

# Inkrementelle Nachführung mit AVS/Interlis

# Inhaltsverzeichnis

<b>1. EINLEITUNG.....</b>	<b>4</b>
1.1 AUSGANGSLAGE .....	4
1.2 ZIELE .....	4
1.3 ANFORDERUNGEN AUS BENUTZERSICHT .....	4
1.4 NUTZEN VON INKREMENTELEM DATENAUSTAUSCH .....	5
1.5 AUFBAU DER EXPERTISE .....	5
<b>2. PROBLEMSTELLUNG .....</b>	<b>6</b>
2.1. BEGRIFFE.....	6
2.2. AUFGABENSTELLUNG BEI DER NACHFÜHRUNG.....	6
2.3. AUFGABENSTELLUNG BEI DER NACHLIEFERUNG.....	8
2.4. LÖSUNGSANSATZ FÜR DIE INKREMENTELLE NACHLIEFERUNG MITTELS INTERLIS .....	8
<b>3. INTERLIS ZUSÄTZE .....</b>	<b>9</b>
3.1 EINLEITUNG .....	9
3.2 ERWEITERUNGEN DER MODELLIERUNGSSPRACHE .....	9
3.3 IDENTIFIKATION DER OBJEKTE .....	9
3.4 OPERATIONEN .....	10
3.4.1 <i>Notwendige Operationen</i> .....	10
3.4.2 <i>Codierung der Operationen</i> .....	10
3.4.3 <i>Spezielle Regeln</i> .....	11
3.5 METAINFORMATION.....	11
3.5.1 <i>Version</i> .....	12
3.5.2 <i>Art des Transfers</i> .....	12
3.5.3 <i>Absender</i> .....	13
3.5.3 <i>Empfänger</i> .....	13
3.5.4 <i>Datenbank</i> .....	13
3.5.5 <i>Anfangszustand</i> .....	13
3.5.6 <i>Endzustand</i> .....	14
<b>4. BEISPIEL .....</b>	<b>15</b>
4.1 TESTANORDNUNG.....	15
4.2 TESTABLAUF.....	15
4.3 GEWONNENE ERFAHRUNGEN .....	16
<b>5. SPEZIELLE HINWEISE .....</b>	<b>17</b>
5.1 ERZEUGUNG VON EINDEUTIGEN GID'S .....	17
5.2 NACHFÜHRUNG VON BENUTZERDATEN.....	17
5.3 ERZEUGUNG DES INKREMENTS .....	17
3.3.1 <i>Systemneutrale Werkzeuge</i> .....	18
5.4 VERARBEITUNG DES INKREMENTS.....	19
5.5 ORGANISATORISCHE MASSNAHMEN.....	19
<b>6. AUSWIRKUNGEN AUF AV93.....</b>	<b>20</b>
<b>7. EMPFEHLUNGEN .....</b>	<b>21</b>
7.1. ÜBERARBEITUNG DES INTERLIS-DOKUMENTES .....	21
7.2. OBJEKTIDENTIFIKATIONEN INNERHALB DER AMTLICHEN VERMESSUNG DER SCHWEIZ .....	21

**ANHANG**

<b>A1 LITERATURVERZEICHNIS .....</b>	<b>22</b>
<b>A2 BEGRIFFE .....</b>	<b>22</b>
<b>A3 VERWORFENE LÖSUNGSANSÄTZE .....</b>	<b>23</b>
A3.1 SPERRMECHANISMEN .....	23
A3.2 MÖGLICHKEITEN DER OBJEKTIDENTIFIKATION .....	24
A3.3 HISTORISIERUNG UND INKREMENTELLE NACHLIEFERUNG .....	24
<b>A4 INKREMENTELLER DATENSATZ ZUM TESTBEISPIEL .....</b>	<b>25</b>

# 1. Einleitung

## 1.1 Ausgangslage

Die Regelung der Fortführung (= Nachführung und/oder Nachlieferung) ist ein wichtiger Bestandteil jeder langfristigen Datenverwaltung. Die technischen und organisatorischen Probleme der Fortführung wurden bei der Realisierung der TVAV (techn. Verordnung zur amtlichen Vermessung) jedoch nicht umfassend behandelt und beantwortet. Verschiedene Endbenutzer von AV-Daten (Werke, SBB, PTT) sind an Lösungen, insbesondere der inkrementellen Nachlieferung von AV-Daten, interessiert. AVS/INTERLIS eignet sich im derzeitigen Zustand nur für den vollständigen Transfer von AV Daten [1]. Der inkrementelle Transfer von AV Daten ist mit AVS/INTERLIS im Moment nicht möglich.

## 1.2 Ziele

Diese Expertise soll erstens die Anforderungen und Probleme der inkrementellen Fortführung aufzeigen und anschliessend technische und organisatorische Lösungen formulieren. Dabei interessieren vor allem folgende Fragestellungen:

- Was versteht man unter inkrementeller Nachführung bzw. Nachlieferung ?
- Was sind die Anforderungen aus Benutzersicht ?
- Welchen Nutzen kann man von inkrementellem Datenaustausch erwarten ?
- Welche Lösungsvarianten sind denkbar ?
- Welche Voraussetzungen (Funktionen, Datenmodell) muss ein System erfüllen, damit der inkrementelle Datentransfer funktioniert ?
- Muss das Modell AV93, die Sprache INTERLIS oder das Transferformat angepasst bzw. erweitert werden ?
- Welche technischen und organisatorischen Massnahmen müssen getroffen werden ?

## 1.3 Anforderungen aus Benutzersicht

Vor jeder Lösungssuche müssen zuerst die Anforderungen an die Lösung festgelegt werden. Eine Umfrage unter potentiellen Benutzern von inkrementellem Datenaustausch (Werke, Vermessungsämter) ergab folgende Anforderungen:

- Der Datenaustausch soll schneller und einfacher funktionieren als bisher. Durch den schnelleren Datenaustausch können die Datenbestände häufiger aktualisiert werden.
- Von nachgelieferten Objekten abhängige Benutzerobjekte sollen automatisch nachgeführt werden (z.B. durch den Benutzer veränderte Textpositionen). Dadurch ergeben sich wesentliche Zeiteinsparungen beim Endbenutzer.
- Die Nachlieferung soll keinen übermässigen Verwaltungsaufwand bedingen.
- Die Anforderungen an die Systeme sollen möglichst minimal sein, damit die inkrementelle Nachlieferung von möglichst vielen Systemen unterstützt werden kann.
- Das Datenmodell AV93, die Sprache INTERLIS und die AVS sollen nur wenn nötig verändert werden.
- Internationale Normen und Entwicklungen sollen berücksichtigt werden.

Wir haben versucht die Anforderungen bei unserer Lösungssuche so weit als möglich zu berücksichtigen. Wo eine Anforderung nicht oder nur teilweise erfüllt werden konnte, werden wir darauf hinweisen.

## 1.4 Nutzen von inkrementellem Datenaustausch

Der Nutzen von inkrementellem Datenaustausch muss ebenfalls vor der Lösungssuche nachgewiesen werden. Folgender Nutzen wurde von uns festgestellt:

- Es ergeben sich zeitliche Einsparungen beim Datentransfer und bei der Nachbearbeitung der übertragenen Daten. Datenbestände können somit schneller aktualisiert werden.
- Die zu übertragenden Datenmengen sind wesentlich kleiner als beim vollständigen Datentransfer. In vernetzten Umgebungen mit langsamen oder teuren Übertragungsleitungen (Internet, WAN) und beim regelmässigen bzw. häufigen Datenaustausch (Synchronisation) ist dies ein Vorteil.
- Der Empfänger wird über die Änderungen seit der letzten Datenlieferung informiert. Die Änderungen in den Daten können daher vom Empfänger visualisiert bzw. ausgewertet werden.

Der oben aufgezählte Nutzen erscheint die genauere Untersuchung des inkrementellen Datenaustauschs zu rechtfertigen. Vor allem die Benutzerforderung nach aktuellen Daten kann mit inkrementellem Datenaustausch besser erfüllt werden als mit vollständigem Datenaustausch.

## 1.5 Aufbau der Expertise

Die weiteren Kapitel dieser Expertise sind wie folgt aufgebaut:

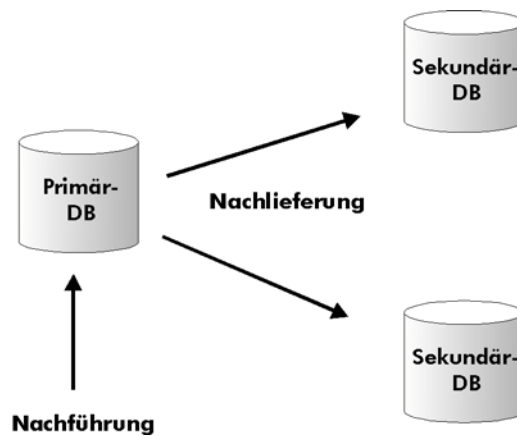
- In Kapitel 2 wird eine Übersicht über die Problematik der inkrementellen Nachführung bzw. Nachlieferung gegeben. Ausserdem wird ein allgemeines Lösungsmodell vorgestellt.
- In Kapitel 3 ist die technische Umsetzung des Lösungsmodells im Detail beschrieben. Auf Konsequenzen für INTERLIS und das Transferformat wird hingewiesen.
- In Kapitel 4 wird die vorgestellte Lösung an einem konkreten Beispiel verdeutlicht.
- In Kapitel 5 werden Hinweise zur technischen und organisatorischen Umsetzung der Lösung gegeben.
- In Kapitel 6 werden die Auswirkungen der vorgeschlagenen Lösung auf das Datenmodell AV93 besprochen.
- In Kapitel 7 werden Empfehlungen zum weiteren Vorgehen gegeben.
- Im Anhang ist ein Literaturverzeichnis, ein Verzeichnis der verwendeten Begriffe und ein komplettes Testbeispiel enthalten.

Aus Gründen der besseren Lesbarkeit des Hauptdokuments wurde auf eine Darstellung aller möglichen Lösungsvarianten im Hauptdokument verzichtet. Die verworfenen Lösungsvarianten sind jedoch im Anhang A3 zusammengestellt.

## 2. Problemstellung

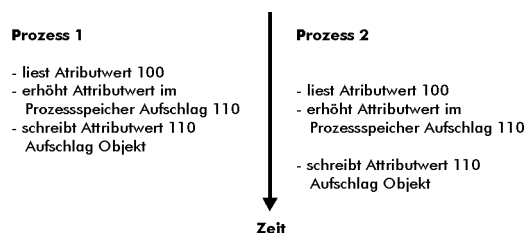
### 2.1. Begriffe

Die Datenbank, auf der die Daten bestimmter Ebenen (Topics) eines bestimmten Gebietes gehalten werden, wird als **Primärdatenbank** bezeichnet. **Nachführungen** werden auf ihr ausgeführt. Nebst der Primärdatenbank können die gleichen Daten noch auf weiteren Datenbanken gehalten werden. Solche **Sekundärdatenbanken** sind Kopien eines bestimmten Zustands der Primärdatenbank. Sie werden bei Bedarf mittels einer **Nachlieferung** wieder auf den aktuellen Stand der Primärdatenbank gebracht.



### 2.2. Aufgabenstellung bei der Nachführung

Normalerweise ist es möglich, dass mehrere Nachführungstätigkeiten gleichzeitig ausgeführt werden. Damit stellt sich das Problem, dass als Folge mehrerer paralleler Modifikationen inkonsistente Zustände entstehen können. Wird z.B. ein Objektattribut W mit dem Wert 100 parallel durch zwei Modifikationsprozesse je um 10 erhöht, ist es ohne Massnahme möglich, dass letztlich ein Resultatwert von 110 entsteht, was offensichtlich falsch ist.

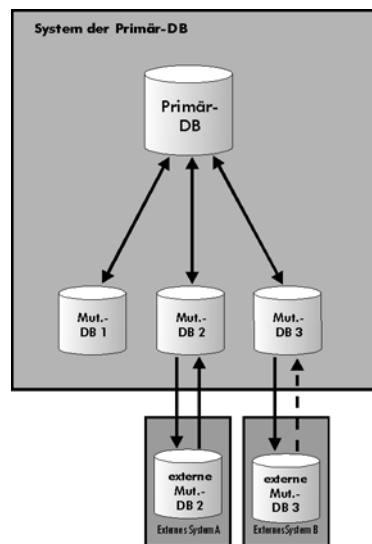


Es muss dafür gesorgt werden, dass die Modifikationsprozesse konzeptionell sequentiell ablaufen. Der erste Prozess erhöht also auf 110, der zweite beginnt bei 110 und erhöht auf 120.

Ein Blick auf konkrete GIS-Programme zeigt, dass die verschiedenen Hersteller zur Lösung des Problems recht unterschiedliche Strategien gewählt haben. Es macht keinen Sinn, dass INTERLIS auf eine dieser Strategien ausgerichtet würde. Ohne die Systemneutralität zu verlieren, kann aber davon ausgegangen werden, dass eine Nachführungstätigkeit konzeptionell wie folgt ablaufen muss:

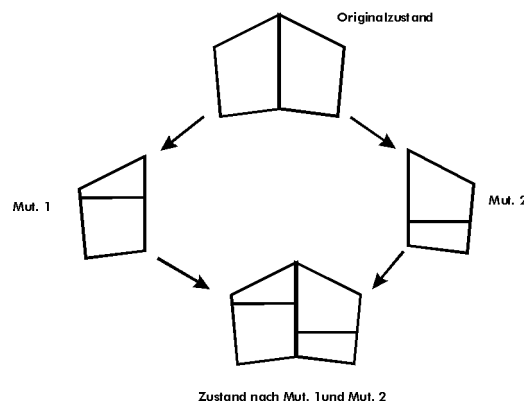
1. Bildung einer **Mutationsdatenbank** als Kopie des Zustands der Primärdatenbank vor der Nachführung.
2. Durchführung der Nachführung auf der Mutationsdatenbank.
3. Integration der modifizierten Mutationsdatenbank in die Primärdatenbank.

Dieses Konzept lässt es offen, ob die Nachführung direkt auf dem System der Primärdatenbank oder auf einem externen System ausgeführt wird. Wird ein externes System eingesetzt, müssen die Daten der Mutationsdatenbank vollständig auf dieses System transferiert werden. Für die Übermittlung der Resultate an das System der Primärdatenbank könnten die Daten wiederum vollständig transferiert werden. In diesem Fall könnte in beiden Richtungen der bisherige vollständige INTERLIS-Transfer eingesetzt werden. Aus verschiedenen Gründen (Datenmenge, Kontrollmöglichkeiten) kann es wünschbar sein, dass nur die Veränderungen mitgeteilt werden. In diesem Fall kann das Prinzip des inkrementellen INTERLIS-Transfers zur Anwendung kommen. (vgl. Kap. 2.4 und 3)



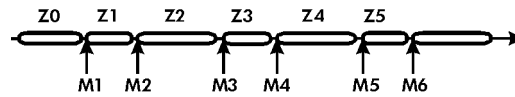
Beispiel von Nachführungen:  
 - Mut. 1: direkt auf dem System der Primär-DB  
 - Mut. 2: Bearbeitung auf dem externen System A Transfer in beiden Richtungen vollständig  
 - Mut. 3: Bearbeitung auf dem externen System B Hi-Transfer vollständig, Rück-Transfer inkrementell

Die korrekte Behandlung des konkurrierenden Zugriff und den damit verbundenen Randproblemen ist in jedem Fall Aufgabe des Systems der Primärdatenbank. Werden z.B. zwei benachbarte Gebiete in zwei unabhängigen, parallelen Mutationen geteilt, gehen primär beide Mutationen davon aus, dass sie die vollständige gemeinsame Linie aufteilen. Das System der Primärdatenbank hat das Problem zu lösen, dass die spätere Mutation nicht mehr die vollständige sondern die bereits geteilte Linie zu unterteilen hat.



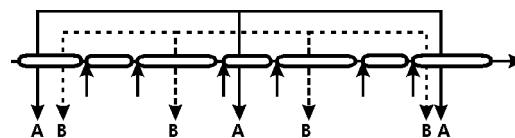
### 2.3. Aufgabenstellung bei der Nachlieferung

Wie in Kapitel 2.2. ausgeführt, muss sich die Primärdatenbank so verhalten, wie wenn die verschiedenen Nachführungstätigkeiten in einer bestimmten Sequenz und nicht teilweise parallel ausgeführt würden. Mit jeder Änderung der Primärdatenbank entsteht ein neuer **Zustand** von ihr.



Neue Zustände nach Abschluss von Nachführungen

Jeder Zustand der Primärdatenbank hat einen eindeutigen Namen. Ein bestimmter Zustand kann z.B. durch die Bezeichnung der verändernden Mutation, durch den Zeitpunkt der Integration der Änderung in die Primärdatenbank oder durch eine automatisch vergebene Laufnummer identifiziert werden. Für jeden Übergang von einem Ausgangszustand zu einem neuen Zustand muss das System der Primärdatenbank eruieren können, welche Nachlieferung dazu nötig ist. Es ist dem System der Primärdatenbank überlassen, ob es z.B. für jede Nachführung sogleich die entsprechende Nachlieferung bereitstellen will oder ob es sich die verschiedenen Zustände merken will, um eine Nachlieferung von einem beliebigen früheren Zustand zum aktuellen Zustand zusammenstellen zu können. Idealerweise kann es die Nachlieferungen entsprechend dem Bedarf der Empfänger erstellen.



Nachlieferungen für Empfänger A (——) und B (-----)

### 2.4. Lösungsansatz für die inkrementelle Nachlieferung mittels INTERLIS

Statt der erneuten Lieferung der vollständigen Daten werden nur die Veränderungen (neue Objekte, Veränderungen von Objekten, Löschung von Objekten) mitgeteilt. Um zu wissen, auf welche Objekte sich die Veränderungen beziehen, ist es nötig, dass die Objekte identifiziert werden können. Sie brauchen also eine eindeutige und stabile **Objektidentifikation**. Um einen bestimmten Transfer genauer beschreiben zu können (Sender, Datenbank, Anfangs-, Endzustand, etc.) wird dem Transfer **Metainformation** beigefügt (s.a. 3.5).



## 3. INTERLIS Zusätze

### 3.1 Einleitung

Die aktuelle Definition von INTERLIS eignet sich nur für den Transfer von ganzen Datensätzen, jedoch nicht für den inkrementellen Datentransfer. Es sind also Erweiterungen von INTERLIS notwendig. Es muss dabei zwischen Erweiterungen des Datenmodells AV93, Erweiterungen der Modellierungssprache INTERLIS und Erweiterungen des Transferformats (AVS) unterschieden werden.

Erweiterungen eines bestehenden Standards müssen immer mit besonderer Vorsicht vorgenommen werden. Im Idealfall sollen die Zusätze in keinem Widerspruch zu alten Elementen des Standards stehen. Die in den nächsten Abschnitten vorgestellten Zusätze erfüllen diese Forderung, d.h. ein Datensatz nach dem alten Standard ist gleichzeitig auch ein gültiger Datensatz nach der neuen Definition.

### 3.2 Erweiterungen der Modellierungssprache

An der Modellierungssprache INTERLIS müssen *keine* Änderungen vorgenommen werden. Dies kann wie folgt plausibel gemacht werden:

- Ob ein Datentransfer inkrementell oder vollständig durchgeführt wird, ist aus Modellierungssicht irrelevant. Der Datentransfer muss lediglich das Zielsystem von einem definierten Zustand A in einen definierten Zustand B überführen. Dies kann sowohl durch inkrementellen wie auch durch vollständigen Datentransfer erreicht werden.
- Obwohl an der Sprache INTERLIS keine Veränderungen vorgenommen werden müssen, schlagen wir vor, das Schlüsselwort IDENT in Zukunft durch UNIQUE zu ersetzen. Begründung: Mit der aktuellen Definition ist nicht die Objektidentifikation gemeint, sondern eine Konsistenzbedingung, die definiert, dass die angegebenen Attributwerte innerhalb der Tabelle eindeutig sein sollen. Dieser Sachverhalt wird besser durch UNIQUE als mit IDENT ausgedrückt. Aus Kompatibilitätsgründen wird jedoch weiterhin IDENT zugelassen.
- Die einzige zwingende Forderung die an die Systeme gestellt werden muss, ist die Verwaltung einer eindeutigen und stabilen Objektidentifikation (s.a. 3.3). Die Objektidentifikation ist bereits implizit in der Sprache INTERLIS für jede Tabelle definiert. Eine Erweiterung der Sprache um die Objektidentifikation ist daher nicht notwendig.

Da an der Modellierungssprache keine Änderungen vorgenommen werden, beziehen sich die folgenden Abschnitte nur noch auf Änderungen des Transferformats (auf Konsequenzen für das Modell AV93 wird gesondert in Kapitel 6 hingewiesen).

### 3.3 Identifikation der Objekte

Bis jetzt durfte die Transferidentifikation (TID) für jedes Objekt beim Datentransfer neu vergeben werden. Dadurch konnte ein Objekt bei mehreren Datentransfers unter verschiedenen TID's übermittelt werden. Für den inkrementellen Datentransfer ist dies nun nicht mehr erlaubt. Beim inkrementellen Datentransfer müssen alle Objekte eindeutig und stabil identifiziert werden. An die Stelle der Transferidentifikation tritt neu die **globale Objektidentifikation** (GID). Was heisst hier nun *eindeutig* und *stabil* ? Folgende Definitionen dazu:

Unter einer **eindeutigen Identifikation** verstehen wir eine Identifikation, die innerhalb einer Tabelle eines Datenmodells über eine Transfergemeinschaft hinweg eindeutig ist.

Unter einer **stabilen Identifikation** verstehen wir eine Identifikation, die während dem Lebenszyklus eines Objekts nicht verändert werden darf. Die Identifikation von gelöschten Objekten darf nicht mehr verwendet werden.

Die Transfergemeinschaft ist die Menge aller Systeme die Daten untereinander austauschen wollen. Der Aufbau der GID ist grundsätzlich freigestellt, sofern die GID eindeutig und stabil ist. In Kapitel 5 werden Hinweise zur Erzeugung der GID gegeben.

## 3.4 Operationen

### 3.4.1 Notwendige Operationen

Mit Hilfe der GID können Objekte des Zielsystems nun eindeutig identifiziert werden. Folgende Operationen sind für den inkrementellen Datentransfer notwendig und müssen vom Zielsystem bereitgestellt werden:

- **INSERT.** Ein bisher noch nicht übermitteltes Objekt soll im Zielsystem neu eingefügt werden.
- **UPDATE.** Ein oder mehrere Attribute eines Objekts des Zielsystems sollen modifiziert werden. Das Objekt wird über seine GID identifiziert.
- **DELETE.** Ein Objekt des Zielsystems soll gelöscht werden. Das Objekt wird über seine GID identifiziert.

Mit den Funktionen INSERT, UPDATE und DELETE können alle Modifikationen eines Datenbestands mitgeteilt werden (die Operationen entsprechen den Manipulationsoperationen der Sprache SQL). Die Operation UPDATE ist in einem gewissen Sinn redundant, da UPDATE auch durch eine Folge von INSERT und DELETE erreicht werden kann. Folgende Überlegungen sprechen jedoch für die Einführung der UPDATE Operation:

- Im INTERLIS Datenmodell kann es sein, dass Objekte nicht einfach gelöscht werden dürfen, da sie noch von anderen Objekten referenziert werden. Das Löschen eines referenzierten Objekts würde dann mindestens temporär eine Verletzung der Konsistenzregeln bedeuten.
- In vielen Systemen ist die Realisierung der DELETE Operation aufwendig. Es ist daher sinnvoll zu wissen, dass ein Objekt nur modifiziert werden soll.

### 3.4.2 Codierung der Operationen

Die Operationen INSERT, UPDATE und DELETE werden wie folgt auf die Transferdatei kodiert:

- Die Operation INSERT wird durch eine Transferzeile mit Zeilenkennzeichnung OBJE kodiert. Diese Definition ist kompatibel zur alten Definition, wenn man bedenkt, dass der vollständige Transfer als inkrementeller Transfer vom leeren Datenbestand auf den neuen Datenbestand verstanden werden kann.
- Die Operation UPDATE wird durch eine Transferzeile mit Zeilenkennzeichnung UPDO kodiert. Auf der Transferzeile *müssen alle* Attribute (nicht nur die geänderten) geliefert werden. Die Codierung nur der geänderten Attribute ist aufwendiger und wurde daher verworfen.

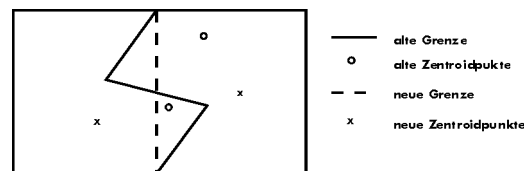
- Die Operation DELETE wird durch eine Transferzeile mit Zeilenkennzeichnung DELO kodiert. In der Transferzeile *sollen alle* Attribute (nicht nur die GID) geliefert werden. Mindestens muss jedoch die GID übertragen werden. Die Forderung, dass für DELETE Operationen alle Attribute des zu löschenden Objekts geliefert werden sollen, erscheint auf den ersten Blick überflüssig. Es gilt jedoch zu bedenken, dass auch einfache Systeme (ohne Verwaltung einer GID pro Objekt) an inkrementellen Datensätzen interessiert sein können. Diese Systeme können die in einer DELO Zeile enthaltene Information z.B. dazu nutzen, das zu löschende Objekt in einer speziellen Farbe oder auf einem speziellen Layer/Level darzustellen.

Beim inkrementellen Datentransfer müssen in der Transferdatei nur die Topics und Tables geliefert werden, die auch Daten enthalten. Für den vollständigen und initialen Transfer gelten die üblichen Regeln.

### 3.4.3 Spezielle Regeln

Die Codierung der Funktionen auf dem Transferfile ist durch die vorgestellten einfachen Regeln grundsätzlich klar. Trotzdem gibt es einige wichtige Spezialfälle für die zusätzliche Regeln gelten sollen:

- Die GID von Objekten kann nicht verändert werden. Dies ergibt sich direkt aus der Stabilitätsforderung.
- Das Empfängersystem kann davon ausgehen, dass nach der vollständigen Verarbeitung aller Daten einer inkrementellen Nachlieferung wieder ein konsistenter Zustand herrscht.
- Wenn Objekte gelöscht werden die von anderen Objekten referenziert werden, müssen immer auch die referenzierenden Objekte mit DELO oder UPDO geliefert werden.
- Bei Änderungen von Gebietslinien mit UPDATE müssen immer auch die angrenzenden Gebietsobjekte (Gebietsreferenzpunkt) mit UPDO geliefert werden.



- Auch Viewobjekte können inkrementell übertragen werden. Views sind nichts anderes als Erweiterungen einer Basistabelle um zusätzliche Attribute aus einer anderen Tabelle (Erweiterungstabelle). Als Objektidentifikation der Viewobjekte kann daher die Objektidentifikation der Basistabelle übernommen werden. Das Sendersystem muss erkennen, ob Objekte der Basistabelle oder Objekte der Erweiterungstabelle geändert haben und Viewobjekte mit der entsprechenden Operation liefern.

## 3.5 Metainformation

Die Durchführung von inkrementellen Transfers benötigt ein gewisses Mass an Verwaltungsaufwand. Z.B. muss bei einer Nachlieferung klar sein für welches System die Nachlieferung gedacht ist und von welchem Zustand des Datenbestands die Nachlieferung ausgeht. Diese **Metainformation** könnte grundsätzlich zwischen den Transferparteien mündlich oder schriftlich zusammen mit den Daten ausgetauscht werden. Um Fehler zu vermeiden und um das Verfahren automatisieren zu können (z.B. für die Synchronisation) ist eine Formalisierung dieser Mitteilungen jedoch sinnvoll. Folgende Metainformation soll zusätzlich im Vorspann (SCNT) der Transferdatei mitgeliefert werden:

- Angabe der INTERLIS Version.
- Angabe der Art des Tansfers.
- Angabe des Absenders.

- Angabe des Empfängers.
- Benennung der Datenbank.
- Benennung des Anfangszustands der Datenbank.
- Benennung des Endzustands der Datenbank.

Syntax:

```

VersionsInfo = ['VERS' Version]
TransferInfo = ['TINF'
               Art Absender [Empfänger]
               [Datenbank Anfangszustand Endzustand]
               'ETIF']

```

Die Angaben sind optional um die Kompatibilität zur alten INTERLIS Beschreibung zu gewähren. Falls Angaben weggelassen werden, gelten implizit die Standardwerte. Andere Angaben innerhalb des SCNT Vorspanns werden weiterhin als Kommentar betrachtet.

### 3.5.1 Version

Mit den inkrementellen Zusätzen zu INTERLIS wird eine erste Änderung am ursprünglichen Standard vorgeschlagen. Es ist daher sinnvoll, dass in Zukunft die gültige INTERLIS-Version aus der Transferdatei abgelesen werden kann. Die Versionsinformation soll als erste Zeile des Transferdatei geliefert werden, also noch vor SCNT.

Es werden folgende Versionsnummern festgelegt (Standardwert = 1.0):

- 1.0. Transferdatei gemäss der 1. offiziellen Fassung.
- 2.0. Transferdatei gemäss unseren Vorschlägen für die inkrementelle Nachlieferung.

### 3.5.2 Art des Transfers

Es werden folgende Transferarten festgelegt (Standardwert = FULL):

- FULL. Vollständiger Transfer, als Basis für die Nachlieferungen jedoch ungeeignet da die Objektidentifikationen nur für diesen Transfer gültig sind (TID's). Transferdateien gemäss Version 1.0 entsprechen FULL.
- INITIAL. Erste Nachlieferung. Der Datensatz enthält nur OBJE Transferzeilen mit gültigen GID's. Beim Erhalt eines INITIAL Datensatz, muss der Empfänger eine bereits vorhandene Datenbank vor dem Transfer löschen.
- UPDATE. Nachlieferung. Der Datensatz kann OBJE, UPDO und DELO Transferzeilen mit gültigen GID's enthalten. UPDATE Datensätze dürfen vom Zielsystem nur verarbeitet werden, wenn der Anfangszustand des Datensatzes bereits empfangen wurde.

Syntax:

```

Art = 'KIND' '=' ('FULL' | 'INITIAL' | 'UPDATE') ';'

```

### 3.5.3 Absender

Der Absender kann wie folgt spezifiziert werden (Standardwert = UNKNOWN):

- Name. Bezeichnung des Absenders (alphanumerische Zeichenkette ohne Leerzeichen, z.B. V+D). Die Bezeichnung des Absenders ist innerhalb einer Transfergemeinschaft eindeutig und muss von den Transferbeteiligten festgelegt werden.
- UNKNOWN. Der Absender ist unbekannt, bzw. es wurde von den Beteiligten keine verbindliche Bezeichnung festgelegt.

Syntax:

```
Absender = 'SENDER' '=' (Name | 'UNKNOWN') ';' ;'
```

### 3.5.3 Empfänger

Der Empfänger kann wie folgt spezifiziert werden (Standardwert = UNKNOWN):

- Name. Bezeichnung des Absenders (z.B. V+D). Die Bezeichnung des Empfängers ist innerhalb einer Transfergemeinschaft eindeutig und muss von den Transferbeteiligten festgelegt werden.
- UNKNOWN. Der Absender ist unbekannt, bzw. es wurde von den Beteiligten keine verbindliche Bezeichnung festgelegt.

Syntax:

```
Empfänger = 'RECEIVER' '=' (Name | 'UNKNOWN') ';' ;'
```

### 3.5.4 Datenbank

Benennung der Datenbank:

- Name. Bezeichnung der Datenbank die übertragen werden soll. Der Name der Datenbank kann vom Sender frei vergeben werden. Der Sender muss jedoch für unabhängige Datenbanken, verschiedene Namen verteilen.
- Der Name der Datenbank muss in den Transferarten INITIAL und UPDATE immer geliefert werden.

Syntax:

```
Datenbank = 'DBNAME' '=' Name ';' ;'
```

### 3.5.5 Anfangszustand

Benennung des Anfangszustands:

- Name. Bezeichnung des Datenbankzustands vor dem Transfer. Die Bezeichnung darf vom Sendersystem frei vergeben werden. Das Sendersystem muss die Zustände pro Datenbank eindeutig benennen. Der gleiche Zustandsname darf innerhalb der gleichen Datenbank nur einmal verwendet werden. Ein Zielsystem nimmt inkrementelle Daten nur entgegen, wenn der aktuelle Anfangszustand bereits als Endzustand des letzten Transfers empfangen wurde. Das Zielsystem macht den Benutzer darauf aufmerksam, wenn die empfangenen Zustände nicht zusammen passen.
- Der Anfangszustand muss in den Transferarten INITIAL und UPDATE immer geliefert werden.

Syntax:

```
Anfangszustand = 'START_STATE' '=' Name ';' ;
```

### 3.5.6 Endzustand

Benennung des Endzustands:

- Name. Bezeichnung des Datenbankzustands nach dem Transfer. Die Bezeichnung darf vom Sendersystem frei vergeben werden. Das Zielsystem merkt sich den Namen als Bezeichnung des zuletzt empfangenen Zustands.
- Der Endzustand muss in den Transferarten INITIAL und UPDATE immer geliefert werden.

Syntax:

```
Endzustand = 'END_STATE' '=' Name ';' ;
```

## 4. Beispiel

Die in Kapitel 3 vorgeschlagenen INTERLIS Erweiterungen wurden von uns an einem konkreten Beispiel überprüft. Die Testdurchführung und die aus dem Test gewonnenen Erfahrungen sind in diesem Kapitel beschrieben.

### 4.1 Testanordnung

Für den Test wurden folgende Systeme und Datensätze benutzt:

- Als Primärdatenbank wurde ein GIS System eines Schweizer Herstellers benutzt (Landinformationssystem).
- Als Sekundärdatenbank wurde ein verbreitetes CAD System mit angeschlossener Sachdatenbank eingesetzt (Werksystem).
- Als Testdatensatz wurde der Testdatensatz Nr. 2 der Eidg. Vermessungsdirektion benutzt und die im Fragebogen zum Pilotprojekt AVS/INTERLIS verlangten Mutationen [3] als Vorlage für die inkrementelle Nachlieferung verwendet.

### 4.2 Testablauf

Der Test wurde wie folgt durchgeführt:

1. Der Testdatensatz Nr. 2 (zu Beziehen bei der eidg. Vermessungsdirektion) wurde in die Primärdatenbank und die Sekundärdatenbank geladen (Transferart: INITIAL). Die Metainformation in der Transferdatei wurde wie folgt definiert:

```
VERS 2.0
SCNT
TINF
  KIND          = INITIAL;
  SENDER        = V+D;
  DBNAME        = Test2;
  START_STATE  = Leer;
  END_STATE     = Original;
ETIF
////
```

2. In der Primärdatenbank wurden die Mutationen zum Testdatensatz Nr. 2 durchgeführt (Erfassung eines Gebäudes, Teilung einer Parzelle, Mutation eines Flurnamens) und daraus ein inkrementeller Datensatz erzeugt. Der vollständige inkrementelle Datensatz ist in Anhang A4 abgedruckt. Die Metainformation für die inkrementelle Nachlieferung wurde wie folgt definiert:

```
VERS 2.0
SCNT
TINF
  KIND          = UPDATE;
  SENDER        = V+D;
  DBNAME        = Test2;
  START_STATE  = Original;
  END_STATE     = Mut1;
ETIF
////
```

3. Der inkrementelle Datensatz wurde in die Sekundärdatenbank eingelesen (Transferart: UPDATE).

## 4.3 Gewonnene Erfahrungen

Folgende Erfahrungen sind während dem Test gemacht worden:

- Der inkrementelle Datenaustausch funktioniert und erfüllt vor allem in puncto Reduktion der Datenmenge und Geschwindigkeit die in ihn gesetzten Erwartungen. Im Fall der Übertragung der Mutationen des Testdatensatz Nr. 2 konnte eine Reduktion der Datenmenge um den Faktor 250 und eine Erhöhung der Übertragungsgeschwindigkeit um den Faktor 50 gegenüber dem initialen Datensatz beobachtet werden.
- Der inkrementelle Datenaustausch ist sicher. Wegen der automatischen Übermittlung der Metainformationen im SCNT Vorspann, können Fehler (z.B. doppeltes Laden des gleichen inkrementellen Datensatzes) vermieden werden.
- Obwohl für den Test keine vollständige Implementierung der inkrementellen Nachlieferung im Sender- und Zielsystem vorgenommen wurde, konnte festgestellt werden, dass sich die inkrementelle Nachlieferung mit relativ kleinem Aufwand implementieren lässt.
- Probleme werden sich vor allem bei der Implementierung der stabilen Objektidentifikation für *alle* Objekt ergeben (z.B. für Beschriftungspositionen).
- Beim Sendersystem stellt sich zudem das Problem der Aufbewahrung der gelöschten Objekte. Gelöschte Objekte können erst definitiv gelöscht werden, wenn sie an alle daran interessierten Sekundärdatenbanken übermittelt wurden.
- Die Auflösung von referenzierten Objekten ist nun aufwendiger als bisher, da der Empfänger nicht mehr davon ausgehen kann, dass alle in der Transferdatei referenzierten Objekte auch in der Transferdatei enthalten sind. Falls ein Objekte nicht in der Transferdatei aufgelöst werden kann, muss der Empfänger das Objekt in seiner Datenbank suchen.

Zusammenfassend kann gesagt werden, dass die gemachten Erfahrungen sehr positiv sind. Die technischen Probleme sind offensichtlich lösbar.



## 5. Spezielle Hinweise

### 5.1 Erzeugung von eindeutigen GID's

Das in Kapitel 3 vorgestellte Verfahren für den inkrementellen Datenaustausch funktioniert nur mit eindeutigen und stabilen Identifikationen(GID's). Die Frage nach der Erzeugung von GID's, haben wir bewusst noch offen gelassen. Dies soll nun nachgeholt werden. Nachfolgend werden zwei mögliche Lösungen vorgestellt, die sich je nach Situation mehr oder weniger gut eignen:

1. Die Objekte innerhalb einer Transfergemeinschaft sind bereits durch ein oder mehrere Benutzerattribute eindeutig und stabil identifiziert. In diesem Fall kann natürlich direkt das Benutzerattribut als GID verwendet werden. Bei mehreren Attributen kann die GID durch Zusammenhängen der Benutzerattribute wie folgt erzeugt werden:  
GID = <Attribut 1>.<Attribut 2>. . . .<Attribut n>
2. Die Objekte innerhalb jeder Primärdatenbank der Transfergemeinschaft sind eindeutig und stabil identifiziert. In diesem Fall kann die lokale Objektidentifikation durch das Hinzufügen einer Datenbankbezeichnung (DBID) global eindeutig gemacht werden: GID = <DBID>.<lokale Identifikation>. Jede Primärdatenbank innerhalb der Transfergemeinschaft erhält also eine eindeutige Bezeichnung. Diese Bezeichnung muss von den Transferbeteiligten festgelegt werden.

Je nach Sachlage kann Variante 1 oder 2 eingesetzt werden. Wesentlich ist dabei nur, dass die Transferbeteiligten vor dem Datenaustausch eine entsprechende Abmachung treffen müssen (organisatorische Massnahmen). Für die amtliche Vermessung der Schweiz geben wir in Kapitel 7 eine entsprechende Empfehlung.

### 5.2 Nachführung von Benutzerdaten

In der Praxis fügen Nutzer von Basisdaten diesen Daten häufig noch eigene Informationen zu. Bei Werken z.B. wird für die Beschriftung einer Liegenschaft oder Strasse häufig eine eigenständige Position geführt, da die Darstellungs-Anforderungen im Werkplan nicht denjenigen des Plans für das Grundbuch entsprechen.

Ohne inkrementelle Nachlieferung würde periodisch der ganze Bestand der amtlichen Vermessung neu übermittelt. Für den Empfänger ist es dann relativ mühsam, zu erkennen, wo relevante Änderungen ausgeführt wurden. Mit der inkrementellen Nachlieferung gestaltet sich dieser Prozess einfacher: Auf Grund der stabilen Objekt-Identifikation kann eine Änderungen eines Objekts klar erkannt werden. Bei einer Änderungen der Position im Grundbuchplan kann das Empfangsprogramm z.B. anmerken, dass die eigene Positionsinformation überprüft werden muss. Im Falle einer neuen Parzelle wird es die eigene Positionsinformation gleich derjenigen des Grundbuchplanes setzen und ebenfalls anmerken, dass sie noch überprüft werden muss.

Selbstverständlich sind solche zusätzlichen Massnahmen nicht direkt durch das inkrementelle INTERLIS abgedeckt. Die inkrementellen Erweiterungen von INTERLIS bieten jedoch eine gute Grundlage, um solche Aufgaben elegant zu lösen.

### 5.3 Erzeugung des Inkrements

Die Codierung des Inkrements auf die Transferdatei ist in Kapitel 3 beschrieben. Offen bleibt, wie das Inkrement von den Systemen berechnet werden kann. Dazu folgende Definition:

Das **Inkrement** ist die Differenz zwischen zwei Datenbankzuständen der Primärdatenbank.

Die Bildung des Inkrements betrifft also hauptsächlich die Primärdatenbank. Folgende Lösungsansätze sind denkbar und werden z.T. bereits heute in verschiedenen Systemen eingesetzt:

1. **Ansatz mit Zeitmarke und Mutationstyp.** In der Primärdatenbank wird pro Objekt das Datum der letzten Mutation und die Art der Mutation (INSERT, UPDATE, DELETE) festgehalten. Ausserdem muss pro Sekundärdatenbank festgehalten werden, zu welchem Zeitpunkt der letzte inkrementelle Datenaustausch stattfand. Bei einer neuen inkrementellen Nachlieferung muss nun über die Zeitmarke überprüft werden welche Objekte seit dem letzten Datenaustausch geändert haben. Nur diese Objekte werden auf die Transferdatei kodiert. Das einzige Problem, dass sich bei dieser Variante stellt, ist das gelöschte Objekte erst dann definitiv gelöscht werden können, wenn an alle Sekundärdatenbanken eine entsprechende DELETE Operation übermittelt wurde.
2. **Mengenansatz mit Zeitmarke.** Die Primärdatenbank merkt sich über die GID welche Objekte an welche Sekundärdatenbank bereits geliefert wurden. Ausserdem wird in der Primärdatenbank pro Objekt der Zeitpunkt der letzten Mutation (INSERT, UPDATE) festgehalten. Wie in Variante 1 muss bekannt sein zu welchem Zeitpunkt die letzte Nachlieferung erfolgte. Aufgrund der Mengendifferenz zwischen dem zuletzt gelieferten Zustand und dem aktuellen Zustand und dem Mutationsdatum kann entschieden werden welche Objekte sich wie verändert haben.
3. **Reiner Mengenansatz.** Die Primärdatenbank merkt sich alle Objekte (inkl. deren Attributwerte) die an die Sekundärdatenbank übermittelt wurden und vergleicht bei der Nachlieferung den aktuellen Zustand der Datenbank mit dem bereits gelieferten Zustand. Diese Variante erscheint auf den ersten Blick aufwendig, kann jedoch elegant mit systemneutralen Werkzeugen unterstützt werden (s.a. 5.3.1).

Wie man sieht, kann das Inkrement auf verschiedene Arten erzeugt werden. Es werden also keine systemspezifischen Ansätze speziell bevorzugt. Systeme die bereits Einrichtungen für die inkrementelle Nachlieferung haben (z.B. Zeitmarken), können diese Einrichtungen für die inkrementelle Nachlieferung nutzen. Selbst Systeme, die über keine derartigen Einrichtungen verfügen, können durch systemneutrale Werkzeuge unterstützt werden.

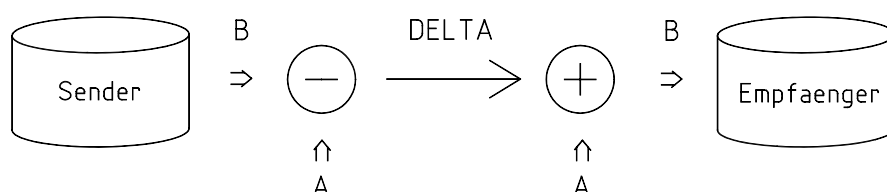
### 5.3.1 Systemneutrale Werkzeuge

Definitionen:

Ein **Additionsprozessor** ist ein Programm das aus einem vollständigen INTERLIS Datensatz A mit GID's und dem Inkrement DELTA den vollständigen Datensatz B berechnet (Formal:  $B = A + DELTA$ ).

Ein **Differenzenprozessor** ist ein Programm das aus zwei vollständigen INTERLIS Datensätzen A und B mit GID's die Differenz DELTA (Formal:  $DELTA = B - A$ ) als inkrementellen Datensatz berechnet.

Die folgende Graphik veranschaulicht den Gebrauch des Additions- bzw. Differenzenprozessors:



Der Additionsprozessor unterstützt also den Empfänger und der Differenzenprozessor den Sender von inkrementellen Daten. Beim Einsatz des Additions- und Differenzenprozessors muss weder das Sendersystem noch das Empfängersystem den inkrementellen Mechanismus selber implementieren. Einzig die Verwaltung der GID pro Objekt muss von den Systemen selbst implementiert werden.

## 5.4 Verarbeitung des Inkrements

Beim vollständigen INTERLIS-Transfer ist es möglich, die Daten sequentiell zu verarbeiten, da Referenzen immer nur zu bereits transferierten Objekten zeigen und keine anderen Abhängigkeiten zwischen verschiedenen Datenobjekten bestehen.

Beim inkrementellen INTERLIS-Transfer ergeben sich zusätzliche Abhängigkeiten. So ist es z.B. möglich, dass ein bestimmtes Empfängersystem, neue Linien erst entgegennehmen kann, wenn alte Linien, die mit der neuen kollidieren, gelöscht sind. Es entsteht also eine Abhängigkeit innerhalb der Linienobjekte. Eine solche Abhängigkeit ergibt sich aber aus dem Zielsystem, ist also nicht typisch für den Datentransfer. Beim inkrementellen Transfer werden deshalb die Daten nicht gemäss solchen Abhängigkeiten geordnet. Innerhalb einer bestimmten Tabelle ist die Reihenfolge der einzelnen Objektzeilen (egal ob neu, Änderung oder Löschung) beliebig.

Ein andere Art von Abhängigkeiten kann sich daraus ergeben, dass auf einem System Objekte erst gelöscht werden können, wenn keine Referenzen mehr darauf zeigen. Auf Grund der Reihenfolge der Tabellen in INTERLIS ist es klar, dass die Tatsache, dass ein Objekt gelöscht wird, vor der Tatsache gemeldet wird, dass eine Referenz dazu gelöscht oder durch eine andere Referenz ersetzt wird. Es ist Sache des Empfängersystem, mit solchen Situationen fertig zu werden, indem es z.B. das Transfer-File mehrmals liest oder die Objekte umordnet.

Die Initiaillieferung und jede Nachlieferung enthalten immer die Angaben über die Datenbank und den Anfangs- und Endzustand. Speichert das Empfängersystem den Datenbanknamen und den aktuellen Zustand, ist es in der Lage zu überprüfen, ob eine Nachlieferung tatsächlich den aktuellen Zustand ergänzt. Mit dem Abschluss der Verarbeitung einer Nachlieferung wird der Zustand als der aktuelle vermerkt.

## 5.5 Organisatorische Massnahmen

Die Transfergemeinschaft ist die Menge aller Systeme die Daten untereinander austauschen wollen. Bei inkrementellem Datenaustausch muss die Vergabe der GID's durch die Transferbeteiligten so festgelegt werden, dass die GID stabil und eindeutig ist (s.a. Kapitel 3). Die Regeln für die Bildung der GID sollte durch eine zentrale Stelle verbindlich festgelegt werden, um Mehrfachvergaben von GID's zu vermeiden (s.a. 5.1). Die zentrale Stelle ist auch für die allfällige Vergabe von Erzeugeridentifikationen verantwortlich (s.a. 7.2).

## 6. Auswirkungen auf AV93

Die Auswirkungen auf das Datenmodell AV93 der amtlichen Vermessung wurden bis jetzt noch nicht besprochen. Wir haben aber festgestellt, dass am Datenmodell AV93 nur *minimale* Änderungen vorgenommen werden müssen. Dies mag auf den ersten Blick überraschen, kann jedoch wie folgt begründet werden:

- Der in Kapitel 3 vorgestellte Mechanismus für den inkrementellen Datentransfer funktioniert unabhängig vom aktuellen Datenmodell.
- Der Mechanismus verlangt keine zusätzlichen Attribute pro Tabelle oder gar zusätzliche Tabellen im Datenmodell.
- Die Forderung nach eindeutigen und stabilen Identifikationen führt zur einzigen notwendigen Änderungen. Die Länge der Transferidentifikation war bis jetzt auf 10 Zeichen festgelegt (TIDSIZE=10) und als Identifikation wurden nur Zahlen zugelassen (TID=I32). Wegen der globalen Eindeutigkeit muss die Länge der Identifikation gemäss den Vorschlägen in 7.2 erweitert werden und als Typ sind nun auch alphanumerische Werte zugelassen (TID=ANY).
- Ausserdem empfehlen wir in Zukunft IDENT durch UNIQUE zu ersetzen (s.a. 3.2).

Im Datenmodell AV93 werden auch Nachführungstabellen (z.B. LFPNachführung) beschrieben. Damit können zwei Vorstellungen verbunden sein:

- Ansatz für eine inkrementelle Nachlieferung oder sogar Nachführung.
- Dokumentation der Änderungstätigkeit, indem bei neuen und veränderten Objekten eingetragen wird, im Rahmen welcher Tätigkeit (z.B. Neuvermessung, Katastererneuerung) die Veränderungen vorgenommen wurden. Der Eintrag erfolgt mittels einer Referenz auf ein Objekt der Nachführungstabelle (Mutationsobjekt), das die Tätigkeit näher beschreibt (z.B. Mutationsnummer, Datum etc.).

Obwohl die erste Vorstellung durch die von uns vorgeschlagenen Nachlieferungs- und Nachführungsverfahren abgedeckt ist, machen die Nachführungstabellen als Dokumentationstabellen der Änderungstätigkeit weiterhin Sinn. Typischerweise entsteht mit der Integration einer Mutation in eine Primärdatenbank ein neuer Zustand. Das Inkrement wird für die Nachführungstabellen der betroffenen Ebenen je ein neues Mutationsobjekt enthalten, das die Mutation beschreibt. Die neuen bzw. geänderten Objekte werden dann auf dieses Mutationsobjekt verweisen.

## 7. Empfehlungen

### 7.1. Überarbeitung des INTERLIS-Dokumentes

Die Definition der INTERLIS-Beschreibungssprache und die zugehörigen Transferregeln sind in kompakter Form beschrieben. Aus Sicht des Informatikers, der Programme im Zusammenhang mit INTERLIS erstellt, ist es wichtig, dass diese kompakte Form auch beibehalten werden kann.

Wir schlagen deshalb vor, dass der vorliegende Vorschlag zur Erweiterung von INTERLIS um die inkrementelle Nachlieferung in das INTERLIS-Hauptdokument [1] eingearbeitet wird. Diese Einarbeitung wäre auch die Gelegenheit, um einige Druckfehler und Unklarheiten im Hauptdokument zu beseitigen.

Ob andere Erweiterungen (z.B. Einführung von Zusätzen im Sinne der objektorientierten Programmierung, weitere Datentypen) bei gleicher Gelegenheit oder erst später eingefügt werden sollen, hängt vor allem davon ab, ob solche Zusätze schon genügend ausgereift sind. Unseres Erachtens sollte deswegen die Aufnahme der inkrementellen Nachlieferung nicht verzögert werden.

### 7.2. Objektidentifikationen innerhalb der amtlichen Vermessung der Schweiz

Die INTERLIS-Definition für die inkrementelle Nachlieferung lässt die Art der Objektidentifikationen offen. Um unnötigen Schwierigkeiten vorzubeugen, wäre es wünschenswert, wenn im Rahmen der amtlichen Vermessung der Schweiz ein einheitliches Prinzip für diese Identifikationen erarbeitet und sodann festgelegt würde.

Als grobe Idee schlagen wir vor, dass eine Objektidentifikation durch den Objekterzeuger festgelegt und wie folgt aufgebaut ist:

<Kantonskennzeichen>.<Erzeugerid>.<Laufnummer>

Dank einer solchen hierarchischen Struktur der Identifikation können die verschiedenen Teilnehmer weitgehend unabhängig arbeiten, ohne dass die Gefahr von Konflikten zwischen den Identifikationen besteht.

# Anhang

## A1 Literaturverzeichnis

- [1] Eidg. Vermessungsdirektion. INTERLIS ein Daten-Austausch-Mechanismus für Land-Informationen-System, Oktober 1991
- [2] Eidg. Justizdepartement. Datensatz der amtl. Vermessung, 1993
- [3] Eidg. Vermessungsdirektion. Fragebogen zum Pilotprojekt AVS/INTERLIS, 1996

## A2 Begriffe

### **System**

Gesamtheit aller zu einer EDV Anlage gehörigen Komponenten (Hardware und Software) die für einen bestimmten Zweck genutzt werden.

### **Objekt**

EDV Implementierung eines Dings der realen Welt.

### **Objektidentifikation**

Eindeutige Bezeichnung eines Objekts die während dem Lebenszyklus des Objekts (Erzeugen, Modifizieren, Löschen) nicht verändert werden darf.

### **Datenbank**

Logische Verwaltungseinheit für die Bearbeitung und persistente Speicherung von Objekten. Auf einem System können mehrere Datenbanken betrieben werden. Es ist auch möglich, dass eine Datenbank über mehrere Systeme verteilt ist.

### **Mutation**

Konsistenzerhaltende Operation auf einer Datenbank.

### **Datenbankzustand**

Benannter Zustand einer Datenbank. Eine Datenbank wird durch eine oder mehrere Mutationen von einem benannten Zustand in den nächsten übergeführt (Nachführung).

### **Primärdatenbank**

Datenbank in der die Objekte bestimmter Ebenen (Topic's) eines bestimmten Gebiets längerfristig verwaltet werden.

### **Sekundärdatenbank**

Kopie eines Datenbankzustands einer Primärdatenbank. Die Sekundärdatenbank befindet sich normalerweise nicht auf dem gleichen System wie die Primärdatenbank.

### **Mutationsdatenbank**

Temporäre Datenbank in der Objekte mutiert werden. Eine Mutationsdatenbank nimmt ihre Objekte von einer Primärdatenbank entgegen und gibt sie nach der Bearbeitung wieder an dieses zurück (Nachführung). Eine Mutationsdatenbank kann auf dem gleichen System wie die Primärdatenbank (interne Mutationsdatenbank) oder auf einem anderen System (externe Mutationsdatenbank) betrieben werden.

### **Nachführung**

Eine oder mehrere Mutationen auf einer Primärdatenbank. Eine Nachführung führt die Primärdatenbank von einem Datenbankzustand in den nächsten über. Die

Mutationen auf der Primärdatenbank können zeitlich parallel ausgeführt werden. Die Primärdatenbank muss bei parallelem Mutationen die Konsistenz des Resultats gewähren.

**Nachlieferung**

Vollständige oder inkrementelle Übertragung eines Zustands der Primärdatenbank auf eine Sekundärdatenbank. Die Nachlieferung läuft immer sequenziell ab, d.h. eine Sekundärdatenbank muss nie gleichzeitig mehrere Nachlieferungen empfangen.

**Fortführung**

Oberbegriff zu Nachführung und Nachlieferung. Eine Fortführung kann Nachführungen und/oder Nachlieferungen umfassen.

**Sendersystem**

Datenquelle eines Datentransfers.

**Zielsystem / Empfängersystem**

Datensenke eines Datentransfers.

**Datentransfer**

Übertragung von Daten vom Sender- zum Zielsystem.

**Vollständiger Datentransfer**

Vollständige Übertragung eines Datenbankzustands vom Sender- zum Zielsystem.

**Inkrementeller Datentransfer**

Übermittlung der Differenz zwischen zwei Datenbankzuständen vom Sender- zum Zielsystem.

**Transfergemeinschaft**

Gesamtheit aller Systeme mit gemeinsamen Datenmodell die Daten untereinander austauschen wollen.

**Primärdatenbank mit vollständiger Historisierung**

Primärdatenbank die jeden Datenbankzustand zu jeder Zeit wieder herstellen kann.

**Synchronisation**

Automatischer und regelmässiger Abgleich der Zustände zweier Datenbanken.

**Zeilenkennzeichnung**

Die ersten 4 Zeichen einer Transferzeile (ausser den Transferzeilen des SCNT Blocks).

## A3 Verworfenene Lösungsansätze

Während der Erstellung der Expertise wurden verschiedene Lösungsansätze für die inkrementelle Nachführung und Nachlieferung diskutiert. Wir möchten dem interessierten Leser unsere Gedankengänge nicht vorenthalten und haben deshalb hier alle wichtigen Lösungsansätze und Überlegungen zusammengestellt.

### A3.1 Sperrmechanismen

Am Anfang stand die Frage wie Objekte in der Primärdatenbank bei konkurrenzierendem Zugriff gesperrt werden können. Wir haben dazu die in heutigen Systemen verwendeten Strategien miteinander verglichen. Grundsätzlich sind folgende Ansätze im Gebrauch:

- **Es wird nichts gesperrt.** Der Benutzer ist selbst für die Konsistenz der Primärdatenbank verantwortlich.
- **Individuelles Sperren** von Objekten durch setzen eines Benutzer oder Systemattributs in der Primärdatenbank. Der erste Prozess der ein Objekt mutieren will, setzt die Sperre und gibt das Objekt erst wieder frei wenn die Mutation abgeschlossen ist. Alle anderen Prozess werden durch die Sperre am Mutieren des Objekts gehindert.

- **Sperren durch Repräsentaten.** Ein Repräsentant bezeichnet eine bestimmte Region durch ein Rechteck oder ein Polygon. Alle Objekte innerhalb der Region werden durch den Repräsentanten stellvertretend gesperrt.
- **Optimistische Verfahren mit Konfliktbereinigung.** Bei diesem Verfahren wird zunächst auf der Primärdatenbank nichts gesperrt. Bei der späteren Integration der Mutationen müssen allfällige Konflikte durch den Benutzer oder das System bereinigt werden.

Wie man sieht sind die Ansätze sehr unterschiedlich und z.T. untereinander unverträglich (individuelles Sperren verfolgt eine ganz andere Strategie als die optimistischen Verfahren). Die Art des Sperrmechanismus wird daher ganz bewusst durch INTERLIS offen gelassen. Der Sperrmechanismus muss vom System der Primärdatenbank implementiert werden (s.a. 2.2).

## A3.2 Möglichkeiten der Objektidentifikation

Folgende Variante der Objektidentifikation wurden intensiv diskutiert:

- **Eindeutige und stabile Objektidentifikation (GID).** Die von uns empfohlene Variante.
- **Eindeutige aber nur teilweise stabile Objektidentifikation.** Bei dieser Variante erhalten nur die Hauptobjekte eine stabile GID. Hauptobjekte sind Objekte die selbst keine anderen Objekte referenzieren. Diese Variante wurde verworfen, weil gezielte Mutationen von Unterobjekten (z.B. Textpositionen) praktisch unmöglich sind.
- **Globale, stabile Benutzeridentifikation.** Für jede Tabelle wird eine Attributkombination festgelegt, die das Objekt eindeutig identifiziert. Diese Variante hat den Nachteil, dass für jede Tabelle eine entsprechende Attributkombination definiert werden muss (umständlich). Benutzerschlüssel erfüllen ausserdem oft das Stabilitätskriterium nicht. Oft ist die identifizierende Attributkombination eines Objekts nicht offensichtlich (z.B. bei Textpositionen).

Wir glauben, dass die Einführung der GID die einfachste Lösung darstellt. Auch internationale Normierungen tendieren in die gleiche Richtung (s.a. OGIS).

## A3.3 Historisierung und inkrementelle Nachlieferung

Der von uns vorgeschlagene Datenaustauschmechanismus erlaubt den Austausch der Differenz zwischen zwei Datenbankzuständen Z1, Z2 der Primärdatenbank (Inkrement). Die Primärdatenbank ist nicht verpflichtet den Zustand Z1 weiter aufzubewahren, sobald sie das Inkrement von Z1 nach Z2 an alle daran interessierten Sekundärdatenbanken übermittelt hat. Aus Sicht der inkrementellen Nachlieferung ist die Aufbewahrung aller Zustände (Historisierung) also nicht notwendig. Eine Primärdatenbank mit vollständiger Historisierung kann jedoch besonders einfach die inkrementelle Nachführung implementieren.



# A4 Inkrementeller Datensatz zum Testbeispiel

```

VERS 2.0
SCNT
TINF
  KIND          = UPDATE;
  SENDER        = V+D;
  DBNAME        = Test2;
  START_STATE   = Original;
  END_STATE     = Mut1;
ETIF
////
MTID Datenkatalog
MODL Datenkatalog
TOPI Bodenbedeckung
TABL BoFlaeche_Geometrie
OBJE VD_M1_B1      0
STPT 675944.241 245329.263
LIPT 675944.237 245324.998
LIPT 675921.681 245325.016
LIPT 675921.675 245339.089
LIPT 675930.514 245339.104
LIPT 675930.521 245330.842
LIPT 675939.027 245330.838
ARCP 675941.764 245330.480
LIPT 675944.241 245329.263
ELIN
ETAB
TABL BoFlaeche
OBJE VD_M1_B2      @          675944.189 245327.131          \
CONT @              0 Test_Inkrementell
UPDO 10143         @          675902.087 245308.853          \
CONT @              8 @
ETAB
TABL Gebaeudenummer
OBJE VD_M1_B3      VD_M1_B2  3471          675923.414 245333.118 100.0 0 2
ETAB
TABL Objektname
OBJE VD_M1_B4      VD_M1_B2  Post
OBJE VD_M1_B5      VD_M1_B2  Gemeindehaus
ETAB
TABL ObjektnamenPos
OBJE VD_M1_B6      VD_M1_B4  675923.733 245340.216 100.0 0 2
OBJE VD_M1_B7      VD_M1_B5  675925.326 245327.259 100.0 0 2
OBJE VD_M1_B8      VD_M1_B4  675933.067 245332.526 0.0 0 2
ETAB
ETOP
TOPI Liegenschaften
TABL Grenzpunkt
OBJE VD_M1_L1      @          @          675722.069 245459.980 5.0 0 0 \
CONT @              @
OBJE VD_M1_L2      @          @          675759.302 245465.564 5.0 0 0 \
CONT @              @
ETAB
TABL Grundstueck
DELO 7076          @          2783          0 0          \
CONT @
OBJE VD_M1_L3      @          2783.B          0 0          \
CONT @
OBJE VD_M1_L4      @          2783.A          0 0          \
CONT @
ETAB
TABL GrundstueckPos
DELO 9076          7076          675740.595 245475.841 100.0 1 2
OBJE VD_M1_L5      VD_M1_L3  675732.995 245437.732 100.0 1 2
OBJE VD_M1_L6      VD_M1_L4  675740.595 245475.841 100.0 1 2

```

```

ETAB
TABL Liegenschaft_Geometrie
DELO 13279
STPT 675722.281 245426.781
LIPT 675722.002 245470.524
ELIN
OBJE VD_M1_L7
STPT 675722.281 245426.781
LIPT 675722.069 245459.980
ELIN
OBJE VD_M1_L8
STPT 675722.069 245459.980
LIPT 675722.002 245470.524
ELIN
DELO 13588
STPT 675759.352 245455.164
LIPT 675759.233 245480.155
ELIN
OBJE VD_M1_L9
STPT 675759.352 245455.164
LIPT 675759.302 245465.564
ELIN
OBJE VD_M1_L10
STPT 675759.302 245465.564
LIPT 675759.233 245480.155
ELIN
OBJE VD_M1_L11
STPT 675722.069 245459.980
LIPT 675759.302 245465.564
ELIN
ETAB
TABL Liegenschaft
UPDO 11019      7019      675716.189 245488.918      3141
UPDO 11021      7021      675786.486 245490.243      3358
DELO 11076      7076      675740.595 245475.841      1942
OBJE VD_M1_L12  VD_M1_L3  675757.038 245481.316      1005
OBJE VD_M1_L13  VD_M1_L4  675722.225 245443.381      938
ETAB
ETOP
TOPI Nomenklatur
TABL Flurname
UPDO 20001      @          Chrapf          675784.510      \
CONT 245342.563 @
ETAB
ETOP
EMOD
ENDE

```