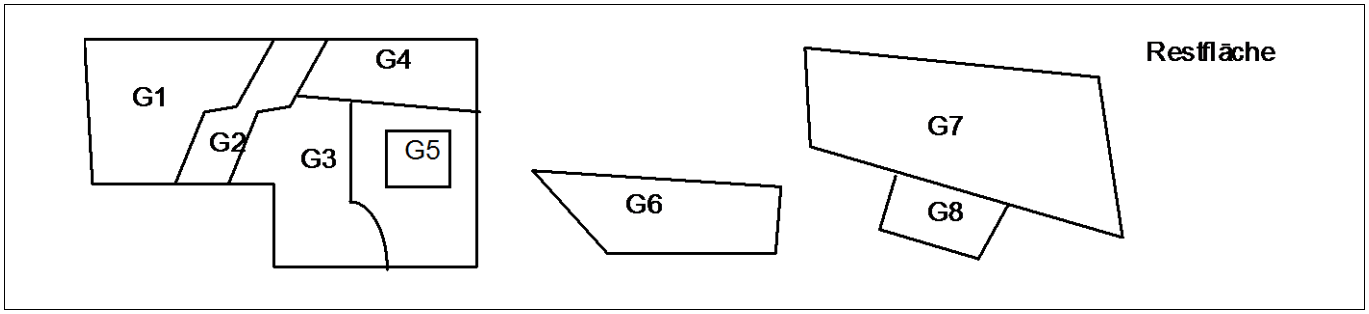

Abbildung 10: Gebietseinteilung (AREA)

Syntax

```
Flächentyp = ('SURFACE' Form Stützpunkte [Schnittdef]
             |'AREA' Form Stützpunkte Schnittdef)
             [Linattrdef].
Linattrdef = 'LINEATTR' '='
               Attribute [Identifikationen]
             'END'.
```

Beim geometrischen Attributstyp AREA darf keine OPTIONAL-Angabe gemacht werden.

Tabellenstruktur

Im Unterschied zu allen andern Attributtypen wird bei Einzelflächen (SURFACE) und Gebietseinteilungen (AREA) implizit eine weitere Tabelle, die die Linien enthält, gebildet. Als Tabellenname wird <Haupt-Tab-Name>_<Attribut-Name> genommen.

Die Aufspaltung der Ränder in einzelne Linien und der Richtungssinn der Linien ist beliebig und darf bei verschiedenen Datenübertragungen unterschiedlich sein.

Für Linienzüge können bei Bedarf auch Attribute festgelegt werden (Linattrdef). Es dürfen dabei aber keine Flächentypen vorkommen.

Linien von Gebietseinteilungen

Linienobjekte einer Gebietseinteilung müssen immer echte Grenzlinien sein. Es dürfen also keinen Linien existieren, bei denen auf beiden Seiten die gleiche Fläche liegt (Abbildung 11).

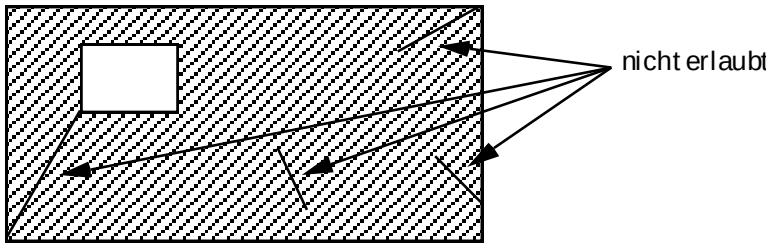

Abbildung 11: Nicht erlaubte Flächenkonfigurationen

Auf der ganzen Länge eines Linienobjekts dürfen die Flächen auf den beiden Seiten der Linie nicht ändern. Linien dürfen also nicht über echte Knoten (d.h. mehr als zwei zusammentreffende Linien) hinausgehen.

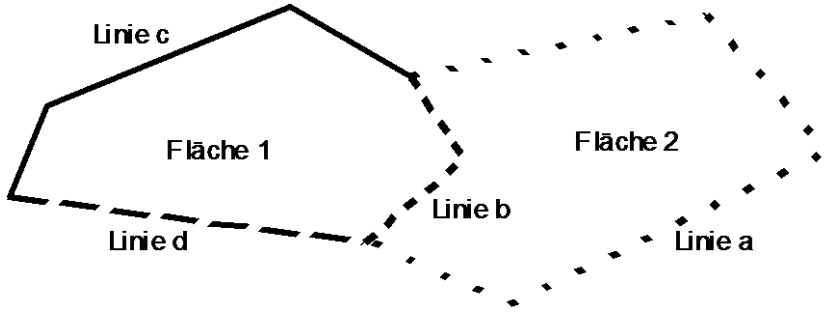

Abbildung 12: Linienobjekte a, b, c, d mit Flächen 1 und 2 der Gebietseinteilung

In Abbildung 12 enthält die Haupttabelle die Objekte Fläche 1 und Fläche 2 der Gebietseinteilung, die Linientabelle die Objekte Linie a, Linie b, Linie c und Linie d.

Die Zuordnung zwischen Gebietsobjekten und den durch die Linienobjekte gebildeten Flächen wird beschrieben, indem bei jedem Gebietsobjekt für das Gebietsattribut ein Gebietsreferenzpunkt ausgegeben wird. Dieser Gebietsreferenzpunkt darf dabei nicht in Überlappungsbereichen von Linien liegen. Problemlos ist der Bereich innerhalb der gestrichelten Linie (vgl. Abbildung 13).

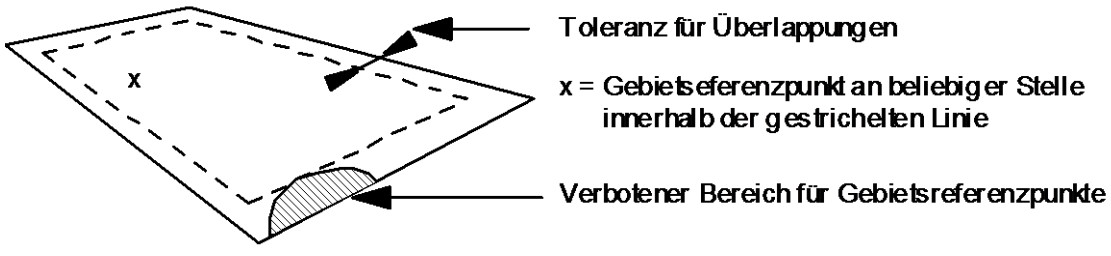

Abbildung 13: Erlaubter Bereich von Gebietsreferenzpunkten

Die Lage des Gebietsreferenzpunkts darf bei verschiedenen Datenübertragungen unterschiedlich sein.

2.2.7. Auswertungen

```
Auswertungen = 'DERIVATIVES' Auswertung-Name
                 [Globale-Wertebereichdef] (* Thema *).
               'END' Auswertung-Name '.'.
```

Auswertungen sind von der Struktur gleich aufgebaut wie das Modell. Sie definieren jedoch keine neuen Daten. Die einzelnen Attributwerte sind Auswertungen, Funktionen der Modelldaten.

2.2.8. Sichten

Mit den Sichten wird beschrieben, wie die Daten des Modells und der Auswertungen auf dem Transferfile angeordnet werden. Es können vier Dinge beeinflusst werden:

- Bei der Ausgabe von Linien kann verlangt werden, dass nebst den Punktkoordinaten noch zusätzliche Informationen ausgegeben werden (VERTEXINFO).
- Zeigt eine Tabelle auf eine andere (Beziehungsattribut), kann verlangt werden, dass sie im Rahmen der bezeigten Tabelle ausgegeben wird (<-).

Beispiel:

```
MODEL X
  TOPIC Y =
    TABLE A =
      a1: TEXT*10;
      a2: TEXT*5;
    IDENT
      a1;
    END A;

    TABLE B =
      b1: TEXT * 12;
      b2: -> A;
    IDENT
      b1;
    END B;
    :
    :
  VIEW  X
        Y.A <- B.b2;
```

Im Rahmen der Ausgabe der Tabelle A werden im Rahmen eines Objektes a die Daten derjenigen Objekte von B, die auf das aktuelle a verweisen, als Auswertungsattribute ausgegeben (vgl. 3.4.7). Wenn alle Beziehungen einer Tabelle in dieser Art ausgegeben werden, wird die Tabelle selbst nicht ausgegeben.

- Bei Gebietseinteilungen und Einzelflächen kann verlangt werden, dass die Geometrie nicht im Rahmen der Linientabelle sondern direkt als Auswerteattribut (vgl. 3.4.7.) ausgegeben wird (CONTOUR).

- Bei Gebietseinteilungen stellt sich zudem die Frage, ob die Restfläche als Objekt ausgegeben werden soll oder nicht. Mit WITH PERIPHERY dies verlangt.

```
Sicht = 'VIEW' Modell-Name
          {Thema-Name '.' Tabellen-Name ':' Sichtdefinition
          {',' Sichtdefinition} ';'}
        'END' Modell-Name '.'.
Sichtdefinition = ('VERTEXINFO' LinAttr-Name Erläuterung
                    | 'WITH' 'PERIPHERY' GebAttr_Name
                    | 'CONTOUR' GeboderFlAttr-Name ['WITH' 'PERIPHERY']
                    | '<-' Tabellen-Name '.' Attribut-Name).
```

2.2.9. Format

Für den Transfer muss festgelegt werden, ob die Datei nach den Regeln des fixen oder freien Formates aufgebaut ist (vgl. Kap. 3.2.). Beim fixen Format ist die Zeilenlänge und die Länge der Identifikationen festzulegen.

```
Format = 'FORMAT' ( 'FREE'
            | 'FIX' 'WITH' 'LINESIZE' '=' Poszahl',' 'TIDSIZE' '=' Poszahl) ';'.
```

2.2.10. Kodierung

Damit die Zeichen auf dem Zielsystem zweifelsfrei erkannt werden können, ist der verwendete Zeichensatz festzuhalten. Wird keine Font-Definition ausgegeben, gilt der 8-Bit-Zeichensatz des ISO-Standards (ISO 8859-1:1987). Sonst ist der zu verwendende Zeichensatz mittels einer Erläuterung zweifelsfrei anzugeben. Für die zulässigen Zeichen im Rahmen von Textattributten sind die Anforderungen in Kapitel 3.4.5.5 zu beachten.

Für einige Sachverhalte werden auf dem Transferfile spezielle Zeichen verwendet. INTERLIS definiert dafür nur Vorschläge (DEFAULT). Sie können jedoch für besondere Fälle verändert werden. Dann ist die Codezahl des gewünschten Zeichens innerhalb des gewählten Zeichensatzes anzugeben.

Es sind folgende Sachverhalte zu regeln:

Innerhalb von Textattributen können Leerzeichen vorkommen. Damit trotzdem die logische Gliederung in Felder nicht gestört wird (vgl. Kap. 3.1.), wird statt des Leerzeichens ein Ersatzzeichen verwendet. INTERLIS benätzt dafür üblicherweise - DEFAULT - den Unterstrich (_) (ASCII 0x5f).

Bei Attributen, die als optional definiert sind, können die Werte undefiniert sein. Es wird dann statt einem Wert ein spezielles Zeichen gesetzt. INTERLIS verwendet dafür üblicherweise @ (ASCII 0x40).

Es kann vorkommen, dass die Zeilenlänge nicht ausreicht, um alle Felder aufzunehmen. Die logische Zeile wird dann in mehrere physische Zeilen aufgetrennt (vgl. Kap. 3.1.). Damit es klar ist, dass die logische Zeile noch nicht abgeschlossen ist, wird unmittelbar vor dem Zeilenende ein Fortsetzungszeichen eingefügt. INTERLIS verwendet dafür üblicherweise \ (ASCII 0x5c).

Innerhalb des Transfers erhält jedes Objekt eine eindeutige Identifikation. Einerseits können damit Beziehungen auf einfache Art definiert werden, andererseits kann die Identifikation auch bei allfälligen Problemen eine gute Hilfe sein. Der Aufbau der Transfer-Identifikation ist grundsätzlich freigestellt. Sie wird als Text beliebiger Länge aufgefasst. Bei fixem Format muss jedoch eine maximale Länge festgelegt werden (vgl. oben). Um die Herstellung von Beziehungen auf dem

Zielsystem zu erleichtern, ist es nützlich, die Art der Identifikation noch näher zu spezifizieren. Mit I16 bzw. I32 wird ausgesagt, dass die Identifikation eine Integer-Zahl mit 16 bzw. 32 Bit Genauigkeit ist.

```
Kodierung = 'CODE' [Font] Spezzeichen TransferId 'END' '.'.
Font = 'FONT' '=' Erläuterung ';'.
Spezzeichen = 'BLANK' '=' ('DEFAULT' | Code) ','
              'UNDEFINED' '=' ('DEFAULT' | Code) ','
              'CONTINUE' '=' ('DEFAULT' | Code) ';'
TransferId = 'TID' '=' ('I16' | 'I32' | 'ANY' | Erläuterung) ';'.
```

3. Transferfile-Aufbau

3.1. Systemorientierte Strukturierung

Ein Transferfile besteht - unabhängig vom sachlichen Inhalt - aus einer Folge von logischen Zeilen. In den folgenden Beschreibungen wird das Zeilenende jeweils mit NL angezeigt. Der Aufbau einer logischen Zeile kann unter zwei verschiedenen Gesichtspunkten betrachtet werden:

Physische Gliederung

Eine logische Zeile kann beliebig in physische Zeilen aufgeteilt werden. Der Anfang der ersten und das Ende der letzten physischen Zeile einer logischen Zeile braucht keine spezielle Behandlung. Am Ende einer physischen Zeile, die nicht am Ende einer logischen Zeile steht, muss unmittelbar vor dem Zeilenende das Fortsetzungszeichen gesetzt werden. Üblicherweise wird dafür \ verwendet. Mit der Beschreibung der Kodierung (vgl. Kap. 2.2.10) kann ein anderes Zeichen dafür definiert werden. Am Anfang einer Fortsetzungszeile wird als erstes Feld die Zeilenkennzeichnung CONT und ein Leerzeichen eingefügt. Zur logischen Zeile gehören die Zeichen vor dem Fortsetzungszeichen und die Zeichen nach dem eingefügten Leerzeichen der Fortsetzungszeilen, nicht aber das Fortsetzungszeichen, die Zeilenkennzeichnung CONT und das anschliessende Leerzeichen.

Wie ein Transferfile in physische Zeilen gegliedert wird, wird durch INTERLIS nicht geregelt. Es muss auf einem tieferen Schnittstellen-Niveau definiert werden. Ohne besondere Absprachen werden physische Zeilen mit LF (Line Feed, ASCII 0xA) abgeschlossen. Unmittelbar vor LF darf fakultativ ein CR-Zeichen (ASCII 0xD) stehen.

Logische Gliederung

Die logische Zeile besteht aus einer Folge von Feldern. Die Anordnung der Felder auf der Zeile und der Zusammenhang mit der physischen Gliederung ist abhängig von der Art der Formatierung des Datenteils des Transferfiles (vgl. Kap. 2.2.9).

Die nachfolgende Beschreibung der Datenstrukturierung und der Codierung der einzelnen Felder baut nur auf der logischen Gliederung auf. Die Frage wie die einzelnen Felder voneinander getrennt werden, hängt ebenfalls davon ab, ob die Datei gemäss freier oder fixer Formatierung aufgebaut ist (vgl. Kap. 3.2). Beide Fälle haben gemeinsam, dass zwischen zwei Feldern der gleichen logischen Zeile mindestens ein Trennungszeichen (Leerzeichen oder Tabulator) eingefügt wird.

3.2. Freies und fixes Format

Der Datenteil eines Transferfiles kann frei oder fix formatiert werden. Freies Format ist vor allem für modernere Programmiersprachen wie PASCAL, MODULA geeignet, während das fixe Format vor allem bei Sprachen wie BASIC und FORTRAN Anwendung findet.

3.2.1. Freies Format

Im freien Format sind die einzelnen Felder durch mindestens ein Leerzeichen oder Tabulator zu trennen. Die Feldlänge ist jeweils gleich der Länge des aktuellen Wertes. Eine logische Zeile kann an einer beliebigen Stelle, insbesondere auch mitten in einem Feld, aufgetrennt werden (vgl. Kap. 3.1). Die für die Auftrennung nötigen Zeichen müssen also bei der Decodierung wieder eliminiert werden, um den Zeichenfluss der logischen Zeile wieder zu erhalten.

Müssen auf einer logischen Zeile nur noch Undefiniert-Zeichen ausgegeben werden, ist es im freien Format zulässig, an ihrer Stelle zwei unmittelbar aufeinanderfolgende Undefiniert-Zeichen auszugeben und die logische Zeile abzuschliessen.

3.2.2. Fixes Format

Im fixen Format sind die einzelnen Felder durch genau ein Leerzeichen abgetrennt. Das einzelne Feld hat folgenden Aufbau:

Erstes Zeichen

(Nur sofern das entsprechende Attribut fakultativ ist) Zeichen für die Anzeige, ob der Wert definiert oder undefiniert ist. Ist der Wert definiert, enthält die Position ein Leerzeichen. Bei undefiniertem Wert wird das Undefiniert-Zeichen (vgl. Kap. 3.4.4) gesetzt.

Weitere Zeichen

weitere Zeichen für den Wert. Dabei werden für ein bestimmtes Feld immer soviele Zeichen reserviert, wie es für denjenigen Wert nötig ist, der am meisten Platz braucht (Maximale Wertlänge). Die konkreten Definitionen sind im Kapitel 3.4 festgehalten. Numerische Felder werden immer rechtsbündig, Textfelder immer linksbündig angeordnet. Leerstellen am Schluss eines Feldes werden immer als Leerstellen, solche am Anfang oder innerhalb eines Textfeldes mit dem Ersatzzeichen (vgl. Kap. 3.4.5. und 2.2.10) dargestellt.

Haben nicht alle verbleibenden Felder innerhalb der für das fixe Format festgelegten Zeilenlänge (vgl. Kap. 2.2.9) Platz, muss die Zeile unterteilt werden. Es wird so unterteilt, dass ein Feld vollständig auf einer Zeile angeordnet ist. Eine Fortsetzungs-Zeile wird erst begonnen, wenn das aktuelle Feld und - sofern es nicht das letzte Feld der logischen Zeile ist - die Zeichen für Feldtrennung (Leerzeichen) und Fortsetzung auf der aktuellen Zeile keinen Platz mehr finden. Feldtrennung und Fortsetzungszeichen müssen also immer auf der alten Zeile stehen.

Das fixe Format ist demnach so aufgebaut, dass es auch nach den Regeln des freien Formates gelesen werden kann.

Im Hinblick auf das Lesen im fixen Format wurde die Abfolge der Zeilen und Zeilenkennzeichnungen so gewählt, dass die nächste Zeile immer mit einem dafür bestimmten Format gelesen werden kann. Nochmaliges, internes Lesen (Reread) ist also nicht nötig. In vielen Fällen kommen Zeilen mit der gleichen Zeilenkennzeichnung in mehrfacher Folge vor. Den Abschluss der Sequenz bildet eine Zeile, die nur aus einer für den Abschluss typischen Zeilenkennzeichnung besteht. Man liest also stets mit dem gleichen Format und prüft, ob die Zeilenkennzeichnung nicht den Abschluss signalisiert. Im Bereich der Liniengeometrien kommt hinzu, dass zwar das gleiche Format erwartet werden kann, aber von unterschiedlichen Zeilenkennzeichnungen für unterschiedliche Verbindungsgeometrietypen ausgegangen werden muss.

3.3. Sachliche Strukturierung

```
Transferfile = 'SCNT' NL Inhalts-Beschreibung
                ('MOTR' NL Modell und Transferbeschreibung
                |'MTID' Transfer-Name NL)
                Daten 'ENDE' NL.
Beschreibung = {Beschreibungszeile} '////' NL.
Beschreibungszeile = beliebig_ausser_////_am_ Zeilenanfang NL.
Daten = {ModellD}.
ModellD = 'MODL' Modell-Name NL
          {ThemaD} 'EMOD' NL.
```

Ein Transferfile enthält zunächst eine Angabe über den Inhalt, namentlich den geographischen Ausschnitt. Diese Angabe ist als reiner Text ohne Formalisierung definiert. Die am Transfer Beteiligten können jedoch selbst Formalisierungen festlegen. Dann folgt die Transferbeschreibung (vgl. Kap. 2.2) bzw. ein Hinweis, der eine beidseitig bekannte Transferbeschreibung eindeutig identifiziert. Anschliessend folgen die Daten. Diese werden ebenenweise, in der Reihenfolge der Modelldefinition ausgegeben. Es ist jedoch zulässig, dass nicht alle Themen gemäss Modell übermittelt werden. Wird ein Thema jedoch übermittelt, muss es vollständig geschehen, d.h., es muss der Inhalts-Beschreibung genügen.

```
ThemaD = 'TOPI' Themen-Name NL {TabellenD} 'ETOP'.
TabellenD = 'TABL' Tabellen-Name NL
            { ObjektD }
            [Perimeterangabe]
            'ETAB' NL.
ObjektD = 'OBJE' TransferId AttributD.
AttributD = { Basisattribut | GebRef-Koordinate } NL
            { Liniensequenz } { FlächenA | GebietsA } { TabellenA }
Perimeterangabe = 'PERI' TransferId-Text NL.
```

Die Reihenfolge der Tabellen ergibt sich aus der Reihenfolge der Tabellendefinitionen innerhalb des Themas. Pro Tabelle werden alle Objekte der Tabelle ausgegeben. Die Reihenfolge ist dabei beliebig. Im Falle von Gebietseinteilungen wird nach dem letzten Objekt noch angegeben, welches Objekt den Perimeter (PERI) bildet, sofern der Perimeter überhaupt angegeben werden soll (vgl. Kap. 2.2.8).

Die Reihenfolge der Attribute pro Objekt entspricht primär der Reihenfolge der Attribute in der Beschreibung. Für Basisattribute gilt diese Regel ohne Ausnahme.

Bei Gebieteinteilungs- oder Einzelflächenattribute gelten folgende Regeln:

Ausgabe der Geometrie von Einzelflächen im Rahmen der Linientabelle:

Die Linientabelle wird unmittelbar nach der Haupttabelle ausgegeben. Sind mehrere Einzelflächenattribute vorhanden, ergibt sich die Reihenfolge der Linientabellen aus der Reihenfolge der Einzelflächenattribute.

Beim Linienobjekt wird als erstes Attribut ein Beziehungsattribut angegeben, das den Hinweis auf das Flächenobjekt enthält (vgl. Kap. 2.2.6).

Ausgabe der Geometrie von Gebietseinteilungen im Rahmen der Linientabelle:

Die Linientabelle wird unmittelbar vor der Haupttabelle ausgegeben.

Beim Gebietsobjekt werden an der Stelle des Gebietseinteilungsattributes die Koordinaten des Gebietsreferenzpunktes ausgegeben, der im Innern des Gebietsobjekts liegt (vgl. Kap. 2.2.6).

Ausgabe der Geometrie mit dem Gebiets- oder Flächenobjekt (CONTOUR):

Die Reihenfolge ergibt sich aus der Reihenfolge der Attributdefinition. Die Perimeterangabe erfolgt in diesem Fall nur, wenn auch das Objekt ausgegeben, also WITH PERIPHERY verlangt wurde.

Linienattribute und Auswertungsattribute (Einzelflächen-, Gebietseinteilungsattribute, Tabellenauswertungen) werden nicht direkt auf der Objekt-Zeile ausgegeben. Ihre Ausgabe erfolgt nach den Basisattributen und Gebietsreferenzpunkten. Zuerst werden die Linienattribute ausgegeben, dann folgen die Einzelflächen- und Gebietseinteilungsattribute, dann die Tabellenauswertungen je in der Reihenfolge der Definition.

3.4. Definition der Codierung

3.4.1. Zeilenkennzeichnung

Das erste Feld jeder logischen Zeile enthält die Zeilenkennzeichnung. Sie hat immer vier Zeichen.

3.4.2. Themen- und Tabellennamen

Die maximale Wertlänge von Themen- und Tabellennamen ist 24 Zeichen. Weitere Zeichen werden ignoriert.

3.4.3. Transfer-Identifikation

Die Transfer-Identifikation (`TransferId-Text`) belegt ein Feld. Von der Struktur her wird sie grundsätzlich wie Text behandelt. Die maximale Wertlänge wird für fixes Format in den Transfer-Parametern beschrieben (Kap. 2.2.9). Für freies Format muss sie nicht festgelegt werden.

3.4.4. Undefinierte Attribute

Attribute, die in der Modellbeschreibung mit "OPTIONAL" bezeichnet sind, können undefinierte Werte aufweisen. Zur Anzeige dient das Undefiniert-Zeichen. Üblicherweise wird dafür das Zeichen @ verwendet. Im Rahmen der Beschreibung der Kodierung (vgl. Kap. 2.2.10) kann es jedoch umdefiniert werden. Bei freiem Format wird das Undefiniert-Zeichen anstelle des Wertes ausgegeben. Bei fixem Format wird es an der dafür vorgesehenen Stelle angezeigt. Diese Stelle ist unabhängig vom Platz, der für den Wert reserviert ist, damit das Undefiniert-Zeichen nicht mit der Format-Anweisung für den Wert kollidieren kann (vgl. Kap. 3.2.2.). An den für den Attributwert vorgesehenen Stelle werden Leerzeichen ausgegeben.

3.4.5. Basisattribute

Als Basisattribute werden die Beziehungsattribute und die Attribute mit einem Basistyp als Wertebereich bezeichnet.

3.4.5.1. Beziehungsattribute

Beziehungsattribute belegen ein Feld. Als Attributwert wird die Transfer-Identifikation des bezeigten Objektes angegeben (vgl. Kap. 3.4.3).

3.4.5.2. Dezimalzahlenattribute

Attribute, deren Wertebereich mit "Dez" angegeben wurde (Koordinaten, Längen etc.), werden immer gemäss der Wertebereichdefinition ausgegeben. Als Format gilt die Angabe ohne den Skalierungsanteil. Der Dezimalpunkt wird gesetzt, sofern er in der Definition angegeben wurde.

3.4.5.2. Koordinaten

Koordinaten bestehen aus zwei (KOORD2) oder drei (KOORD3) Feldern. Das erste Feld enthält den Ostwert, das zweite den Nordwert, das dritte die Höhe. Sind die Koordinaten undefiniert, müssen alle Felder das Undefiniert-Zeichen enthalten. Die maximale Wertlänge wird für jede Komponente separat festgelegt.

3.4.5.3. Längen, Flächenmass und Winkel

Längen, Flächen und Winkel bestehen aus einem Feld.

3.4.5.4. Zahlenbereiche

Werte von Zahlenbereichen belegen ein Feld.

3.4.5.5. Text

Ein Textattribut besteht aus einem Feld. Damit allfällige Leerzeichen innerhalb eines Textes nicht als Abschluss des Feldes aufgefasst werden, sind die Leerzeichen innerhalb des Textes durch ein anderes Zeichen zu ersetzen. Ohne weitere Angabe wird dafür der Unterstrich (ASCII-Wert 0X5F) verwendet. Abweichungen können in der Beschreibung der Transfer-Parameter (vgl. Kap. 2.2.10) definiert werden. Um Probleme bei der Darstellung der Zeichen auf den Systemen zu vermeiden, sind ohne besondere Absprachen nebst den 7-bit-ASCII-Zeichen möglichst nur die nachfolgend aufgelisteten Zeichen (im PC-Zeichensatz definiert), in jedem Fall aber nur die alfabetischen Zeichen (vg. ISO 6937/2-1983) des ISO-8859-1-Zeichensatzes (umfasst alle Umlaute und Akzente auch in Grossschrift) zu verwenden.

Verwendbare Zeichen des PC-Zeichensatzes:

```
Ä, â, ä, à, á, æ,
Ç, ç,
É, ê, ë, è, é,
î, ï, ì, í,
Ñ, ñ,
Ö, ô, ö, ò, ó,
Ü, û, ü, ù, ú
```

3.4.5.6. Datum

Das Datum belegt ein Feld. Es wird in der Form JJJJMMTT ausgegeben (JJJJ steht für das Jahr, MM für die Zahl des Monats [01 bis 12], TT für den Tag). Der 1. Dezember 1997 wird also mit 19971201 angegeben.

3.4.5.7. Aufzählung

Ein Aufzählwert belegt ein Feld. Die Angabe erfolgt numerisch mittels Laufnummern. Numeriert werden dabei alle zulässigen Werte, also alle Elemente der Definition, die keine Unteraufzählung aufweisen. Die Numerierung beginnt mit 0. Die maximale Wertlänge ergibt sich aus der maximalen Laufnummer.

Beispiel:

Im Falle der Aufzählung

```
Farbe: (rot (dunkelrot, karmin, orange), gelb, gruen (hellgruen, dunkelgruen),
blau, violett);
```

entstehen folgende Werte:

```
0 für rot-dunkelrot 1 für rot-karmin ... 3 für gelb 4 für hellgruen ... 6 für
blau 7 für violett.
```

3.4.5.8. Horizontale und vertikale Alignierung

Da horizontale und vertikale Alignierung als vordefinierte Aufzählung aufgefasst werden können, erfolgt die Ausgabe gemäss den Regeln der Aufzählung.

3.4.6. Linienattribute

Als Linienattribute werden Attribute bezeichnet, deren Wertebereich als Linie definiert sind. Die einzelnen Linienstücke des Linienzuges werden als Liniensequenz ausgegeben. Pro Stützpunkt entsteht dabei eine Zeile. Zusätzliche Zeilen entstehen je nach Art der Verbindungsgeometrie. Bei Kreisbogen wird ein zusätzlicher Punkt in der Nähe der Bogenmitte ausgegeben. Für Spezialverbindungen ist die Codierung im Rahmen der allgemeinen Codierungsregeln von INTERLIS freigestellt. Damit man mit dem gleichen Format lesen kann, soll eine Zeile den Aufbau "Zeilenkennzeichnung Punkt weitere_Angabe NL" haben.

```
Liniensequenz = [Startpt (*Verbindung*)] 'ELIN' NL.
Startpt = 'STPT' Start-Punkt NL.
Verbindung = (Gerade | Kreisbogen | Spezialverbindung) 'LIPT' Punkt NL.
Gerade = .
Kreisbogen = 'ARCP' Zwischen-Punkt NL.
Spezialverbindung = Spezialverbindung-Fremdregel.
Punkt = Koordinate [Punktinfo].
```

Für die Datenübergabe an verschiedene Systeme ist es praktisch, wenn nebst den Punktkoordinaten noch weitere Punktinformationen angegeben werden. Die Definition ist in Kapitel 2.2.8 (VERTEXINFO) beschrieben. Die Codierung ist den am Transfer Beteiligten überlassen, soll aber den Regeln von Kap. 3.1 und 3.2 genügen. Will man mit fixem Format lesen können, ist darauf zu achten, dass an Stellen, wo verschiedener Input erwartet werden muss, dennoch mit dem gleichen Format gelesen werden kann.

3.4.7. Auswertungsattribute

Als Auswertungsattribute werden Attribute bezeichnet, die nicht unmittelbar zur aktuellen Tabelle gehören, sondern aus Auswertungen anderer Tabellen entstehen. Dies gilt für die Typen Einzelfläche und Gebietseinteilung, die eine weitere Tabelle implizieren und für Beziehungsattribute. Die Definitionsmöglichkeiten sind im Kapitel 2.2.8 beschrieben. Hier ist nur der entsprechende File-Aufbau dargestellt.

```
FlächenA = (* Rand *) 'EFLA' NL.
GebietsA = FlächenA.
Rand = 'EDGE' NL (* LinAttr Liniensequenz *) 'EEDG'NL.
LinAttr = 'LATT' { BasisAttribut } NL.
```

Die Reihenfolge der Ränder ist beliebig. Ein Rand kann beliebig in einzelne Linienzüge unterteilt werden. Die Reihenfolge der Linienstücke muss so gewählt werden, dass die beschriebene Fläche jeweils rechts der aktuellen Linie liegt (der äussere Rand wird also im Uhrzeigersinn, die Enklaven im Gegenuhrzeigersinn umfahren). Die Attribute des Linienobjektes werden auf einer Zeile mit Kennzeichnung LATT ausgegeben. Diese enthält als erstes Feld die Objektidentifikation. Sofern Linienattribute definiert wurden (vgl. Kap. 2.2.6) folgen diese anschliessend.

```
TabellenA = 'TABA' Tabellen-Name NL {'SOBJ' TransfreID-Text AttributD NL}
            'ETBA' NL.
```

Es werden alle Objekte der ausgewerteten Tabelle angegeben, die eine Beziehung (Beziehungs-Attribut) zum aktuellen Objekt haben. Pro Objekt entsteht eine Sequenz SOBJ AttributD NL. Das Beziehungs-Attribut selbst wird nicht ausgegeben.

4. Der INTERLIS-Compiler

Um eine Transferdefinition formal prüfen zu können, steht der INTERLIS-Compiler zur Verfügung.

Der INTERLIS-Compiler erzeugt in seiner einfachsten Form aus einer Transferbeschreibung eine Liste mit den Formaten, die im entsprechenden Transferfile vorkommen können und eine Liste, die die eingegebene Beschreibung samt allfälliger Fehlermeldungen enthält.

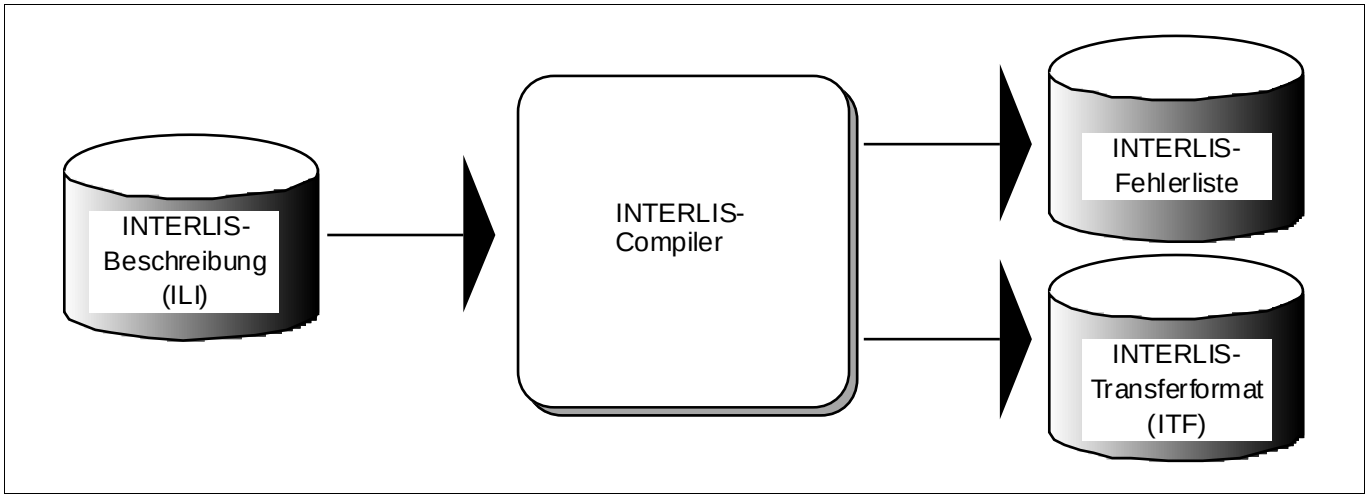

Abbildung 14: Der INTERLIS-Compiler

Der Compiler überprüft die als Input gelieferte Transfer-Beschreibung und erstellt eine Datei mit allfälligen Fehlermeldungen sowie - bei fehlerfreier Definition eine Datei, die die Formate aller zu transferierenden Tabellen beschreibt.

Wird in der Transferdefinition freies Format spezifiziert, wird dennoch eine Formatbeschreibung generiert. Es gelten folgende Annahmen:

- Zeilenlänge 60 Zeichen
- Grösse der Transferidentifikation 1 Zeichen.

5. Beispiel

Das im Kapitel 2.2.2. bereits aufgeführte Beispiel wird hier nochmals aufgenommen und näher erläutert.

```
TRANSFER Beispiel;

MODEL Beispiel

  DOMAIN
    LKoord = COORD2 480000.00   60000.00
                    850000.00  320000.00;

  TOPIC Bodenbedeckung =

    TABLE BoFlaechen =
      Art : (Gebaeude, befestigt, humusiert, Gewaesser,
             bestockt, vegetationslos);
      Form : AREA WITH (STRAIGHTS, ARCS) VERTEX LKoord
             WITHOUT OVERLAPS > 0.10;
    NO IDENT
      !! Suche ueber Form oder Gebaeude
    END BoFlaechen;

    TABLE Gebaeude =
      AssNr : TEXT*6;
      Flaeche : -> BoFlaechen // Art = Gebaeude //;
    IDENT
      AssNr;   !! Annahme AssNr sei eindeutig
      Flaeche; !! Dem Gebaeude ist genau eine Flaeche zugeordnet
    END Gebaeude;

  END Bodenbedeckung.
END Beispiel.

FORMAT FREE;

CODE
  BLANK = DEFAULT, UNDEFINED = DEFAULT, CONTINUE = DEFAULT;
  TID = ANY;
END.
```

Im Sinne des relationalen Datenmodells kann die Struktur schematisch wie folgt dargestellt werden:

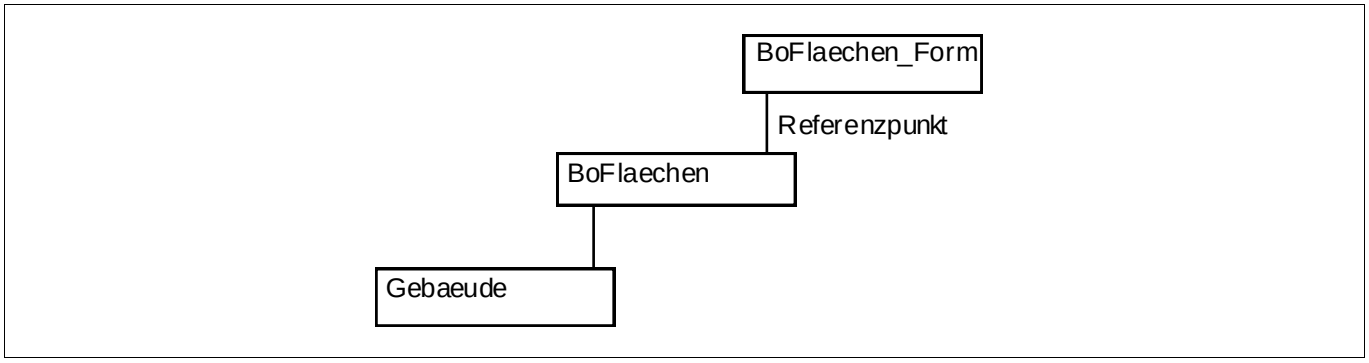

Abbildung 15: Schematische Darstellung der Beziehungen im Beispiel

Primär werden also Bodenbedeckungsflächen beschrieben. Um ihre Eigenschaft, Beschaffenheit unterscheiden zu können, wurde das Attribut Art eingeführt. Im Falle von Gebäuden soll noch weitere Information beschrieben werden, nämlich die Assekuranznummer. Da diese Informationen nur für Gebäude und nicht für alle Bodenbedeckungsflächen anfällt, wird eine spezielle Tabelle gebildet, diese aber auf diejenige der Bodenbedeckungsflächen bezogen.

Um ein konkretes Transferfile aufzeigen zu können, wählen wir einen einfachen konkreten Fall:

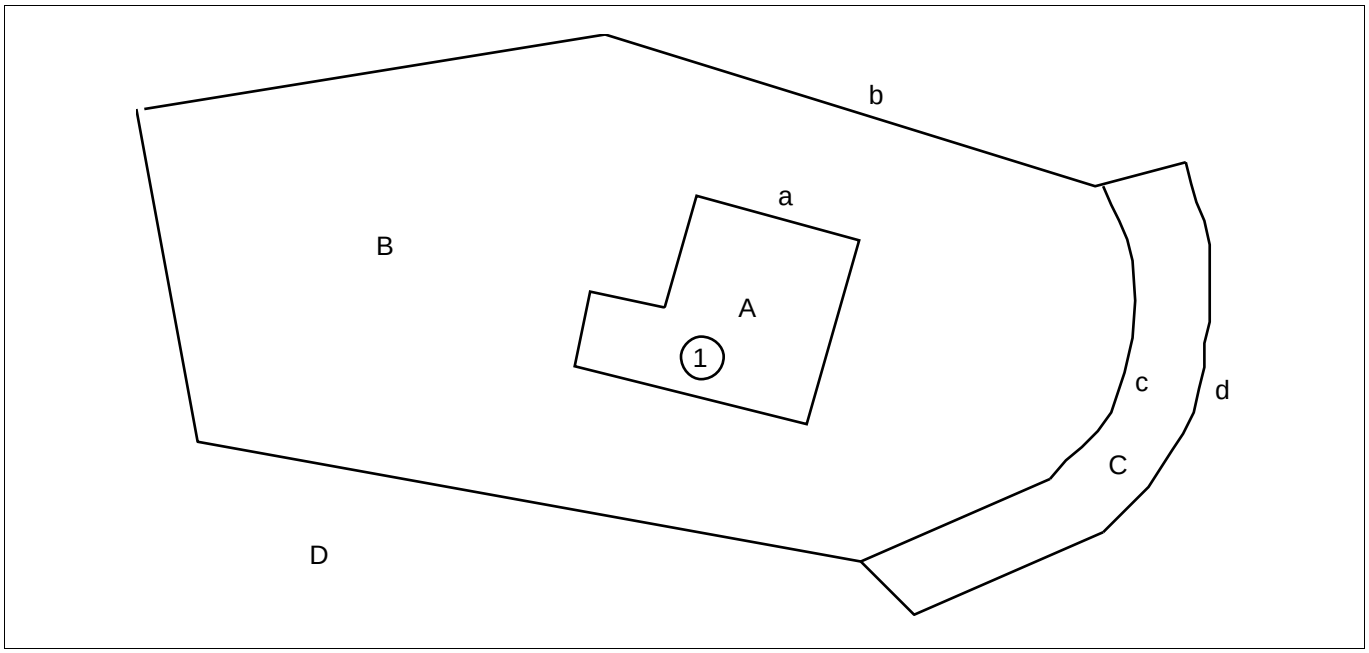

Abbildung 16: Beispiel

Ein Haus (mit der Assekuranznummer 958) liegt innerhalb der humusiertem Fläche, das an eine befestigte Fläche (Strasse) anstösst.

Mit Kleinbuchstaben sind die im Transferfile-Beispiel angewendeten Transfer-Identifikationen der Linien angegeben, mit Grossbuchstaben diejenigen der Flächen, mit eingekreister Ziffer diejenigen der Gebäude.

Der INTERLIS-Compiler übersetzt die Transferbeschreibung in folgende Formate:

```
Topic Bodenbedeckung

Table BoFlaechen_Form

OBJE-Format
OBJE 1
1: Objektident          Objektidentifikation der Linie
Followed by
Line-Records of line-attr   Line:

llab  111111.11  222222.22
1: Koordinate
2: Koordinate

Sequence of objects closed by
ETAB-Record

Table BoFlaechen

OBJE-Format
OBJE 1  2   333333.33   444444.44

1: Objektident          Die Bedeutung der einzelnen Felder wird angegeben
2: Art
3: Form
4: Form

Followed by
ETAB-Record

Table Gebäude

OBJE-Format
OBJE 1 222222 3
1: Objektident          Objektidentifikation des Gebaeudes
2: AssNr
3: Flaeche              Verweis zu Boflaeche
Sequence of objects closed by
ETAB-Record
```

Aus dem vorliegenden Beispiel ergibt sich demnach folgendes Transferfile:

```
SCNT
Transferfile des Beispiels
////
MTID gemaess Beschreibung des Beispiels
MODL Beispiel
TOPI Bodenbedeckung
TABL BoFlaechen_Form
OBJE a                          Lin-Id
STPT 600146.92 200174.98
LIPT 600138.68 200187.51
LIPT 600147.04 200193.00
LIPT 600149.79 200188.82
LIPT 600158.15 200194.31
LIPT 600163.64 200185.96
LIPT 600146.92 200174.98
ELIN
OBJE b
STPT 600140.69 200156.63
LIPT 600118.19 200179.82
LIPT 600113.00 200219.97
LIPT 600148.30 200228.97
LIPT 600186.38 200206.82
ELIN
OBJE c
STPT 600186.38 200206.82
ARCP 600183.52 200188.65
LIPT 600170.18 200176.00
LIPT 600140.69 200156.63
ELIN
OBJE d
STPT 600186.38 200206.82
LIPT 600194.26 200208.19
ARCP 600190.75 200185.21
LIPT 600174.10 200169.00
LIPT 600145.08 200149.94
LIPT 600140.69 200156.63
ELIN
ETAB
TABL BoFlaechen
OBJE A 0 600148.20 200183.48  A ist Objekt-ID; 0 für Gebäude; RefPt
OBJE B 2 600133.95 200206.06  B ist Objekt-ID; 2 für humusiert; RefPt
OBJE C 1 600168.27 200170.85  C ist Objekt-ID; 1 für befestigt; RefPt
ETAB
TABL Gebaeude
OBJE 1 958 A
ETAB
ETOP
EMOD
ENDE
```

Index

Das Beispiel in Kap. 5 wurde am 17.08.2016 durch KOGIS korrigiert.