VSS
Vereinigung Schweizerischer Strassenfachleute
Zürich

Forschungsprojekt 21/95

Strassendatenbank
für das Management der Strassenverkehrsanlagen

Konzeptuelle Daten-Modellierung
(Beschreibungssprache und Beispiele)

INSER SA, Lausanne
Rosenthaler + Partner AG, Muttenz

Autoren:
Chr. Rosenthaler
Rainer Oggier

Version 1.00
21.3.2001

Inhaltsverzeichnis

Version	Datum	Kommentar	Status
0.00	06. 10. 1997	Initialversion Berichtsstruktur Ro/JCJ	In Bearbeitung
0.05	08. 05. 1998	Ergänzung Ro/JCJ	In Bearbeitung
0.10	10. 10. 1998	Gesamtüberarbeitung, Revision Struktur Ro	in Bearbeitung
0.20	08. 11. 1998	Review JCJ/Ro eingearbeitet	in Bearbeitung
0.30	29. 11. 1998	Überarbeitung Ro: Review Projekt-Team 26.11.98	in Bearbeitung
0.31	12. 10. 1999	Ergänzungen OR, an PM, JCJ,Ro zugestellt	in Bearbeitung
0.32	11. 01. 2000	Ergänzungen OR	in Bearbeitung
0.35	18. 02. 2000	Ergänzungen OR nach Besprech. mit Ro	in Bearbeitung
0.40	30. 05. 2000	Layout Überarbeitung OR	in Bearbeitung
0.45	27. 07. 2000	Umstrukturierung des Berichts	in Bearbeitung
0.50	07. 08. 2000	Anpassung Dok-Vorlage R+P, Korrekturen Ro	in Bearbeitung
0.62	4.12.2000	Korrektur OR und Ro	Prüfung intern
0.69	18.12.2000	Letzte Korrekturen vor Versand an FK7	Prüfung extern
0.70	14.3.2001	Korrekturen zu OO-Definitionen	Prüfung intern
1.00	21.3.2001		Prüfung extern

1 Zusammenfassungen (inkl. Zielsetzung)

1.1 Zusammenfassung und Zielsetzung (deutsch)

Das Informationssystem für das gesamtheitliche Management der Strassenverkehrsanlage spielt eine zentrale Rolle für alle Beteiligten im Strassenmanagement. Im vorliegenden Bericht nennen wir dieses Informationssystem SMIS: Strassen-Management-Informations-System. Es muss sich auf Informatikwerkzeuge abstützen, weil nur diese die systematische Pflege der grossen Datenmenge erlauben.

Das SMIS beinhaltet einerseits eine Datenbank, die alle für das Strassenmanagement sachdienlichen Informationen enthält, und andererseits Regeln, die die Datenerhebung, -erfassung, -verwaltung -auswertung und -darstellung beschreiben, um die Bedürfnisse der Beteiligten zu befriedigen.

Die Modellierung der Strassendatenbank aus Benutzersicht (semantische Modellierung) ist in den heute vorhandenen Normen "Datenkataloge für Strassendaten (SN640940 ff.)" dokumentiert. Bis heute fehlt jedoch die Standardisierung auf der konzeptuellen Ebene, die als Pflichtenheft für die nachfolgende Informatikumsetzung dient. Ohne diese Standardisierung sind aber Aspekte wie die Austauschbarkeit, die Kombinierbarkeit und die Vergleichbarkeit der Strassendaten nicht gewährleistet.

Die übergeordnete Zielsetzung des hier dokumentierten Forschungsprojektes besteht darum darin, aus den international bekannten Lösungen eine Beschreibungssprache für die konzeptuelle Datenmodellierung auszuwählen und an einem Beispiel die Machbarkeit nachzuweisen. Diese Zielsetzung lässt sich mit den folgenden fünf Teilzielen zusammenfassen:

- Das Informationssystem SMIS definieren und seinen Lebenszyklus analysieren,
- Die Kommunikations-Bedürnisse der im SMIS Beteiligten erkennen,
- Die Entwurfs- und Umsetzungsparadigmen analysieren und beschreiben ("strukturierter Entwurf" versus "Objekt-orientierter Entwurf"),
- Die unterschiedlichen Beschreibungssprachen für die Modellierung eines Informationssystems identifizieren und beschreiben,
- Eine Sprache für das SMIS wählen und diese Sprache an einem Beispiel anwenden.

Ein wesentlicher Teil des Informationsbedarfs im Leben eines Informationssystems ergibt sich aus dem Ablauf seiner Entstehungs- und Lebensphasen. Ein Informationssystem kann diese Phasen einerseits sequenziell durchlaufen. Die entsprechenden Entwurfsmethoden werden "Wasserfallmodelle" genannt. Methoden wie Merise, HERMES oder (teilweise) das V-Modell können zu dieser Kategorie gezählt werden. Im Gegensatz zu den Wasserfallmodellen stehen andererseits Phasenmodelle, in denen die Phasen Strategie, Entwurf, Umsetzung und Einführung/Betrieb iterativ durchlaufen werden. Die Trennung dieser beiden "Sichten" ist nicht immer eindeutig, da auch in einem Wasserfallmodell innerhalb einer Phase Iterationen vorkommen können.

Die Kommunikation auf semantischer Ebene, d.h. auf der Basis der bestehenden VSS-Datenkatalog-Normen, ist für Fachleute gut geeignet. Sie ist aber für die Anforderungen an ein Pflichtenheft für die Informatikumsetzung, die sogenannte konzeptuelle Modellierung, zu wenig strukturiert. Bei jeder Entstehungs- und Lebensphase muss diese Kommunikation auf konzeptueller Ebene einerseits innerhalb des Entwurfs- oder Entwicklungsteams, und andererseits mit weiteren Beteiligten und Betroffenen gewährleistet werden.

Falls ein externer Beteiligter vom Informationssystem, und vor allem von den verwalteten Daten, profitieren will, muss auch ihm das Datenmodell in einer präzisen und vollständigen Form kommuniziert

werden. Die Beschreibung des Datenmodells auf konzeptueller Ebene genügt, vorausgesetzt, dass die Fachbegriffe dem externen Beteiligten bekannt sind. Diese Fachbegriffe werden in den Datenkatalog-Normen festgehalten.

Um das konzeptuelle Datenmodell des SMIS beschreiben zu können, d.h. ein Datenschema erstellen zu können, wurden zwei Sprachen im Detail analysiert, nämlich INTERLIS 1 und UML:

- INTERLIS 1 ist eine Daten-Beschreibungssprache, die auf der relationalen Modellierung beruht. Der strukturierte Entwurf verlangt die getrennte Betrachtung der Daten, der Organisation und der Funktionen aufgrund der unterschiedlichen Lebensdauer dieser Aspekte. INTERLIS ermöglicht dazu die verbale Beschreibung der Daten. INTERLIS 1 ist eine Schweizer Norm.
- UML (Unified Modelling Language) ist eine Sprache und eine graphische Notation, die die Nutzung der Konzepte des "Objekt-orientierten" Paradigmas (Vererbung, Kapselung, Wiederverwendbarkeit, Polymorphismus usw.) für den Entwurf und die Umsetzung eines Informationssystems erleichtert. Die Objekt-orientierte Modellierung stützt sich auf abstrakte, meist vor implementierte Konzepte und bildet sie auf die Analyse und Entwurfsebenen ab. Sie hebt die Abhängigkeiten und Interaktionen zwischen Daten, Organisation und Funktionen hervor, die im strukturierten Entwurf noch getrennt betrachtet werden.

Für die Bedürfnisse des SMIS ist die strukturierte Entwurfsmethode gut geeignet, da sie eine stabile Datenbank mit normalisierten Datenkatalogen gewährleistet. INTERLIS 1 kann benutzt werden, um Datenkataloge vollständig zu standardisieren und um konzeptuelle Datenschemata an Dritte zu liefern. Die Beschreibung der Strassendaten der bisher publizierten Datenkatalog-Normen befindet sich im Anhang zum vorliegenden Bericht. Diese Beschreibungen können später als Anhänge zu den Datenkatalog-Normen publiziert werden.

Die "Objekt-orientierte" Modellierung hingegen, und somit UML, wird in Zukunft im Entwurf und in der Umsetzung von Software vermehrt eingesetzt werden. Sie wird eine bessere Integration der Software in die Fachprozesse erlauben. Diese Aspekte müssen darum in einem nächsten Schritt von der Expertenkommission 7.03 der VSS bearbeitet werden.

Nach Abschluss der Arbeiten für den vorliegenden Bericht wurde das Handbuch von INTERLIS 2 publiziert, dass die meisten Entwurfskonzepte der Objekt-orientierten Datenmodellierung beschreibt. INTERLIS 2 wird, als verbale Notation, eine ideale Ergänzung zur graphischen Notation von UML darstellen. Darum soll in einem Nachfolgeprojekt dieser Aspekt weiter bearbeitet werden. Er konnte im vorliegenden Bericht nur noch sehr rudimentär beschrieben werden.

1.2 Résumé (français)

Le système d'information pour la gestion globale de l'infrastructure de trafic joue un rôle central pour tous les intervenants de la gestion des routes. Dans ce rapport, nous appelons ce système d'information SMIS: Strassen-Management-Informations-System. Ce système doit s'appuyer sur les outils informatiques qui permettent la mise à jour systématique de grandes quantités de données.

Le SMIS contient d'une part une base de données qui regroupe toutes les informations pertinentes pour la gestion des routes, et d'autre part des règles d'acquisition, de gestion, de traitement, d'exploitation et de représentation des données afin de satisfaire les besoins des intervenants.

La modélisation de la base de données routières du point de vue de l'utilisateur (modèle sémantique) est représentée dans les normes "catalogues de données pour les données routières (SN640940 ss.)" qui sont disponibles aujourd'hui. A ce jour, il manque une standardisation au niveau conceptuel, nécessaire à l'établissement de cahiers de charge pour la réalisation d'outils informatiques adaptés. Sans cette standardisation, la compatibilité nécessaire pour échanger, combiner et comparer les données n'est pas garantie.

Le but de ce projet de recherche consiste à sélectionner - parmi des solutions reconnues au niveau international - un langage de description pour la modélisation conceptuelle de données et à prouver la faisabilité à l'aide d'un exemple. Ce but peut être affiné par les cinq éléments suivants :

- définir le système d'information SMIS et analyser son cycle de vie,
- identifier les besoins en communication des intervenants du SMIS,
- analyser et décrire les paradigmes de conception et de mise en œuvre ("approche structuré" et "approche orienté objet"),
- identifier et décrire les différents langages de description pour la modélisation d'un système d'information,
- choisir un langage pour le SMIS et appliquer ce langage à un exemple.

Une part importante des besoins d'informations dans la vie d'un système d'information résulte du déroulement de ses phases de création et de vie. D'une part, un système d'information peut passer ces phases de manière séquentielle. Les méthodes de conception correspondantes sont appelées les "modèles en cascade". Les méthodes comme Merise, HERMES et (partiellement) le "V-Modell" peuvent être classées dans cette catégorie. A l'opposition des modèles en cascade se trouvent les modèles de phase. Elles décrivent le passage itératif des phases de stratégie, de conception, de la mise en oeuvre et de l'introduction / exploitation. La séparation de ces deux "vues" n'est par toujours univoque, puisque des itérations peuvent avoir lieu à l'intérieur d'une phase d'un modèle en cascade.

La communication au niveau sémantique, c.-à-d. sur la base des normes VSS sur les catalogues de données existantes, est bien adaptée aux spécialistes de la route. Pourtant, elle n'est pas suffisamment structurée pour les exigences envers un cahier de charge pour l'introduction d'une solution informatique (modélisation conceptuelle). A chaque phase de création et de vie, cette communication au niveau conceptuel doit être garantie d'une part à l'intérieur de l'équipe de conception ou de développement, et d'autre part avec d'autres intervenants et personnes concernées.

Si un intervenant externe au système d'information souhaite profiter du système et surtout des données, le modèle de données doit être communiqué dans une forme précise et de manière complète. La description du modèle de données au niveau conceptuel suffit à condition que les notions spécifiques à la gestion des routes soient connues à l'intervenant externe. Ces notions spécifiques sont définies dans les normes sur les catalogues de données.

Pour pouvoir décrire le modèle conceptuel des données du SMIS - c.-à-d. un schéma de données - deux langages étaient analysés en détail, à savoir: INTERLIS 1 et UML:

- INTERLIS 1 est un language de description de données qui est basé sur la modélisation relationnelle. La conception structurée exige l'analyse séparée des données, de l'organisation et des fonctions, en raison de la durée de la vie différente de ces aspects. INTERLIS complète cette conception par la description textuelle des données. INTERLIS 1 est une norme suisse.
- UML (Unified Modelling Language) est un langage et une notation graphique qui facilite l'utilisation des concepts du paradigme "orienté objet" (héritage, encapsulation, réutilisation, polymorphisme etc.) pour la conception et la mise en oeuvre d'un système d'information. La modélisation orientée objet s'appuie sur des concepts abstraits souvent pré-implémentés et les applique à l'analyse et à la conception d'un système d'information. Elle souligne les dépendances et interactions entre les données, l'organisation et les fonctions qui sont considérées séparément dans l'approche structurée.

Pour les besoins du SMIS, la méthode de conception structurée est bien adaptée, puisqu'elle garantit une base de données stable avec les catalogues de données normalisés. INTERLIS 1 peut être utilisé pour standardiser les catalogues de données de manière complète et pour fournir des schémas conceptuels à des tiers. La description conceptuelle des données routières des normes sur les catalogues de données publiées jusqu'à présent, se trouve en annexe à ce rapport. Ces descriptions peuvent être publiées plus tard en annexes aux normes sur les catalogues de données.

La modélisation orientée objet, et ainsi UML, sera appliquée de plus en plus fréquemment pour la conception et la mise en oeuvre de logiciels. Elle permettra une meilleure intégration des logiciels dans les processus métier. Ces aspects doivent être analysés dans une prochaine étape par la commission d'expert 7.03 de la VSS.

Le document de référence INTERLIS 2 a été publié après la conclusion des travaux pour ce projet. Il décrit la plupart des concepts de la modélisation orientée objet. INTERLIS 2 représentera - comme notation textuelle - un complément idéal à la notation graphique d'UML. C'est pourquoi, cet aspect doit être étudié plus en détail dans un projet supplémentaire. Ce rapport seul traite ces aspects d'une manière très rudimentaire.

1.3 Management Summary (english)

The information system for a global traffic infrastructure management plays a central role for all actors concerned about road management. In this report, we will call this information system SMIS: Strassen-Management-Informations-System. Such a system has to rely on information technology tools (IT-tools) allowing systematic update of huge quantities of data.

The SMIS contains in one hand a database which groups all relevant information for road management and in the other hand, rules for data acquisition, storage, manipulation and visualisation in order to satisfy all actors' needs.

The modeling of road databases in a user's point of view (semantic modeling) is documented in the existing VSS norms "data catalogs for road data (SN640'940 ff.)". Today, there's no standardisation at the conceptual level, which is necessary to establish spectifications for IT realisations. Without standardisation aspects like exchangebility, combinability and comparability cannot be guaranteed.

The main objective of this research project consists in choosing among internationally known solutions a description language for conceptual data modeling and in applying this language to an example in ordre to demonstrate feasability. This objective can be summarized into the following elements:

- to define an information system SMIS and to analyze its life cycle,
- to recognize the actors communication requirements,
- to analyze and to describe conception and realisation paradigms (structured versus object oriented approach),
- to identify and describe different description languages for modeling information systems,
- to choose a language for the SMIS and to apply this language to an example.

An important part of the information requirements in the life of an information system results from the passing of the creation and life phases. The information system can pass these phases sequentially. The corresponding conception methods are called waterfall models. Methods like Merise, HERMES or partially "V-Modell" are classified in this category. Opposite to waterfall models, there are phase models in which the phases "strategy", "concept", "realisation" and "introduction/exploitation" are passed iteratively. The separation of these two views isn't always cleary defined, because there may be iterations within a life phase in a waterfall model.

The communication at a semantic level, i.e. on the basis of existing normalized VSS data catalogs, is well adapted for business specialists. It is however not structured enough for requirements on specifications for an IT-realisation (conceptual modeling). In each creation and life phase, the communication at a conceptual level in one hand within conception and developpement teams and in the other hand to other actors or concerned persons has to be guaranteed.

If external actors want to take profit of the information system and its data, the data model has to be communicated in a precise form and as complete as possible. The communication at a conceptual level is suffisant, supposing that all business specific terms are known by the actor These specific terms are defined in normalized data catalogs.

In order to describe a conceptual data model of the SMIS, i.e. to establish a data schema, two languages have been analyzed in detail: INTERLIS 1 and UML.

- INTERLIS 1 is a data description language, which is based on relational modeling. The structured conception requires separate analysis of data, organisation and functions, justified by their different life duration. INTERLIS 1 completes this conception by a lexical data description. INTERLIS 1 is a Swiss norm.
- UML (Unified Modelling Language) is a language and a graphical notation which facilitates the use of concepts from the object oriented paradigm (inheritance, encapsulation, re-use, polymorphism etc.) for conception and realisation of information systems. The object oriented approach is based on abstract mostly preimplemented concepts and applies theses to analysis and conception levels . It points out dependances and interactions among data, organisation and functions which are treated separately in the structured approach.

For SMIS requirements, the structured approach is well adapted, because it assures a stable database with normalized data catalogs. INTERLIS 1 may be used to standardize data catalogs in a complete manner and to deliver conceptual data schemes to third parties. The conceptual description of the road data in the existing data catalog norms can be found as an appendix to this report. This description could later be published as appendices to the data catalog norms.

The object oriented modeling and UML however, will be applied more often in the conception and the realisation of information systems. It will allow a better integration of software into the business processes. These aspects needs to be treated by the expert group 7.03 of the VSS in a next step.

The reference manual of INTERLIS 2 has been published after finishing all works related to this project. It describes most of the concepts of object oriented data modeling. INTERLIS 2 will be as a lexical language a ideal complement to the graphical UML notation. Therefore, theses aspects needs to be analyzed in a new project. In this report, theses aspects could only be described in a very rudimentary manner.

* * *

2 Einleitung, Allgemeine Grundlagen und Gesamtprojekt

2.1 Zweck des Dokuments

Der vorliegende Bericht dokumentiert die Ergebnisse des Forschungsvorhabens VSS 21/95 (6317.01) "Konzeptuelle Datenmodellierung für Strassendatenbanken, insbesondere für den Bereich Strassenverkehr".

In diesem Bericht wird immer wieder der Bezug zu Bauprojekten hergestellt, um Parallelitäten zwischen Informatik-Projekten und Bau-Projekten aufzuzeigen. Der Leser, der sich in vergleichbaren Fragestellungen in Bauprojekten ausgekennt, kann somit die Methodik von Informatik-Projekten verstehen und daraus seine eigenen Schlüsse ziehen.

2.2 Berichtsstruktur

Die Struktur des vorliegenden Berichts kann grafisch wie folgt dargestellt werden:

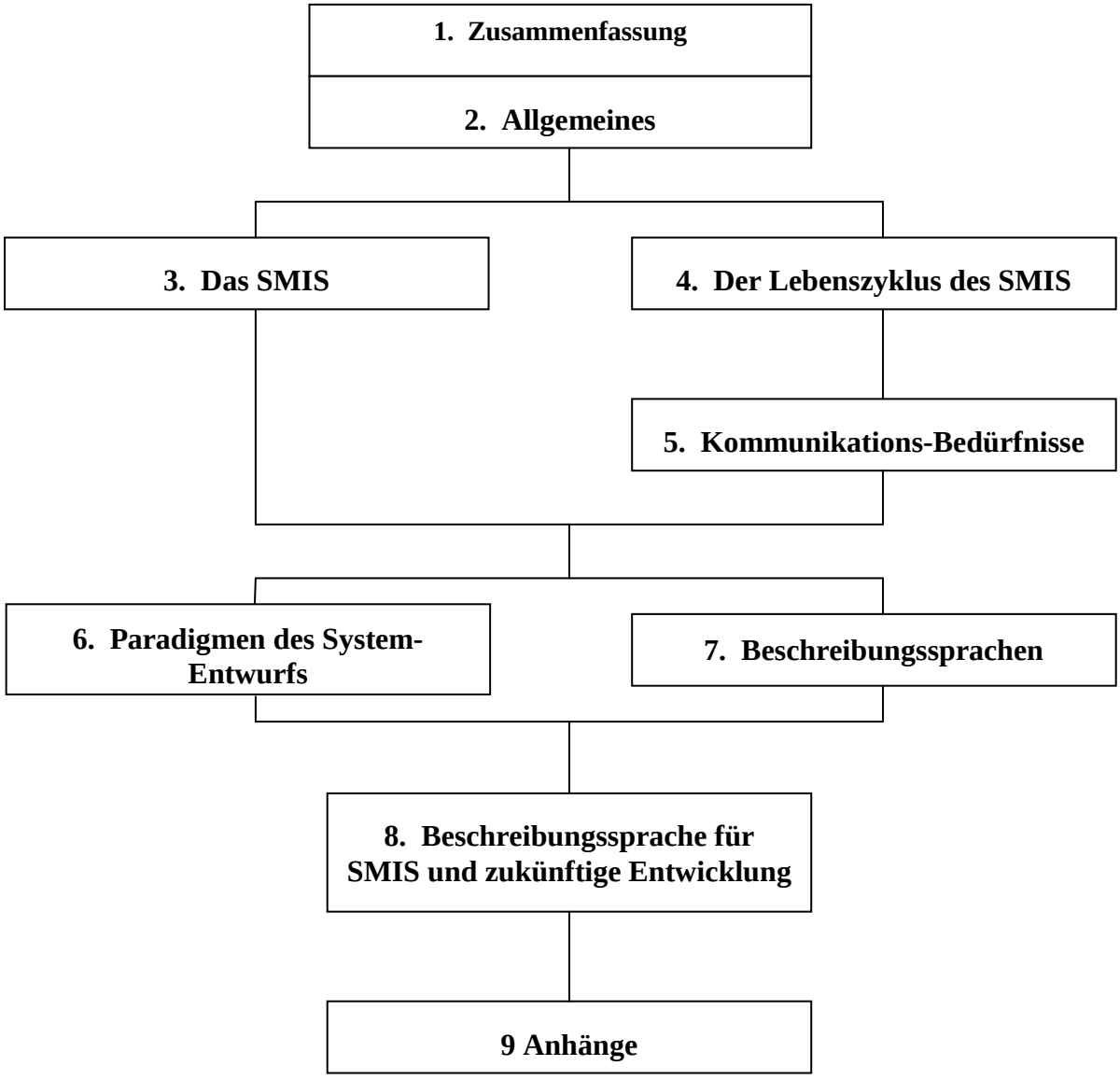


Abbildung 2-1: Struktur des Berichts

2.3 Gegenstand

Der Gegenstand des Forschungsvorhabens ist die **Strassendatenbank** für das **Informationssystem für das Management der Strassenverkehrsanlage**, kurz "Strassen-Management-Informations-System SMIS" genannt.

Im SMIS spielt die **Strassendatenbank** eine ganz zentrale Rolle. Sie stellt den langlebigen, stabilen Teil des SMIS dar. Der Entwurf dieser Strassendatenbank ist darum im Hinblick auf eine nachhaltige Nutzung des ganzen Informationssystems und somit auch aus wirtschaftlichen Gründen von hohem Stellenwert.

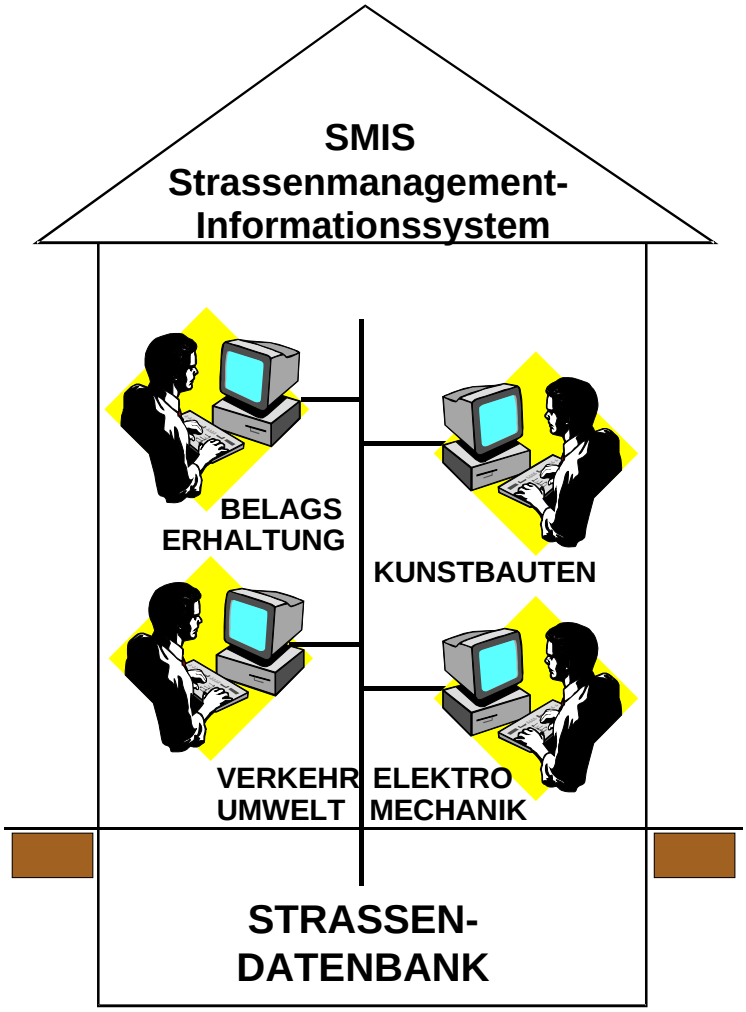


Abbildung 2-1: Themenbereiche im Strassenmanagement-Informationssystem

Die Bedeutung der Strassendatenbank kann mit den folgenden Regeln untermauert werden:

- Eine oft bestätigte Regel besagt, dass sich die gesamten Kosten während dem Lebenszyklus eines Informationssystems im Verhältnis von 1:10:100 auf die Komponenten Hardware, Software und Daten(Bank) verteilen.
- Ebenfalls gilt die Regel, dass die Lebensdauer einer bestimmten Organisation kurz, die Lebensdauer der bearbeiteten Aufgaben (Prozess-Sicht) mittel bis lang, die Lebensdauer - auch Nutzungsdauer - der bearbeiteten Informationen und Daten aber sehr lang ist.

Diese beiden Regeln können darum zu der Aussage kombiniert werden:

Der Entwurf und der Betrieb eines Informationssystems muss sich mit abnehmendem Gewicht auf die folgenden Aspekte konzentrieren:

- **Datenerhebung/-erfassung und Datenpflege (Integrität: sehr hohe Relevanz)**
- **Datenentwurf / Datenmodellierung (hohe Relevanz)**
- **Architektur des Informationssystems (mittlere bis hohe Relevanz)**
- **Software / Funktionalität (geringe bis mittlere Relevanz)**
- **Hardware / Technik (geringe Relevanz)**

In der Strassendatenbank stellt der **Entwurf der Datenstrukturen (die Datenmodellierung)** also den langlebigen, d.h. stabilen Teil dar. Der Dokumentation dieses Daten-Entwurfs, d.h. der Dokumentation der Datenstrukturen und der dazugehörenden Regeln, muss darum ein sehr hoher Stellenwert gegeben werden. Dies geschieht insbesondere im Hinblick auf den Austausch von Strassendaten und Metadaten zwischen den Beteiligten.

Ähnliches gilt analog auch für Bauprojekte im Strassenbereich!

2.4 Geltungsbereich

Die Resultate des Forschungsvorhabens gelten für die Strassendatenbanken gemäss den Normen VSS/SN 640'909 und Folgende. Sie sind aber auch für andere Strassendatenbanken anwendbar.

2.5 Referenzierte Dokumente

Siehe Anhang B Bibliographie

2.6 Begriffe

Siehe Anhang C Glossar

* * *

3 Das Informations-System für das Strassenmanagement SMIS

3.1 Übersicht über das SMIS

3.1.1 Zielsetzung und Aufgaben des SMIS

Das **SMIS** ist das Informationssystem des **"Managements der Strassenverkehrsanlage", kurz "Strassenmanagement"**. Diese Managementaufgabe umfasst alle Prozesse für den Bau, den Unterhalt, den Betrieb und die Nutzung aller Elemente einer Strassenverkehrsanlage.

Das SMIS muss alle Aktivitäten dieses Strassen-Managements unterstützen. Es stellt den für das Strassenmanagement Verantwortlichen **primär die für ihre strategischen und taktischen Entscheide** notwendigen Informationen zur Verfügung.

Das SMIS ist **erst sekundär für rein operationelle Aktivitäten** des Strassenmanagements (Termin- und Ablaufplanung und -kontrolle, Kostenmanagement usw.) konzipiert.

Die Informationen im SMIS haben einen **unterschiedlichen Aggregationsgrad**. Das SMIS ist in gewissen Bereichen, wenn auch in der Regel nicht langzeitlich, auch auf Detailinformation angewiesen. Dies ist besonders dort der Fall, wo im SMIS selbst unterschiedliche Aggregationen aus denselben Detaildaten aufbereitet werden müssen (Variantenstudien, Szenarien zur Entscheidungsfindung usw.).

Die Informationen im SMIS sind in der Regel **Raum- und Zeit-bezogen**. Dies ist eine für das SMIS sehr charakteristische Eigenschaft. Sie hat weitreichende Konsequenzen, sowohl auf die Datenmodellierung als auch auf die Funktionen auf diesen Daten.

Das SMIS muss seinen Benutzern nicht nur reine Strasseninformationen zur Verfügung stellen können. Für die notwendigen Entscheide im Management-Prozess der Strassenverkehrsanlage müssen vielmehr auch **Informationen aus dem Strassenumfeld** zur Verfügung stehen (amtliche Vermessung, Leitunskataster, Raumplanung, Umwelt usw.).

Viele dieser Aussagen gelten analog auch für Bauprojekte im Strassenbereich!

3.1.2 Die Architektur des SMIS

Das SMIS besteht konzeptionell aus den folgenden drei Elementen:

- **Steuerung** des Informationssystems
- Speichern und Verwalten von **Regeln und Wissen** (Datenstrukturen der Informationen, Aufbereitungsregeln von Daten zu neuer Information)
- Speichern und Verwalten der Information: **Strassendatenbank**

Die Strassendatenbank selbst ist ähnlich aufgebaut. Sie besitzt neben den Daten ebenfalls einen Regel- und einen Steuerungsteil.

Die beiden folgenden Figuren illustrieren diese konzeptionelle Systemarchitekur. Die (leicht modifizierte) Figur aus der Norm VSS/SN640909 "Strassendatenbanken, Grundlagen" zeigt den grundsätzlichen Aufbau des Informationssystems. Die Figur "MSE-Informatik" zeigt beispielhaft die Umsetzung dieser Grundsätze mit der Einbettung der Strassendatenbank STRADA-DB in das MSE-Informatik-Umfeld.

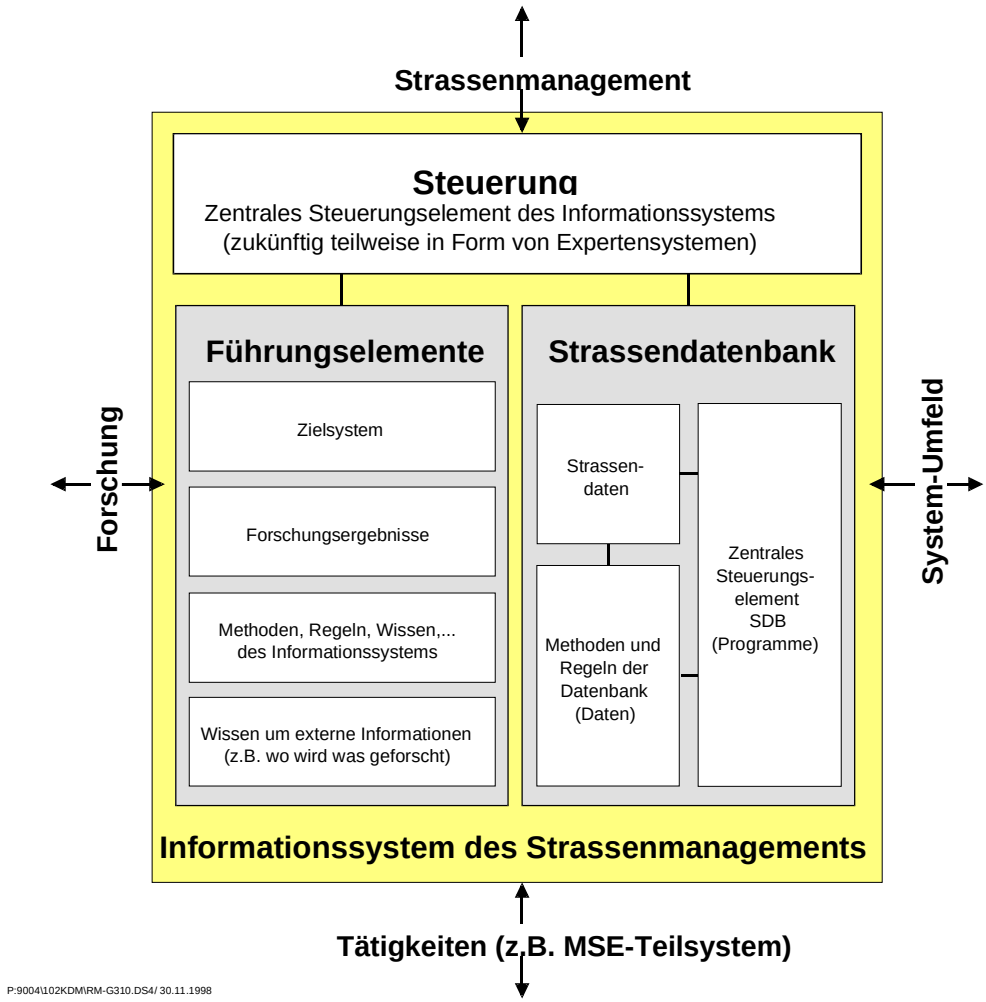
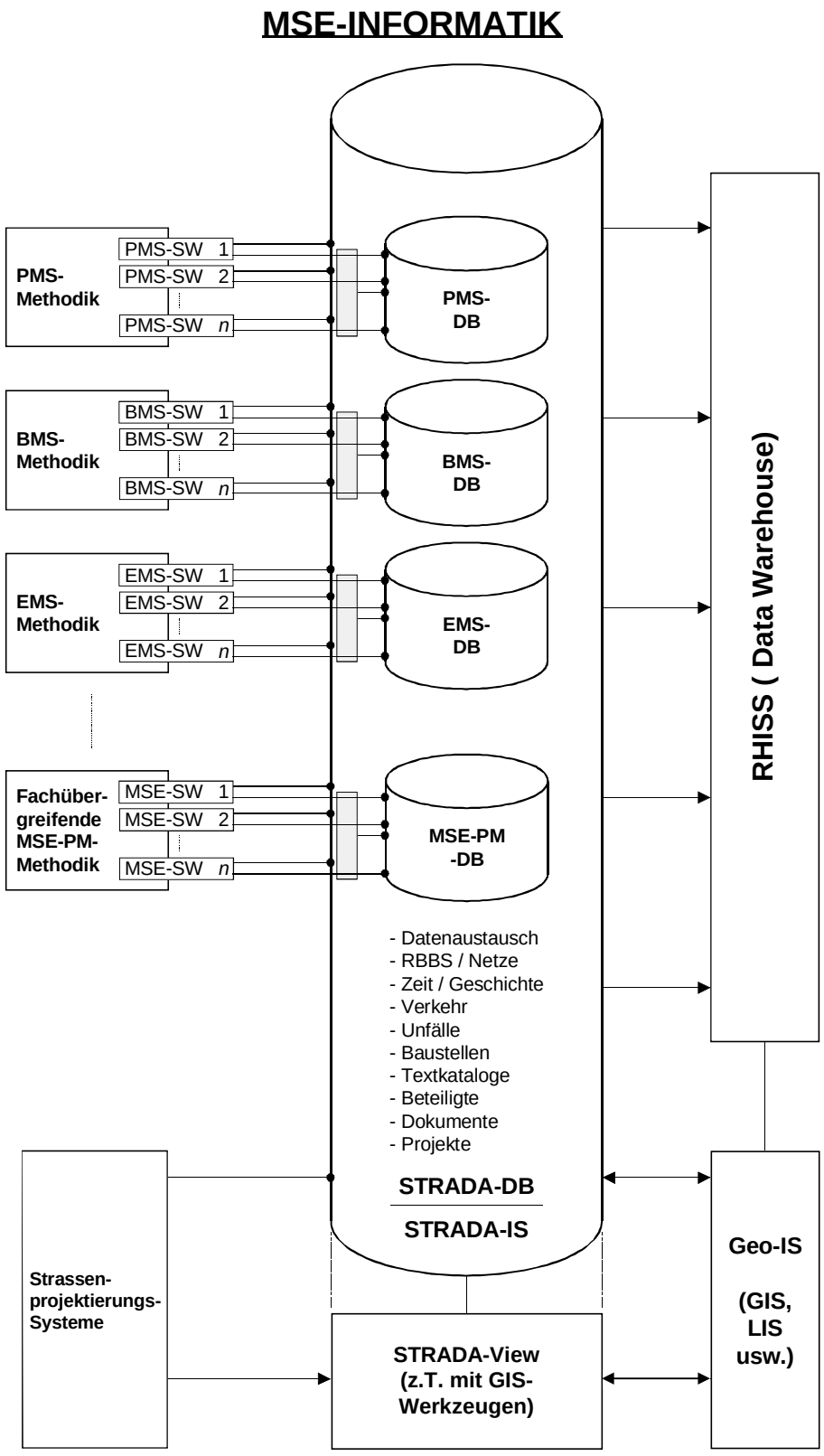


Abbildung 3-1: Systemarchitektur des SMIS



30.11.1998 / ms, by
P:\9004\102KDM\RM-G320A.DSF

Abbildung 3-2: Systemarchitektur der MSE-Informatik

3.1.3 Spezielle Fragestellungen

Das SMIS ist konzeptionell ein **verteiltes System**, da die Aufgaben im Strassenmanagement von sehr vielen Beteiligten, und an sehr vielen Orten, durchgeführt werden. Die Figur auf der folgenden Seite zeigt dies am Beispiel von STRADA-DB.

In der Schweiz ist das SMIS zudem ein **mehrsprachiges System**. Dies muss schon im konzeptionellen Entwurf berücksichtigt werden.

Das SMIS verwaltet aus seiner Zielsetzung heraus nicht nur Substanz, Geschehen und Zustand. Ein ebenso wichtiges Element ist gespeichertes Wissen. Dazu dienen unter anderem die sogenannten **Wissens (Text-) Kataloge**.

3.2 Funktionen des SMIS (Applikations-Sicht)

Im Folgenden wird der Begriff "**Strassendaten**" für alle Daten verwendet, die das SMIS für das Strassen-Management vorhalten muss. Wie erwähnt betrifft dies nicht nur die eigentlichen Daten der Strassenverkehrsanlage, sondern auch vielfältige Daten aus dem Strassenkontext.

3.2.1 Erheben der Strassendaten (von Strassen-Informationen)

Strasseninformationen, resp. die dazu erforderlichen Strassendaten, müssen vor deren Erfassung in das SMIS erhoben werden. **Erheben heisst Suchen, Analysieren, Identifizieren, Strukturieren, Dokumentieren;** dies alles in einer geordneten und nachvollziehbaren Art und Weise.

Eine wesentliche Voraussetzung dafür ist eine durchschaubare und klar dokumentierte Beschreibung der Modellierung dieser Daten im SMIS.

Das systematische und dokumentierte Erheben der Strassendaten ist (zusammen mit der Erfassung) der **teuerste** Teil des Betriebs eines SMIS! Information in einem SMIS, deren Herkunft nicht mehr eruierbar ist, ist wertlos!

3.2.2 Erfassen der Strassendaten

Die aus der Erhebung aufbereiteten Daten müssen auf möglichst wirtschaftliche Art und Weise in das SMIS **eingegeben, d.h. erfasst** werden können. Dabei soll der Erfasser darin unterstützt werden, dass er nur korrekte, plausible und widerspruchsfreie Daten in das System erfassen kann.

Eine wesentliche Voraussetzung dafür ist eine vollständig und klar dokumentierte Beschreibung der Modellierung dieser Daten im SMIS.

3.2.3 Abfragen und Auswerten der Strassendaten zu Strassen-Informationen

Die im SMIS vorgehaltenen Daten müssen dem Benutzer sowohl in "Daten-Form", als auch in zu Strasseninformation aufbereiteter Form, dem Benutzer zur Verfügung gestellt werden können. Dies ist, neben der langfristigen Speicherung, der wesentlichste Aspekt eines jeden Informationssystems. Die Art und Weise, wie die **Abfrage und die Auswertung** geschieht, charakterisiert unter anderem auch die unterschiedlichen Arten von Informationssystemen.

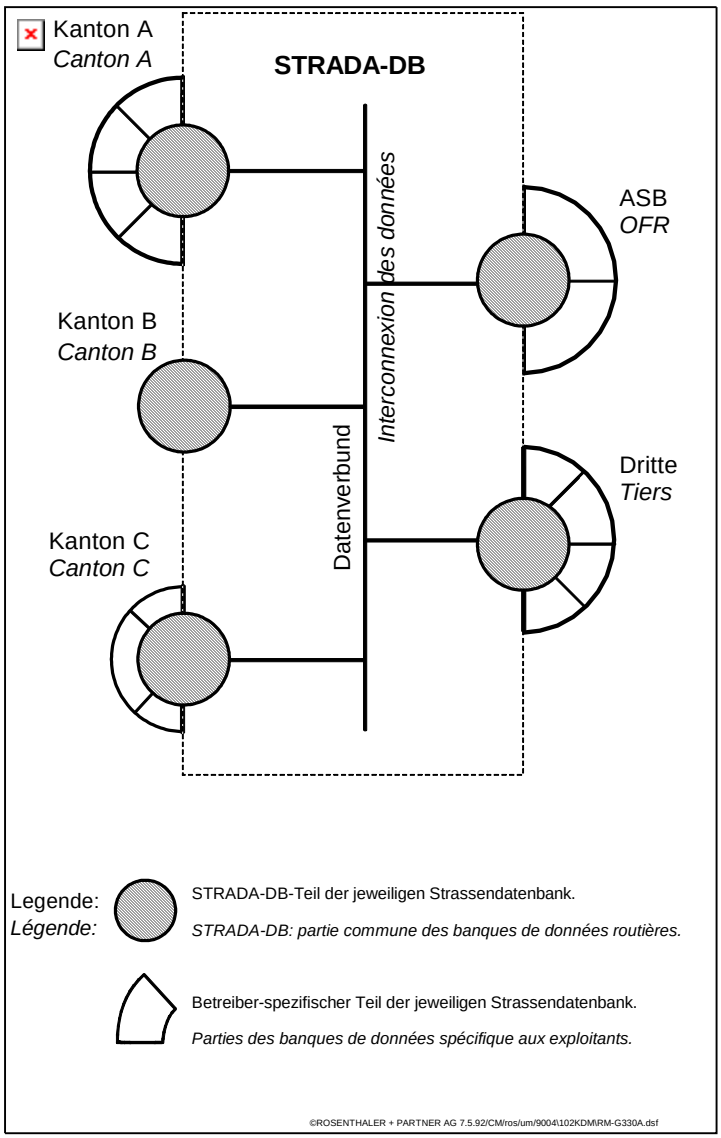


Abbildung 3-3: Kommunikation zwischen dezentralen Datenbanken (STRADA-DB)

Dabei können zwei grundsätzliche Arten von Abfragen (und Auswertungen) unterschieden werden:

- **Einfache Abfrage**: Daten eines Informationsobjekt-Typs werden nach sachlichen Kriterien dargestellt. Solche Kriterien sind in der Regel der Raumbezug, der Zeitbezug, der Bezug auf Projekte, Dokumente, Beteiligte, Einflüsse usw. Diese Abfrageart eignet sich besonders für die Datenkontrolle und für die Betrachtung zeitlicher Versionen von Objekten.
- **Kombinierte Abfrage**: Die Daten mehrerer Informationsobjekt-Typen werden räumlich und ev. zeitlich miteinander verschnitten und dann dargestellt. Diese Abfrageart eignet sich besonders für Antworten auf fachliche Fragestellungen, da hier oft auch der aus der Fragestellung resultierende Raum von Interesse ist: wo sind gewisse Bedingungen erfüllt?

Eine wesentliche Voraussetzung für beide Arten von Abfragen ist eine vollständig und klar dokumentierte Beschreibung der Modellierung dieser Daten im SMIS.

3.2.4 Pflegen der Strassendaten und der Strassendatenbank

Sowohl die Strassendatenbank als auch die darin enthaltenen Daten müssen laufend gepflegt werden.

Der Strassendatenbank-Leiter ist verantwortlich für die **fachliche Qualität und Integrität** der Datenstrukturen, Dateninhalte und des Wissens (Regeln, Wissenskataloge).

Eine wesentliche Voraussetzung dafür ist eine vollständig und klar dokumentierte Beschreibung der Modellierung dieser Daten im SMIS.

3.2.5 Darstellungsarten für Erfassen, Auswerten und Pflegen von Strassendaten

Für verschiedene Funktionen des SMIS können unterschiedliche Darstellungsarten für die Strassendaten von Nutzen sein. Im Folgenden werden die alphanumerische, die graphische und die geographische Darstellung kurz erläutert.

Dies gilt analog auch für Bauprojekte im Strassenbereich!

3.2.5.1 Alphanumerische Darstellung

Die alphanumerische Darstellung von Strassendaten und Strasseninformationen ist insbesondere für die **Datenkontrolle** (Inhalte der Sachdaten) und die **Abfrage von Sachinformationen** nützlich.

Ebenso dient die alphanumerische Abfrage für die Berechnung abgeleiteter Grössen als Entscheidungsgrundlage im Strassenmanagement. Dies können arithmetische Resultate, wie Längen, Flächen, Volumen, Kosten usw., oder auch statistische Auswertungen sein, wie Kennwerte, Extremwerte usw.

Beispiel: Die folgende Seite zeigt eine aus STRADA-DB stammende alphanumerische Auswertung.

Fahrbahnaufbauten (sortiert nach Ax-Schlüssel)

DB-Id: ASB:RP

Parameter: Gültigkeit: **Bezugsdatum:** 30.11.1998 **Versionscode:** EP **Integr.-Status:** Gültig **Original DB?:** A/
Ort: Axe: % **bis:** **Netz:** **Abschnittsgruppe:**

Anfangsort / Endort	VC	Anf.Gült. / End.Gült.	von: / bis:	Bez.-Dat. / A.-Di / Länge	A-Br. / E-Br. / Fläche	E.-Di / F.-Ti / E.-Volumen	Konto / Eigent.-Konto	Schichttyp / Zusatztext / F.-Volumen	Kosten / Kommentar / A.-Volumen	Projekt
Axe: CH:N1+										
26A : 380.0/ +2.000	E	28.05.1965 07:00	11.12.1997		4.00	0.200		ASB:FLSC		SO:ERH97-N1
26A : 700.0/ +2.000		31.05.1965 18:00	0.000		4.00	0.000			ZS ZS	ZS ZS STABI
			319.990		1279.960		255.992	0.000	0.000	
26A : 380.0/ +6.000	E	01.06.1965 07:00	11.12.1997		4.00	0.200		ASB:FLSC		SO:ERH97-N1
26A : 700.0/ +6.000		04.06.1965 18:00	0.000		4.00	0.000			ZS ZS	ZS ZS STABI
			319.990		1279.960		255.992	0.000	0.000	
26A : 700.0/ +9.250	E	22.04.1966 12:00	28.11.1997		2.50	0.080		ASB:HMT16		SO:ERH97-N1
30A : 500.0/ +9.250		22.04.1966 17:00	0.000		2.50	0.000			0.000	
			3817.000		9542.500		763.400	0.000		
26A : 700.0/ +9.250	E	26.09.1983 07:00	30.09.1998		2.50	0.010		ASB:OB3/6		SO:ERH97-N1
30A : 500.0/ +9.250		26.09.1983 18:00	0.000		2.50	0.000			0.000	
			3817.000		9542.500		95.425	0.000		
26A : 700.0/ +2.000	E	05.06.1965 07:00	11.12.1997		4.00	0.200		ASB:FLSC		SO:ERH97-N1
28A : 890.0/ +2.000		08.06.1965 18:00	0.000		4.00	0.000			ZS ZS	ZS ZS STABI
			2204.990		8819.960		1763.992	0.000	0.000	
26A : 700.0/ +6.000	E	09.06.1965 07:00	11.12.1997		4.00	0.200		ASB:FLSC		SO:ERH97-N1
28A : 890.0/ +6.000		18.06.1965 18:00	0.000		4.00	0.000			ZS ZS	ZS ZS STABI
			2204.990		8819.960		1763.992	0.000	0.000	
26A : 700.0/ +9.250	E	13.06.1965 07:00	11.12.1997		2.50	0.200		ASB:FLSC		SO:ERH97-N1
28A : 890.0/ +9.250		14.06.1965 18:00	0.000		2.50	0.000			ZS ZS	ZS ZS STABI
			2204.990		5512.475		1102.495	0.000	0.000	
26A : 735.0/ +9.375	E	12.08.1996 07:00	28.11.1997		2.75	0.090		ASB:HMT32SB6		SO:ERH97-N1

PVL0101.E01 **Ausdruck:** 30.11.98 / 11:54 **Seite:** 1

Abbildung 3-4: Resultat einer alphanumerischen Auswertung (STRADA-DB)

3.2.5.2 Grafische Darstellung

Die grafische Darstellung von Strassendaten und daraus abgeleiteten Informationen ist für den Strassenfachmann sehr geläufig. Es handelt sich dabei insbesondere um:

- **Längenprofile** (Beispiel: STRADA-View/Axband)
- **Querprofile** (Beispiel: STRADA-View/Profil)
- Grafische Darstellung **statistischer Aspekte**

Grafische Darstellungen verwenden in der Regel Ergebnisse der "einfachen Abfragen".

Beispiele: Die folgende Seiten zeigen graphische Darstellungen mit STRADA-View/Axband und STRADA-View/Profil.

3.2.5.3 Geographische Darstellung

Die geographische Darstellung erfolgt in der Regel mit:

- **Geo-Viewer** (Beispiel: STRADA-View/Carto)
- **Geo-Informationssysteme** (Beispiel: STRADA-DB/ARGIS (GE))

Geographische Darstellungen verwenden oft Ergebnisse von "kombinierten Abfragen".

Beispiele: Die folgenden Seiten zeigen (nach den Beispielen aus STRADA-View/Axband) geographische Auswertungen aus STRADA-View/Carto.

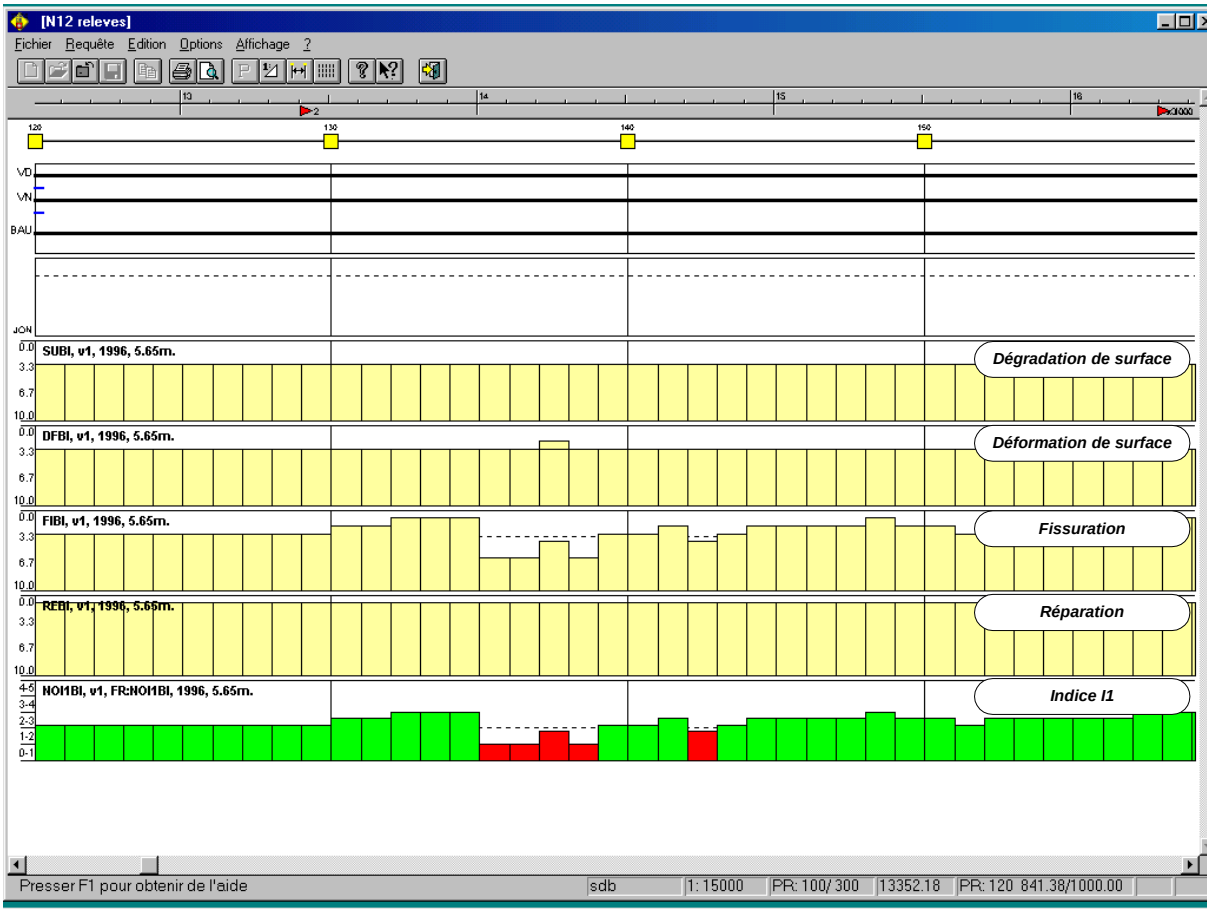


Abbildung 3-5: Graphische Darstellung in STRADA-View/Axband

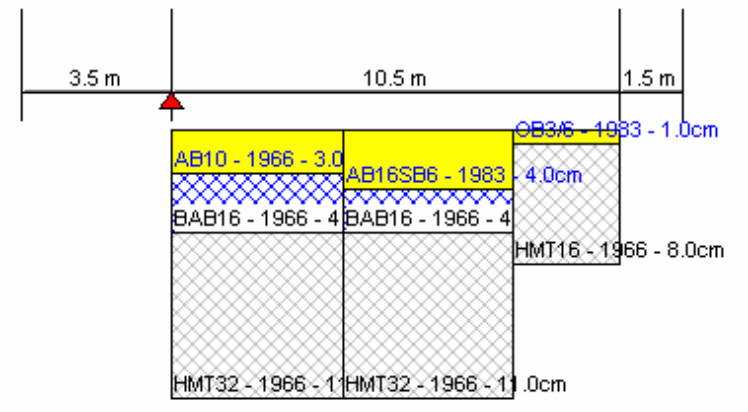


Abbildung 3-6: Graphische Darstellung eines Querprofils in STRADA-View/Profil

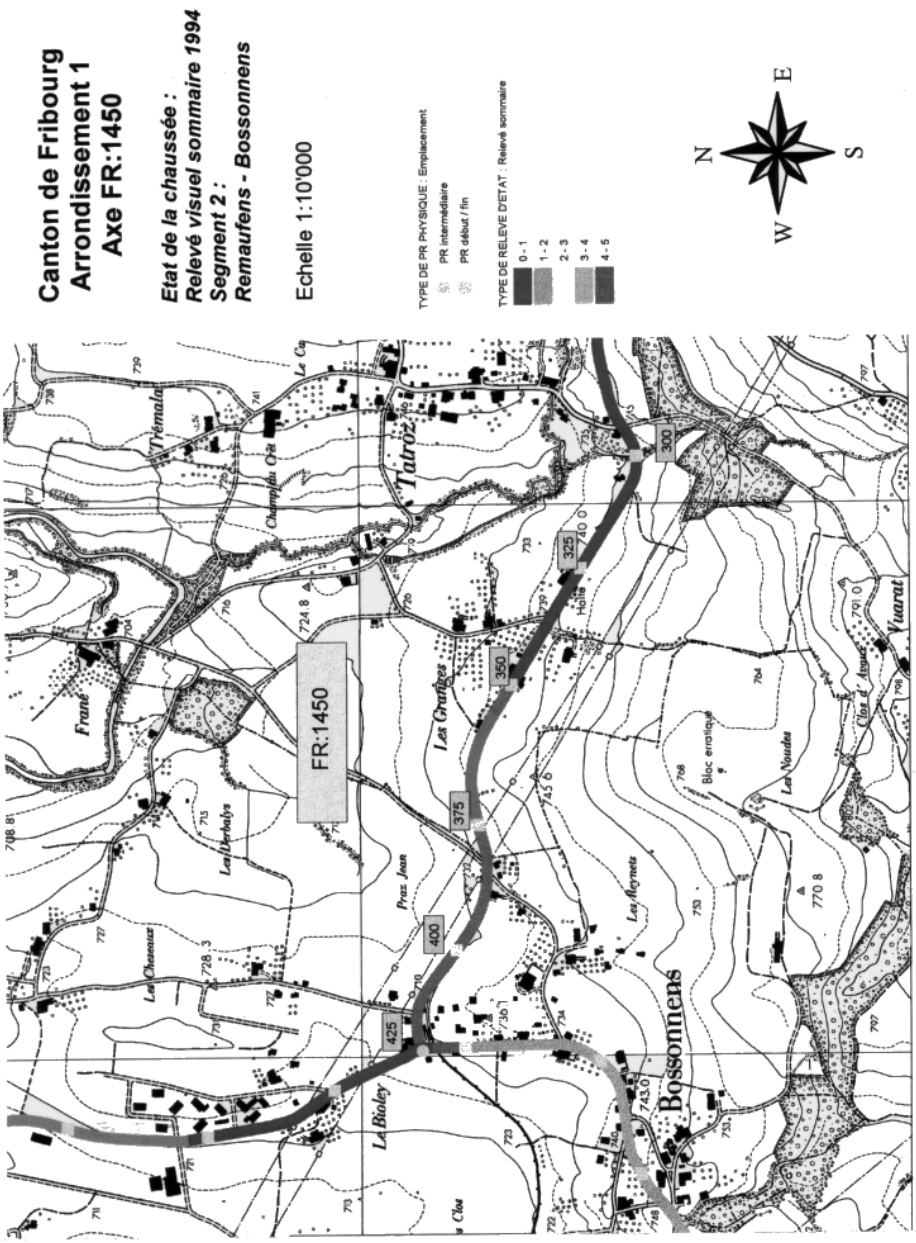


Abbildung 3-7: Kartographische Darstellung in STRADA-View/Carto

3.3 Informationen im SMIS, Daten in der Strassendatenbank

3.3.1 Datenbereiche in der SMIS-DB

Das Strassenmanagement benötigt nicht nur Informationen und entsprechende Daten über die **Strassenverkehrsanlage** und ihre Nutzung selbst. Das Strassenmanagement benötigt auch Informationen aus dem **Umfeld der Strasse**. Die Datenbereiche des SMIS können grob wie folgt beschrieben werden.

3.3.1.1 Strassenverkehrsanlage

In der Norm VSS/SN640909 werden die Datenbereiche für die Strassenverkehrsanlage in einem übergeordneten Datenkatalog beschrieben.

Dieser Katalog beinhaltet Daten aus folgenden Kategorien:

- Raumbezugssystem und Betriebsnetze
- Strassenbeschreibung inkl. Substanz
- Geschehen im Strassenraum
- Betrieblicher und baulicher Unterhalt
- Kontrolle und Zustandserhebungen
- Modelle und Simulationen
- Führung des MSE (Management der Strassenerhaltung)

3.3.1.2 Verkehrsbeeinflussung im Strassenverkehr (Mobilitäts-Management)

Ein zukünftiges Mobilitäts-Management muss sich auf folgende Daten stützen können:

- Statische Daten umfassen alle Daten, die den Strassenraum beschreiben (Infrastruktur)
- Dynamische Daten beschreiben Ereignisse oder Zustände, die eine zeitlich beschränkte Gültigkeit besitzen. Es handelt sich zum Beispiel um Daten über Unfälle, Baustellen, aussergewöhnliche Wetterverhältnisse oder temporäre Verkehrsregelungen. Diese Daten werden zur Information der Verkehrsteilnehmer genutzt, sei es über Radio, RDS-TMC oder andere Übermittlungsmethoden.
- Statistische Daten sind dynamische Daten, die im Hinblick auf statistische Auswertungen periodisch erfasst werden. Es handelt sich um Daten, die per Massenerfassung produziert werden können und für eine Prognose des Verkehrszustandes relevant sind. Solche Daten können beispielsweise von Induktionsschleifen oder Floating Car Data stammen.

Im Anhang K "Régulation du trafic" werden die verschiedenen Standards in der Verkehrsbeeinflussung bzw. Verkehrsinformation erläutert.

3.3.1.3 Amtliche Vermessung

Die AV 93 definiert in ihrer technischen Verordnung TVAV den Datenkatalog der amtlichen Vermessung. Er enthält die folgenden Informationsebenen, auch Themen genannt:

- Fixpunkte,
- Bodenbedeckung,
- diverse Objekte/ lineare Elemente,
- Nomenklatur,
- Höhenlage (Altimetrie),
- Liegenschaften,
- Leitungen, und
- administrative bzw. technische Gebietseinteilungen.

Der Strassenraum wird als eine Fläche im Thema Bodenbedeckung geführt. Die Semantik entspricht jedoch nicht den im Strassenmanagement gebräuchlichen Festlegungen.

Die Daten der amtlichen Vermessung können mit Hilfe der in INTERLIS definierten Schnittstelle (AVS) übertragen, d.h. auch bezogen werden (siehe Anhang E INTERLIS).

3.3.1.4 Leitungskataster

Der SIA hat die Norm SIA405 "Leitungen" komplett überarbeitet. Dabei wurde der gesamte Inhalt in eine neue Norm und zweie Anhänge, sogenannte Merkblätter, gegliedert (siehe Anhang L Geo405).

3.3.1.5 Raumplanung

Im Bereich der Raumplanung interessieren das Strassenmanagement insbesondere Zonen-informationen, sozio-ökonomische Grössen, Ziel-/Quell-Daten, Angebots- und Nachfrageparameter usw.

3.3.1.6 Umwelt

- Lärmbelastung durch den Strassenverkehr
- Luftbelastung durch den Strassenverkehr

3.3.2 Raumbezug im SMIS

3.3.2.1 Linearer Raumbezug (entlang Strassen-Axe)

Die eigentlichen Strassendaten beschreiben in der Regel Objekte in einem sehr **eng beschränkten und linearen Strassenraum**. Viele Prozesse des Strassenmanagements benötigen nur diesen linearen Strassenraum (z.B. PMS, Verkehrsplanung, Verkehrsökonomie usw.). Deshalb ist es sinnvoll, diesen Raum mit einem dafür geeigneten Raumbezugssystem zu beschreiben. Dies ist das räumliche Basis-Bezugssystem für Strassendaten gemäss den entsprechenden VSS-Normen [SN640910] und [SN640920].

Dies gilt analog auch für Bauprojekte im Strassenbereich!

3.3.2.2 Flächiger Raumbezug für den Strassen-Kontext

Der flächige Raumbezug wird in der Schweiz in der Regel mit den sogenannten **Landeskoordinaten** (Y, X, Höhe) hergestellt. Diese werden sowohl in der amtlichen Vermessung als auch in der Landestopographie verwendet.

Dies gilt analog auch für Bauprojekte im Strassenbereich!

3.3.2.3 Topologie

Die Strassenverkehrsanlage lebt von ihrer Topologie. Diese ermöglicht erst die vielfältigen Verkehrsbeziehungen, die der Mobilität dienen.

Es gibt nicht nur eine Topologie des schweizerischen Strassennetzes. Vielmehr werden je nach Ziel und Zweck der zu unterstützenden Fach- oder Geschäftsprozesse sehr viele, z.B. sehr **unterschiedlich fein gegliederte Topologien** definiert. Die Strassendatenbank muss darum sehr viele Möglichkeiten topologischer Strukturen auf derselben Strasse unterstützen.

Die Beschreibung der Topologie geschieht in der Strassendatenbank gemäss der VSS-Norm [SN640911].

3.3.2.4 Geometrie

Die geometrische Beschreibung der Strassen-Axe und weiterer Elemente der Strassenverkehrsanlage geschieht in der Regel im flächigen Raumbezug des Strassenkontext. Sie verbindet somit die Strasse mit diesem Kontext.

Zu beachten ist, dass somit **je Kontextbeschreibung eine Geometrie** der Strassen-Axe erforderlich ist. Die Notwendigkeit mehrerer Kontextbeschreibungen ergibt sich aus der unterschiedlichen Generalisierung dieses Kontexts und der darinliegenden Strasse, z.B. in Abhängigkeit des Massstabs der Kontext-Darstellung.

Dies gilt analog auch für Bauprojekte im Strassenbereich!

3.3.3 Zeitbezug

Ein Informationssystem für die Strassenverkehrsanlage ist massgeblich auf aussagekräftige **Zeitangaben** und die Möglichkeit der **Geschichtsschreibung** für zustandsartige Daten angewiesen (Aktivitäten können mehrere zukünftige Ausprägungen haben; die Beschreibung der effektiv geschehenen Aktivität kann nur eine Ausprägung haben).

Die Zeitaspekte einer Information werden in der Strassendatenbank wie folgt abgebildet:

- **Gültigkeitszeitraum**: Von wann bis wann war die Aussage der Information gültig?
- **Bezugszeitpunkt**: Ab wann war das Wissen über die Aussage der Information vorhanden?

3.3.4 Ausprägungen der Information im Hinblick auf die Datenmodellierung

Verschiedene Ausprägungen der Information sind im Hinblick auf die Datenmodellierung von Interesse. Sie beeinflussen sowohl die Datenmodellierung selbst als auch die Beschreibung der Datenmodellierung.

3.3.4.1 Realität und Abstraktion

Das SMIS muss mit seinen Datenstrukturen und den dazugehörenden Regeln denjenigen Teil der Realität abbilden, der für die Geschäfts- und Fachprozesse des Strassenmanagements erforderlich ist. Der **Abbildungsprozess**, der sogenannte Datenentwurf, muss dahingehend optimiert werden, dass die wesentlichen Eigenschaften der Realität erhalten bleiben. Diese Entwurfstätigkeit ist in gleichem Masse sowohl eine kreative Architekturaufgabe als auch eine systematisch strukturierte Ingenieurtätigkeit. Sowohl für deren Durchführung als auch zu deren Dokumentation sind entsprechende Werkzeuge notwendig: **Beschreibungssprachen** (also das Thema dieses Forschungsprojekts).

Dabei müssen zwei Fragestellungen beantwortet werden:

- **Abgrenzung des Abbildungsbereichs**: Was gehört dazu, was nicht?
- **Abstraktion durch die Abbildung**: Jede Abbildung von Realität in ein Datensystem ist mit einem gewissen Informationsverlust behaftet. Beispiele dafür sind: Datenstrukturen können immer nur Teile der Information abbilden, kartographische Generalisierung, Vereinfachung der Zeitaspekte, Vereinfachung der Beziehungen und Regeln.

Die Anworten auf diese Fragen sind für den Nutzen des SMIS von ausschlaggebender Bedeutung. Die Beantwortung (der Datenentwurf) selbst und die entsprechende Dokumentation sind es daher ebenso.

3.3.4.2 Generalisierung und Spezialisierung

Die Abbildung der Realität muss den für viele Fälle typischen Ausprägungen von Informationsobjekten als Generalisierung oder als Spezialisierung anderer Objekte Rechnung tragen können:

- **Generalisierung** bedeutet: Ein Objekt-Typ vereinigt generelle Eigenschaften anderer Objekt-Typen; z.B. gibt es viele Ausprägungen von Verkehrsgrössen, die Anzahl der Ausprägungen ist aber so gross, dass es sinnvoll ist, alle diese Grössen in einem einzigen Informationsobjekt-Typ zu generalisieren.
- **Spezialisierung** bedeutet: Ein Objekt-Typ "erbt" die meisten Eigenschaften eines anderen Objekt-Typen, er besitzt aber noch einige spezifische Eigenschaften, die nur ihn charakterisieren; z.B. besitzt der Endbezugspunkt eines Ax-Segments alle Eigenschaften eines "generellen" Bezugspunkts, trotzdem muss er speziell ausgezeichnet werden, damit die Aussage "Endbezugspunkt" erhalten bleibt.

Der Datenentwurf muss diese Aspekte der Generalisierung und der Spezialisierung abbilden und beschreiben können.

3.3.4.3 Detail und Aggregation

Die Abbildung der Realität muss den Ansprüchen des SMIS bezüglich Detaillierungsgrad der Information, resp. der dazu erforderlichen Daten, Rechnung tragen können.

Dabei müssen zwei Aufgaben gelöst werden:

- Die Fach- und Geschäftsprozesse bestimmen den **Detaillierungsgrad der Information**, resp. der Daten. Dabei können unterschiedliche Aggregationsprozesse derselben Detaildaten das zumindest zeitweise Vorhalten der Detaildaten notwendig machen. Dies heisst automatisch, dass der Daten-Entwurf diesen Detaildaten inhaltlich Rechnung tragen muss.
- Die Dateninhalte und Aggregationsprozesse bestimmen die zum Vorhalten der Daten erforderlichen **Strukturen**. Dabei können gemeinsame (generalisierte) Strukturen für Aggregate und Details, oder aber spezialisierte Strukturen zur Anwendung gelangen.

Gewisse Entwurfsmethoden bieten zudem verschiedene Möglichkeiten, zwischen der Aggregation und der Komposition zu unterscheiden:

- **Aggregation**: Die Details sind unter anderem Teile des Aggregats; die Details können aber auch eine alleinige Existenz haben, ohne dass das Aggregat "lebt". Beispiel: erhobene Verkehrsvolumen (z.B. Stundenwerte) können gut ohne "ihren" DTV leben**. Aggragationen sind oft räumliche oder/und zeitliche Zusammenfassungen von Detaildaten.**
- **Komposition**: Die Details sind obligatorische Teile des Aggregats; sie können alleine nicht "leben". Beispiel: Sektoren sind Teile der Axe, sie bilden in ihrer Gesamtheit die Axe, sie können alleine nicht leben. **Kompositionen bilden sehr eng zusammenhängende Objekt-Typen ab**.

3.4 Organisationsaspekte für das SMIS

3.4.1 Rahmenorganisation (für Nutzung und Betrieb)

Als Rahmenorganisation bezeichnet man aus Informatik-Sicht diejenigen Organisationsaspekte, die für die **Nutzung und den Betrieb** einer Informatiklösung erforderlich sind. Diese betreffen einerseits die Aktivitäten der Geschäfts- und Fachprozesse, die zur Lösung der fachlichen Aufgabe im "geschäftlichen" Umfeld dienen. Andererseits bringt der Einsatz einer Informatiklösung neue Aktivitäten, die direkt aus deren Nutzung und Betrieb resultieren.

3.4.1.1 Beteiligte und Organisationsstruktur

Aus einer vereinfachten Sicht können die Beteiligten für die Nutzung und den Betrieb von SMIS wie folgt beschrieben werden:

- Manager des Strassenmanagements
- SMIS-Benutzer des Strassenbetreibers
- SMIS-Verwalter des Strassenbetriebers
- Strassennutzer (Verkehrsteilnehmer)
- Politik und Gesellschaft
- Medien

Die Vielfalt der Beteiligten und ihrer Interessen und Aufgaben begründet **intensive und komplexe Kommunikationsbedürfnisse**, um die Aufgaben mit Hilfe des SMIS lösen zu können.

3.4.1.2 Geographische Verteilung der Beteiligten

Die Beteiligten in der Rahmenorganisation sind in der Regel geographisch verteilt. Auch dies begründet **intensive und komplexe Kommunikationsbedürfnisse**, um die Aufgaben mit Hilfe des SMIS lösen zu können.

3.4.1.3 Aufgaben und Zuständigkeiten

Die Aufgaben und Zuständigkeiten bei der Nutzung und dem Betrieb des SMIS können wie folgt strukturiert werden:

- **Benutzer von SMIS-Daten** und SMIS-Informationen, die von einem anderen Beteiligten aufbereitet worden sind. Typische Beispiele sind Politik, Medien usw.;
- **Benutzer** von SMIS, die selbst **Abfragen und Auswertungen** starten und durchführen können;
- **Benutzer** von SMIS, die **Daten verwalten** und die Strassendatenbank **pflegen** können.

Dies begründet ebenfalls **intensive und komplexe Kommunikationsbedürfnisse**, um die Aufgaben mit Hilfe des SMIS lösen zu können.

3.4.1.4 Aktivitäten und Abläufe

Die Aktivitäten der Beteiligten sind in Kapitel 3.2: Funktionen des SMIS (Applikations-Sicht) beschrieben.

3.4.2 Projektorganisation (für Realisierung und Einführung)

Die Projektorganisation für die Realisierung und die Einführung des SMIS orientiert sich an den üblichen Anforderungen einer Projektorganisation für ein Informatik-System (siehe z.B. HERMES).

Dies gilt analog auch für Bauprojekte im Strassenbereich!

3.4.2.1 Beteiligte und Organisationsstruktur

Die Beteiligten in der Projektorganisation für die Realisierung und die Einführung sind (grob)

- Planer
- Entwerfer
- Umsetzer
- Einführer
- Betreiber

Die in der Regel sehr grosse Zahl von Beteiligten begründet **vielfältige Kommunikationsbedürfnisse**, um die Projekt-Aufgaben lösen zu können.

3.4.2.2 Geographische Verteilung der Beteiligten

Die Beteiligten in der Projektorganisation sind in der Regel stark geographisch verteilt. Dies ruft **vermehrte Kommunikationsbedürfnisse** hervor

3.4.2.3 Aufgaben und Zuständigkeiten

Die Aufgaben der Projektorganisation können grob wie folgt zusammengefasst werden:

- **Systemgestaltung**: Die Aufgaben des Entwurfs und der Umsetzung sind beschrieben in Kapitel 4.5 Die Verwendung der Entwurfsebenen in den SMIS-Phasen.
- **Projektorganisations-Struktur**
- **Abläufe und Termine** in der Projektorganisation
- **Umfeld** des Projekts
- **Rechts- und Vertragsaspekte**
- **Kostenmanagement und Finanzierung**
- **Technologische Aspekte** der Projektentwicklung
- **Information und Kommunikation** (menschliche Aspekte in der Projektorganisation)
- **Projektadministration und -dokumentation**

Alle diese Aufgaben begründen **intensive und komplexe Kommunikationsbedürfnisse**.

3.4.2.4 Aktivitäten und Abläufe für Entwurf und Umsetzung

Siehe Kapitel 4.3: Phasen für die zyklische Entwicklung von SMIS

3.5 Technologische Aspekte für SMIS (EDV)

3.5.1 Software-Entwicklungsumgebung (SEU):

Die SEU beeinflusst mit ihren Möglichkeiten sowohl den Entwurf als auch die Umsetzung des SMIS. Zur SEU gehören zum Beispiel:

- Programmiersprachen
- CASE-Werkzeuge
- GUI-Builder (zur Realisierung von benutzerfreundlichen Bildschirmoberflächen)
- Report-Writer
- Grafik-Werkzeuge zur grafischen Darstellung von Strassendaten
- Geo-Viewer zur geographischen Darstellung von Strassendaten
- usw.

3.5.2 Software-Betriebsplattform (SBP):

Die Betriebsplattform stellt der Software die grundlegenden HW- und SW-Funktionalitäten zur Verfügung. Dazu gehören:

- Betriebssysteme
- Middleware (Software zwischen dem Betriebssystem und weiterer Software, in der Regel bei verteilten SW-Architekturen)
- Datenbanksoftware (DBMS)
- usw.

3.5.3 Sachmittel, inkl. HW-Betriebsplattform

Als Sachmittel dienen

- EDV-Systeme
- Telekommunikationssysteme
- Betriebsmittel
- Räume
- usw.

* * *

4 Der Lebenszyklus des SMIS

4.1 Übersicht

Das SMIS, so wie jedes andere technische System (z.B. ein Bauwerk) auch, durchläuft im Laufe seines "Lebens" verschieden **Phasen**. Es wird aus einem Bedürfnis heraus geplant, entworfen, umgesetzt, eingeführt, genutzt und betrieben und am Ende seines Lebens wieder ausser Betrieb gesetzt.

Dies gilt analog auch für Bauprojekte im Strassenbereich!

Dieser Lebenszyklus kann auf unterschiedliche Weise gestaltet werden. Zudem kommen im Laufe dieses Lebenszyklus unterschiedliche Methoden zur **Steuerung und Beherrschung** der Phasen zum Tragen.

Die **Modellierungs-Aspekte** sind stark mit diesem Lebenszyklus verbunden. Darum sollen im Folgenden einige relevante Aspekte des Lebenszyklus des SMIS dargestellt werden.

4.2 Phasenmodelle für den Lebenszyklus eines Informationssystems

Der Detail-Ablauf der einzelnen Lebensphasen ist einerseits an das betrachtete technische System, andererseits an die in den Phasen verwendeten Methoden und Werkzeuge anzupassen. In der Informatik, d.h. auch für das SMIS, können unterschiedliche Phasenmodelle angewandt werden. Da sie einen Einfluss auf die Datenmodellierung haben, sollen sie kurz dargestellt werden. Die Details zu einzelnen Phasenmodellen finden sich in den Anhängen.

4.2.1 Wasserfall-Modelle

Wasserfallmodelle sind Phasenmodelle, in denen jede Stufe (Phase) nur einmal durchlaufen wird. Es sind also rein sequenzielle Phasenmodelle.

Die folgende Figur soll diesen Sachverhalt verdeutlichen. (Phasenmodelle für Bauprojekte sind in der Regel ebenfalls "Wasserfall-Modelle", d.h. auch dort wird jede Phase nur einmal durchlaufen.)

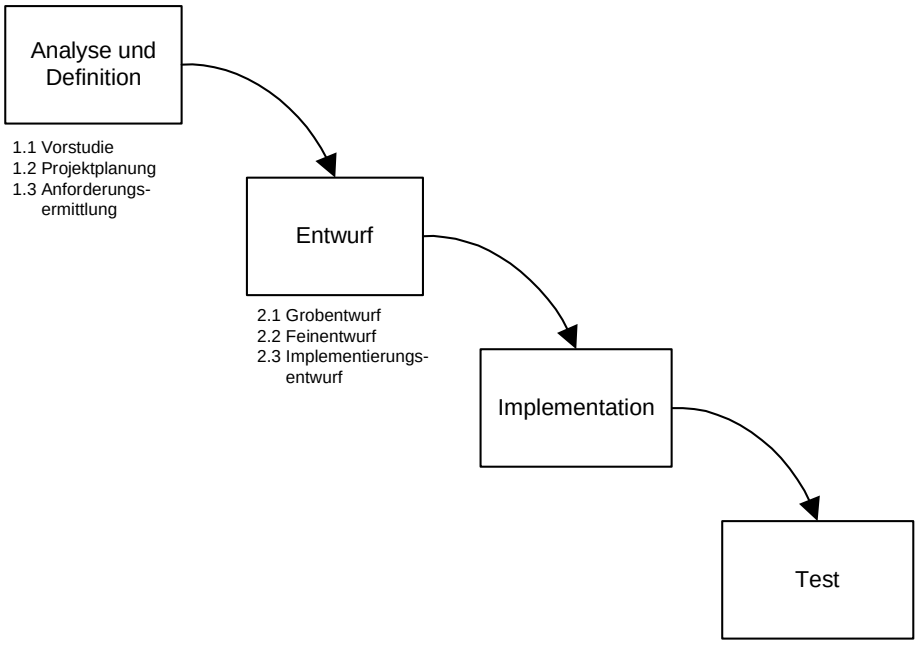


Abbildung 4-1: Beispiel eines klassische Wasserfallmodells [Pagel & Six, 1994]

Ein wesentlicher Nachteil der Wasserfallmodelle ist, dass man nicht zurückschreiten kann. Ist die Analyse- und Definitionsphase beispielsweise abgeschlossen, so kann sie nicht mehr aufgegriffen werden.

Im Folgenden werden einige Phasenmodelle gemäss dem "Wasserfall-Modell" kurz beschrieben.

4.2.1.1 HERMES86

HERMES ist das "**H**andbuch der **E**lektronischen **R**echenzentren des Bundes, **E**ine **M**ethode für die **E**ntwicklung von **S**ystemen" (Handbuch der Systementwicklung in den schweizerischen Bundesrechenzentren). Die Ausgabe 1986 stützt sich auf ein sequentielles "Wasserfall"-Phasenmodell ab, obwohl dann die Erarbeitung der "Rahmenorganisation" (Betriebsorganisation) parallel zur Programmierung angenommen wird. Die Phasen sind:

- Voranalyse: Situationsanalyse, Zielsetzungen, Lösungsansätze
- Konzept: Pflichtenheft für Daten, Funktionen und Technik
- Detailspezifikation: Programmiervorgaben in der gewählten Software-Entwicklungs-Umgebung
- Programmierung: eigentliche Systementwicklung und Tests
- Rahmenorganisation: Betriebsorganisation für die zu entwickelnde Software
- Einführung: Dokumentation, Einführung, Schulung usw.

Siehe dazu den Anhang I "HERMES86".

4.2.1.2 SSADM

Die SSADM oder "Structured Systems Analysis and Design" ist eine Methodik, die von der CCTA (Central Communication and Telecommunication Agency), einer Agentur der britischen Regierung, eingeführt wurde. Sie wird als Wasserfallmodell eingestuft, weil sie die Entwicklung eines IT-Projekts hauptsächlich als linearen Prozess beschreibt.

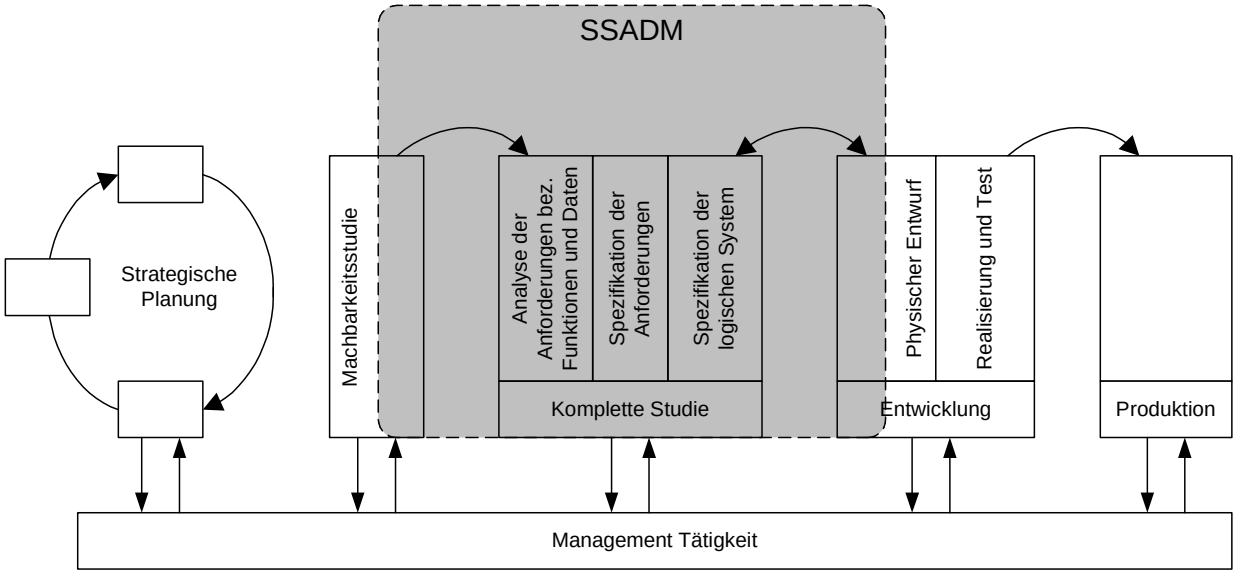


Abbildung 4-2: Positionierung von SSADM innerhalb des Lebenszyklus des Informationssystems

Die SSADM beschreibt vor allem den konzeptionellen und den logischen Entwurf innerhalb des Lebenszyklus des Informationssystems. Das Ziel des SSADM beinhaltet nicht die strategische Planung, die Realisierung und die Produktion.

SSADM bezeichnet folgende Etappen im Entwurf eines Informationssystems:

1. Machbarkeitsstudie
2. Analyse der Anforderungen bezüglich Funktionen und Daten
   2.1 Studie über bestehendes Umfelds
   2.2 Optionen des Fachsystems (business system)
3. Spezifikationen der Anforderungen
   3.1 Optionen des technischen Systems
   3.2 Logisches Design
4. Spezifikation der logischen Systeme
5. Physischer Entwurf

4.2.1.3 Die CASE*Method von ORACLE

Die CASE*Method wurde von der Firma Oracle eingeführt, um den Entwurf eines Informationssystems zu erleichtern und die nötige Software zu liefern. CASE*Method ist eine strukturierte Methode um Systeme zu entwickeln, die den Anforderungen der Benutzer entsprechen. Die CASE*Method basiert auf SSDAM und wurde von Oracle-UK entwickelt.

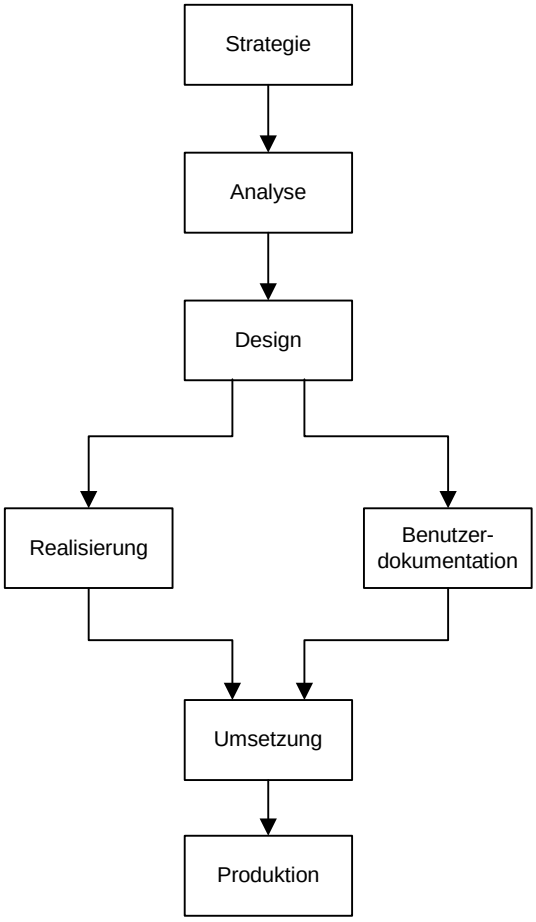


Abbildung 4-3: Die CASE*Method (ORACLE)

Die CASE*Method unterstreicht die Nutzung von unterschiedlichen standardisierten Diagrammen, die eine gute Kommunikation zwischen den Akteuren innerhalb der Systementwicklung ermöglichen. Missverständnisse und Fehler müssen so schnell wie möglich korrigiert werden, um nicht unnötige Kosten entstehen zu lassen. Das CASE*Method strukturiert das ganze Verfahren in kurzzeitige Aktivitäten, was die Planung des ganzen Entwurfs erleichtert.

4.2.1.4 James Martin

In der Terminologie von James Martin wird mit Etappe dasselbe bezeichnet wie der in diesem Kapitel benutzte Begriff "Phase". Gemäss James Martin, besteht der Entwurf eines Informationsssystems aus sieben Etappen:

- **Planungsetappe der Informationsstrategie,** in der die Planer eine generelle Sicht auf die Informationsbedürfnisse des Unternehmens erstellen;
- **Analyse-Etappe für einen bestimmten Tätigkeitsbereich,** in der die Analytiker ein bestimmtes Segment, d.h. einen Tätigkeitsbereich, analysieren;
- **Etappe des funktionalen Entwurfs der Applikation,** in der Entwerfer eine Applikation innerhalb eines Tätigkeitsbereichs spezifizieren;
- **Etappe des technischen Entwurfs,** in der die Entwerfer die Ergebnisse des funktionalen Entwurfs an eine bestimme IT-Umgebung anpassen;
- **Konstruktionsetappe,** in der Entwickler ausführbare Komponenten der Applikation entwickeln.
- **Einführungsetappe,** in der die neue Applikation in einer Produktionsumgebung in Betrieb genommen wird;
- **Produktionsetappe,** in der das Unternehmen die neue Applikation nutzt.

4.2.1.5 MERISE

Die Merise Methode wurde 1978 und 1979 im Rahmen der "Mission à L‘informatique" in Paris definiert. Die Grunddokumentation zu Merise wurde von Hubert Tardieu, Arnold Rochfeld und René Colletti zwischen 1985 und 1989 erstellt. Sie umfasst die Etappen:

- **Richtplan,** die strategischen Ziele der Organisation mit ihren Informationsbedürfnissen;
- **Vorstudie,** eine detaillierte Beschreibung einer Teilmenge des im Richtplan untersuchten Bereichs auf konzeptueller Ebene und die Beschreibung einiger Elemente auf logischer Ebene;
- **Detaillierte Studie,** die funktionalen, detaillierten Spezifikationen eines Systems;
- **Technische Studie,** die Einzelheiten der Architektur der Applikationen, die Organisation der Daten sowie die Funktionen;
- **Produktion,** die eigentliche SW-Entwicklung;
- **Einführung,** die Inbetriebnahme des neuen Systems, die Ausbildung der Benutzer und die Anwendung neuer Organisationsstrukturen;
- **Unterhalt,** Erweiterungen der betriebenen Applikationen, um neue Bedürfnisse der Benutzer und den technischen Fortschritt einzubinden.

Später wurden auch andere Modelle als das reine "Wasserfall"-Modell in Merise benutzt, wie bespielsweise die zyklischen Modelle oder das V-Modell.

4.2.2 Das Vorgehens-Modell (V-Modell)

Das deutsche Vorgehensmodell oder auch V-Modell genannt ist eine Weiterentwicklung des Wasserfallmodells, dass von Bröhl und Dröschel 1995 für das Bundesministerium für Verteidigung entwickelt wurde. Unterdessen findet dieses Modell auch ausserhalb der Bundesministerien seine Anwendung.

Es durchläuft aber während dem Phasendurchlauf eine zweite Dimension (der Abstraktion, siehe Kapitel 4.5 "Die Verwendung der Entwurfsebenen in den SMIS-Phasen"): einmal vorwärts zu immer höherer Abstraktion, dann wieder zurück zum Ursprung, zur Realität. HERMES95 ist übrigens eng an das deutsche V-Modell angelehnt (siehe 4.2.2.1).

Das V-Modell umfasst vier Submodelle: Softwareerstellung, Qualitätssicherung, Projektmanagement und Konfigurationsmanagement.

In diesem Kapitel wird das Schwergewicht auf die Softwareentwicklung (SE) gelegt. Das Lebensphasenmodell wird in folgendem Schema dargestellt.

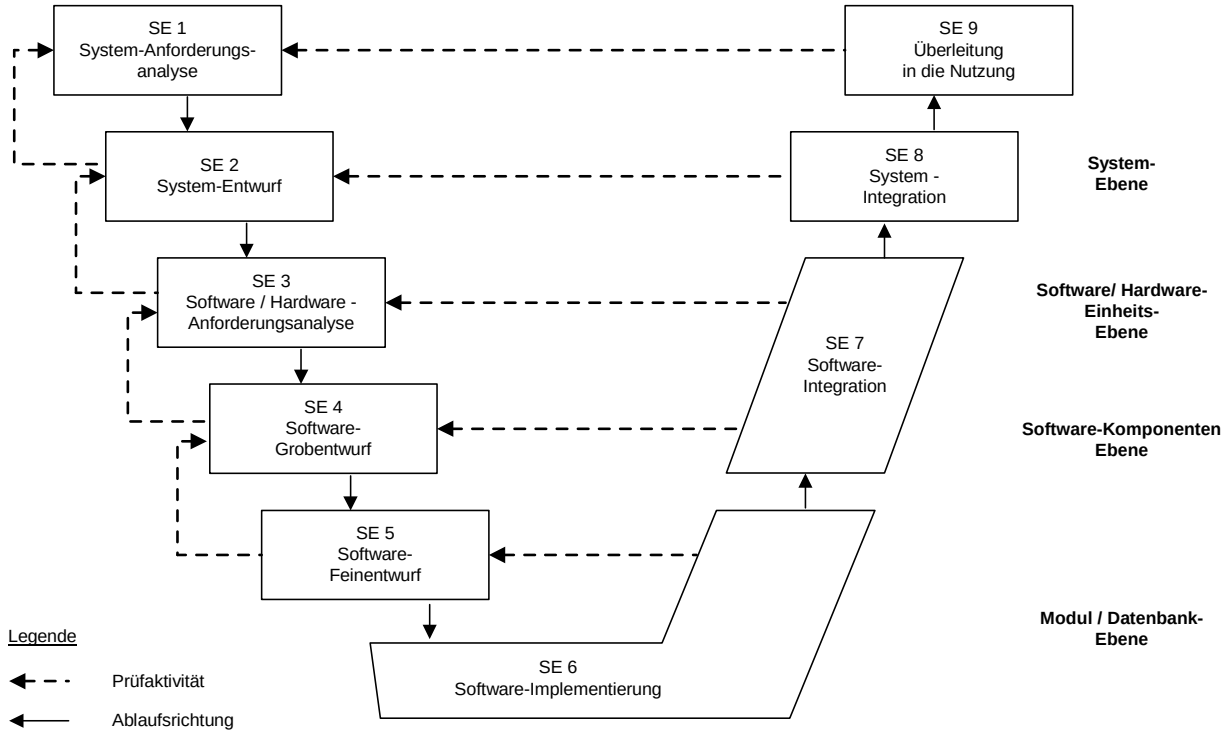


Abbildung 4-4: Das vereinfachte V-Modell (Submodell Softwareerstellung)
[ursprünglich aus Bröhl & Dröschel, 1995]

In dieser Darstellung lassen sich grundsätzlich 3 Bereiche erkennen:

- Entwurf (SE1–SE5): Ausgehend aus einer Geschäfts- oder Fachaktivität werden Systemkomponenten entworfen.
- Realisierung (SE6): Die Systemkomponenten werden umgesetzt (Codierung).
- Integration (SE7-SE9): Die einzelnen Komponenten werden in ein System integriert.

4.2.2.1 HERMES-95

HERMES-95 wurde als offener Standard durch das (schweizerische) Bundesamt für Informatik erarbeitet und ist mit internationalen Normen und Standards abgestimmt. Die Nutzung ist frei und unentgeltlich (Siehe Anhang)

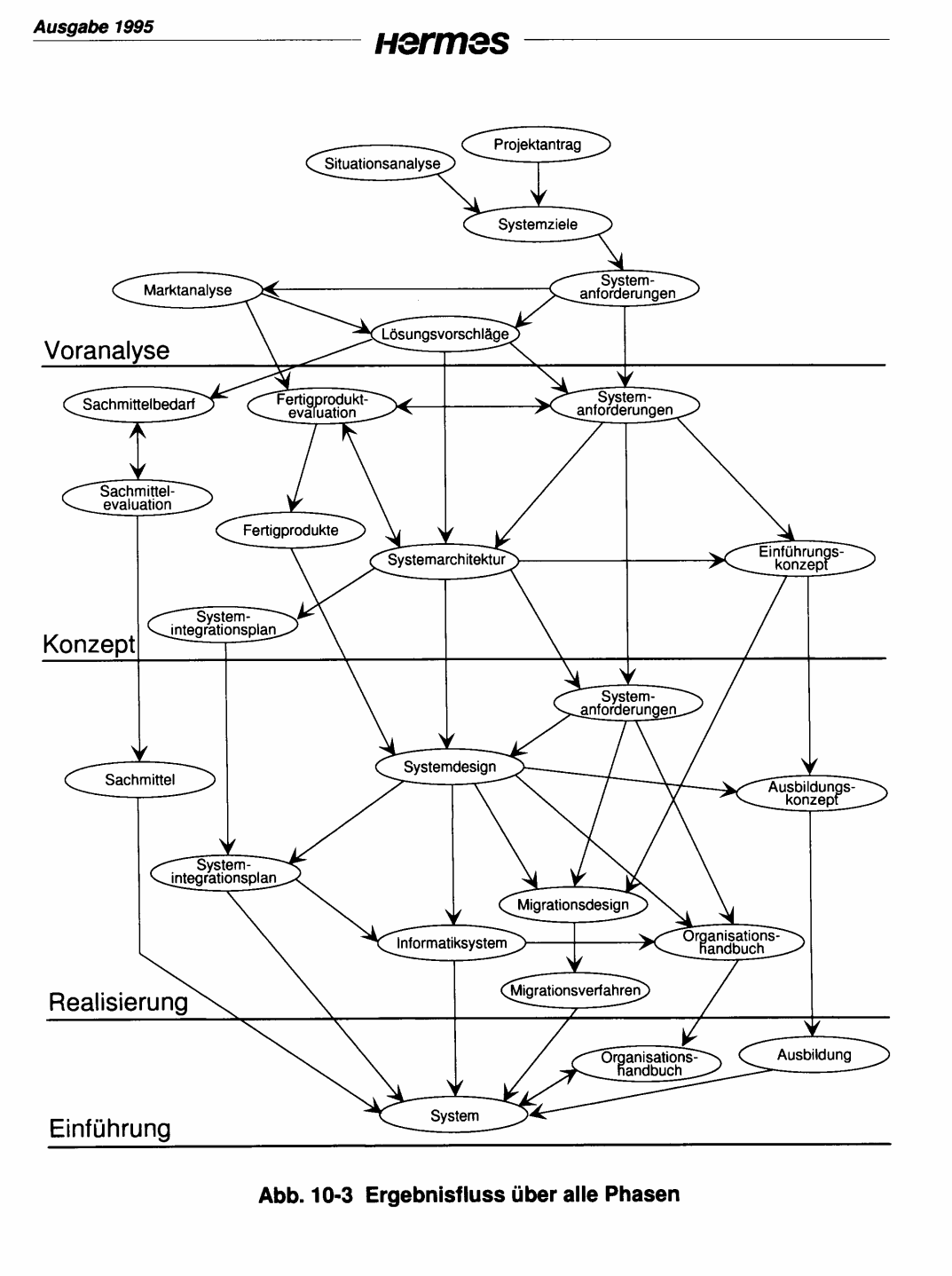


Abb. 10-3 Ergebnisfluss über alle Phasen

Abbildung 4-5: Das Ergebnis-Flussdiagramm von HERMES-95

4.2.3 Zyklische Phasenmodelle

Alle "Wasserfallmodelle" haben ein fundamentales **Problem**: die Gefahr, dass der umgesetzte Entwurf nicht den Erwartungen der Benutzer entspricht, ist sehr gross. Dabei hilft auch der so oft gepriesene frühe Einbezug des Benutzers nichts oder nur ungenügend. Der Grund dafür ist, dass der Benutzer, wenn er beigezogen wird, noch sehr weit von der umgesetzten Lösung entfernt ist.

Dies gilt analog auch für Bauprojekte!

Ein Ansatz zur Korrektur dieser Situation ist das **Prototyping**, von dem es unterschiedliche Arten gibt. Allen ist gemein, dass laufend kleinere Systemteile entwickelt werden, die dem Benutzer gezeigt werden können, die er beurteilen kann und die dann wieder zurück in die Umsetzung oder fallweise auch schon zum Benutzer gehen können. Der an den Benutzer abgegebene Prototyp wird Pilotapplikation genannt und muss nur eingeschränkten Qualitätsanforderungen genügen.

Diese zuerst eher "zufällige" Entwicklung wurde dann zielgerichtet zu **zyklischen Phasenmodellen** weiterentwickelt. Im nächsten Kapitel sollen die wesentlichen Grundzüge am "(zyklischen) Phasenmodell für das SMIS" erläutert werden.

4.2.4 Das (zyklische) Phasenmodell für das SMIS

Für das SMIS soll ein **zyklisches Modell** gewählt werden. Diese Entwicklungsmethode erlaubt beide Paradigmen des Entwurfs und der Umsetzung (siehe Kapitel 6: Entwurfs- und Umsetzungs-Paradigmen).

Das Phasenmodell für SMIS wird detailliert im nächsten Kapitel beschrieben. Es besteht aus den vier Hauptphasen

- **Planung** (engl. analysis), im Sinne der Definition der Strategie
- **Entwurf** (engl. design), im Sinne der Definition der Taktik
- **Umsetzung** (engl. implementation), im Sinne der EDV-technischen Realisierung
- **Nutzung** und Betrieb, im Sinne der operationellen Aspekte und der Rückkoppelung

Diese vier Hauptphasen können analog zu den vier Quadranten eines kartesischen Koordinatensystems aufgefasst und zyklisch durchlaufen werden.

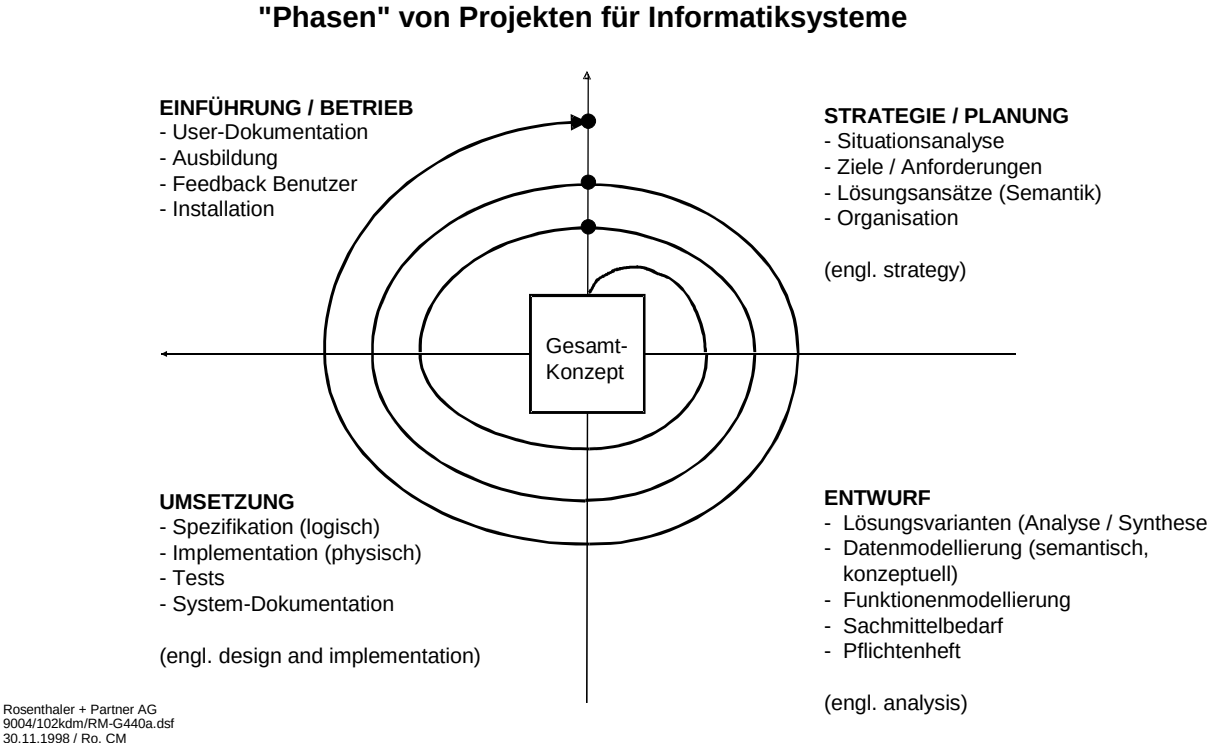


Abbildung 4-6: Das zyklische Phasenmodell

Die vier Hauptphasen werden in den folgenden Kapiteln weiter aufgegliedert und deren Inhalte beschrieben.

4.3 Phasen für die zyklische Entwicklung von SMIS

Die folgenden Phasen werden in dieser Reihenfolge immer wieder durchlaufen. Nach jedem Durchlauf ist wieder ein nächstes Teilstück des Systems eingeführt und nutzbar. Dieses zyklische Phasenmodell erlaubt es, kleine Einheiten zu entwickeln und den Benutzer in einem sehr hohen Masse in diese Entwicklung einzubeziehen ("feahere oriented developement").

Die **Kommunikationsbedürfnisse**, die aus den einzelnen Phasen der Entwicklung entstehen, werden im Kapitel 5: "Kommunikationsbedürfnisse im Lebenszyklus des SMIS" beschrieben. Sie sind vielfältig und haben eine beträchtliche Wechselwirkung mit dem Entwurf und seiner Dokumentation!

4.3.1 Planung (Strategie)

4.3.1.1 Situationsanalyse

Die Anforderungsanalyse untersucht die Bedürfnisse des Benutzers in seinen **Geschäfts- und Fachprozessen**. Sie stellt Stärken und Schwachpunkte zusammen und bewertet sie. Sie leitet daraus die Anforderungen an das System ab. Die Resultate werden dokumentiert.

4.3.1.2 Zielsetzung

Aus der Situationsanalyse kann das **Zielsystem** mit seiner Zielstruktur und den entsprechenden Anforderungen abgeleitet werden. Die einzelnen Ziele, deren Bewertung und Gewichtung (Muss-Ziele, Soll-Ziele) und deren Struktur werden erarbeitet und dokumentiert.

4.3.1.3 Lösungsansätze (strategische Planung)

Aus dem Zielsystem kann die (strategische) Planung abgeleitet werden. Sie beschreibt das **zukünftige System** mit einer sehr benutzerbezogenen Sprache, d.h. auf einer sogenannten "semantischen Ebene".

Dies gilt analog auch für Bauprojekte im Strassenbereich! => Planungsstudie!

In einer Planungsstudie für ein Bauprojekt werden ebenfalls die gleichen Schritte durchlaufen. Auch die strategische Zielsetzung der zu durchlaufenden Schritte ist identisch.

Die Dokumentation der Anforderungsanalyse und der Planung hat im Bauprojekt den genau gleichen Ansprüchen zu genügen: Sie muss vom (zukünftigen) Benutzer verstanden werden können, und sie muss seine Anforderungen enthalten.

4.3.2 Entwurf (Taktik)

4.3.2.1 Konzeptueller Systementwurf

In der Entwurfsphase wird die Planung in einer formalen Sprache in einen Entwurf umgesetzt, der alle **konzeptionell relevanten Elemente des Systems** enthält. Dieser Entwurf ist aber Umsetzungs-unabhängig! Es wird das WAS? der Objekte, Funktionen und Organisation beschrieben, das WIE? interessiert in dieser Phase noch nicht.

Dies gilt analog auch für Bauprojekte im Strassenbereich! => Vorprojekt und Bauprojekt!

Im Vorprojekt und im Bauprojekt für ein Bauwerk werden ähnliche Schritte durchlaufen wie beim Entwurf eines Informationssystems. Auch die Zielsetzung der zu durchlaufenden Schritte ist identisch.

Die Dokumentation des Entwurfs hat im Bauprojekt den genau gleichen Ansprüchen zu genügen: Sie muss sowohl vom (geschulten) Benutzer als auch vom später Ausführenden verstanden werden können. Das heisst, sie muss sowohl die Benutzeranforderungen detailliert darstellen, als auch die Anforderungen an die Ausführung enthalten.

In der Informatik sind heute insbesondere zwei Entwurfs- (und Umsetzungs-) Paradigmen bekannt:

- **Strukturierte Modellierung**
- **Objektorientierte Modellierung**

Im Kapitel 6: Entwurfs- und Umsetzungs-Paradigmen werden diese Paradigmen beschrieben, da sie für die Modellierungsaspekte des Entwurfs von grosser Bedeutung sind.

4.3.3 Umsetzung (EDV, Technik)

4.3.3.1 Spezifikation

Die Resultate der Entwurfsphase müssen in der Folge in eine **Informatik-, resp. EDV-Sprache** umgesetzt werden: die (**Detail-) Spezifikation**. Dazu dienen die Software-Entwicklungsumgebungen SEU des Kapitels 3.5.1. Der gleiche Entwurf kann je nach SEU in die entsprechenden EDV-Produkte, resp. Informatiksysteme umgesetzt werden!

4.3.3.2 Programmierung/Entwicklung (Implementation)

Nach dem Spezifizieren wird das Resultat dann in ein Softwareprodukt umgesetzt.

Dieser Prozess kann **Programmieren** bedeuten, wenn die SEU mit Programmiersprachen arbeitet. Dieser Prozess kann auch **Generieren** bedeuten, wenn die Spezifikation mit CASE-Werkzeugen, z.B. ORACLE-Designer, erfolgt ist.

4.3.3.3 Testen

Als letzte, aber äusserst wichtige Phase der Umsetzung muss der umgesetzte Systemteil für sich (Modultests und Kettentests) und im Zusammenspiel mit anderen Systemteilen (Alpha-Test) getestet werden.

Erst nach erfolgreichem Bestehen der Tests darf es an den Benutzer weitergegeben werden. (Dieser testet dann noch einmal: Beta-Test)

Dies gilt analog auch für Bauprojekte im Strassenbereich!

Auch die Umsetzung eines Strassenprojektes erfordert Ausführungsunterlagen (Spezifikationen), die eigentliche bauliche Ausführung und die Kontrolle der erreichten Qualität des Bauwerks (Tests).

4.3.4 Nutzung und Betrieb (Operation)

4.3.4.1 Einführung

Der umgesetzte Systemteil muss mit dem Benutzer eingeführt werden. Dazu gehören Ausbildung und Übungen sowohl der **Anwendung** (welche applikatorische Funktion für welche Aufgabe?) als auch der **Bedienung** (welche technische Funktion für welchen Schritt der Anwendung?).

Zur Einführung und laufenden Begleitung gehört auch der Betrieb einer "**Supportstelle**", die aus Erfahrung vor allem fachliche und applikatorische Fragen der Anwendung und nicht technische Fragen der Bedienung zu beantworten hat.

4.3.4.2 Betrieb

Die umgesetzten Systemteile müssen betrieben werden. Dazu gehören Tätigkeiten wie Datensicherung, Datenkontrolle (der Datenqualität), Datenbankadministration, DB-Optimierung, Installation, Release-Wechsel usw.

4.3.4.3 Wartung

Die betriebenen Systemteile müssen gewartet werden. Dazu gehören in der Regel:

- Die **technische Wartung**: Optimierung und Anpassung des Zusammenspiels des Systems mit der SW- und der HW-Plattform und deren Veränderungen. Beispiele sind Wechsel des Betriebssystem-Release oder des Betriebssystems selbst, Wechsel von Telekommunikations-SW, usw.
- Die **applikatorische Wartung**: Dazu gehören Fehlerkorrekturen und kleine Anpassungen, die den Nutzungsgrad oder die Funktionalität verbessern, aber die so klein sind, dass sie nicht wie ein neues System-Teil betrachtet und entsprechend geplant, entworfen, umgesetzt und eingeführt werden.

Wartung kann **präventiv oder korrektiv** gemacht werden. In der Praxis wird eine Mischung festgestellt, da nie alle Problemfälle vorausgesehen werden können.

4.3.4.4 Feedback

Die Systemnutzung bringt **Erfahrungen**, die für die weitere Systementwicklung von grossem Nutzen sind. Diese gilt es zu verwerten. Der Benutzer-Feedback ist bewusst zu erheben und zu berücksichtigen.

Dies gilt analog auch für Bauprojekte im Strassenbereich!

Die fertiggestellte Strassenverkehrsanlage muss dem Betrieb übergeben werden (Einführung), sie muss betrieben werden, sie muss unterhalten werden (Wartung) und der Feedback der Betreiber und Benutzer auf die neu zu realisierenden Arbeiten an Strassenverkehrsanlagen sind ebenfalls zu berücksichtigen.

4.4 Entwurfsebenen

Die Methoden für den Entwurf und die Umsetzung gehen alle von zwei klaren Orientierung aus:

- Die Orientierung vom Allgemeinen zum Besonderen, vom Groben zum Feinen, von den Kernaspekten zu den Details (top-down).
- Die Orientierung von den (konzeptionellen) Benutzeraspekten zu deren technischer Umsetzung (EDV).

Die beiden Orientierungen zusammen begründen die sogenannten Entwurfs-Ebenen.

Dies gilt analog auch für Bauprojekte im Strassenbereich!

Der Entwurf einer Strassenverkehrsanlage geschieht ebenfalls mit zunehmender Abstraktion:.

Zu Beginn werden mit dem Benutzer die Anforderungen definiert. Dies geschieht an Hand von Skizzen und Zeichnungen, die die neue Situation z.B. dreidimensional-perspektivisch oder in anderen, für den Benutzer leicht verständlichen Darstellungen aufzeigt. Vielleicht werden auch Modelle angefertigt.

Die mit dem Benutzer gewonnen Erkenntnisse werden in eine standardisierte "Sprache", in der Regel mit Plänen, Beschrieben und Mengengerüsten, übersetzt.

Diese Informationen werden in ausführungsspezifische Daten umgesetzt: Ausführungspläne usw.

Dann kann wirklich umgesetzt, d.h. gebaut werden.

Die Abbildung 4-7 stellt die Entwurfsebenen in den Zusammenhang mit den drei Hauptthemen beim Systementwurf:

- die Daten
- die Funktionen
- die Organisation

Die Bedeutung der Daten beim Entwurf ist aus verschiedenen Gründen wesentlich grösser als diejenige der Funktionen oder der Organisation. Dies kann wie folgt begründet werden:

Wie in den Kapiteln 2 und 3 dargelegt, stellt das Datenmodell (dargestellt durch ein Datenschema siehe Kap. 7.2) den "langlebigen" Teil eines Informationssystems dar. Auf die Modellierung dieses Abbildes der realen Welt ist also sowohl aus Gründen der Nutzung als auch der Wirtschaftlichkeit das Hauptgewicht zu legen.

Bei den Funktionen sind zwei unterschiedliche Ausprägungen zu beachten. Die sehr "Daten-nahen" Funktionen, wie Integritätsregeln, Historisierung usw., sind in der Regel ebenfalls sehr langlebig. Die auf den Daten aufsetzenden Auswertungen sind jedoch viel weniger langlebig. Sie ändern laufend und werden den Geschäfts- und Fachprozessen angepasst.

Bei der Organisation sind ebenfalls zwei Ausprägungen zu beachten: Die "Daten-nahen" Abläufe und organisatorischen Regelungen sind in der Regel relativ langlebig. Die betriebliche "Unternehmens-Organisation" ist hingegen der kurzlebigste Teil des Informationssystems. Sowohl Aufbau- als auch Ablauforganisation können sehr rasch ändern. Das System sollte also möglichst von solchen Aspekten unabhängig sein!

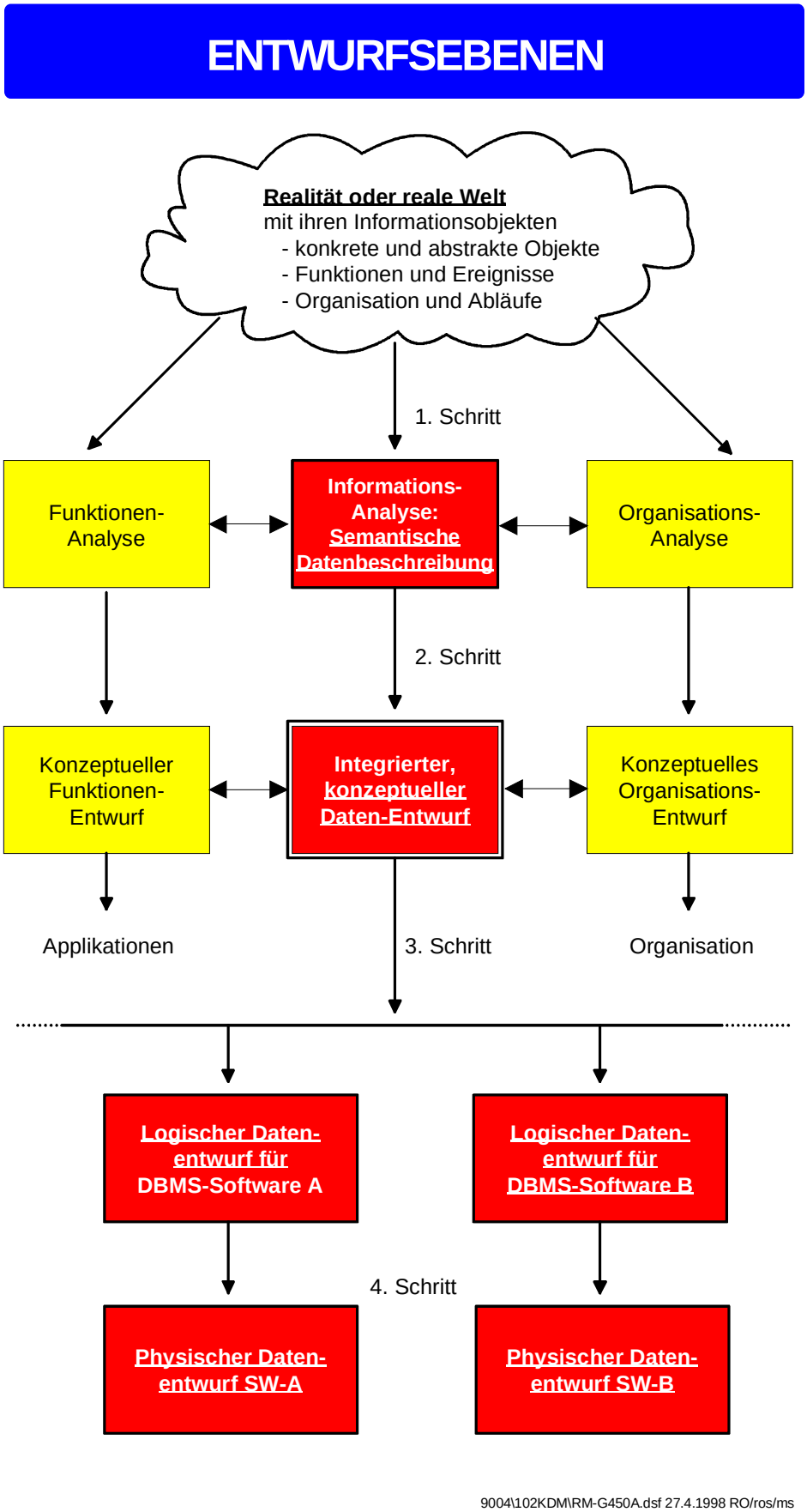


Abbildung 4-7: Die Entwurfsebenen eines Informationssystems

4.4.1 Realität

Die Realität ist die Ausgangslage. Sie beinhaltet die **Geschäfts- und Fachprozesse**. Sie unterliegt keiner Abstraktion. In ihr wird das zu erschaffende Informationssystem existieren und einen Teil dieser Realität abbilden.

4.4.2 Semantischer Entwurf (externe Fachsichten)

Der semantische Entwurf ist das **Resultat der Informations-, Funktions- und Organisations-Analyse**. Er wird im wesentlichen durch Interviews mit dem Benutzer und durch eigene Kenntnisse des Analytikers im Geschäfts- oder im Fachbereich erarbeitet.

Der semantische Entwurf muss unter anderem die Auswahl der interessierenden Realität umfassen ("the universe of discourse"). Zudem müssen die interessierenden Objekte und deren Eigenschaften und Merkmale analysiert werden. Ein ganz wesentlicher Teil des semantischen Entwurfs ist die Analyse und präzise Beschreibung der Begriffswelt des neuen Systems (eben die Semantik).

Der semantische Entwurf zeigt nur das Wesentliche. Er abstrahiert erst wenig. Er ist stark fach-orientiert.

Der semantische Entwurf muss in einer jedem **fachkundigen und interessierten Benutzer** verständlichen Form beschrieben und dokumentiert werden! Dies hat grossen Einfluss auf die Art und Weise der Beschreibung, resp. der entsprechenden Beschreibungssprache.

4.4.3 Konzeptueller Entwurf (interne Fachsicht)

Der konzeptuelle Entwurf ist das Resultat einer **konzeptionellen Arbeit des Entwerfers**. Er wird von diesem, in Rücksprache mit dem Benutzer und durch Validations-Workshops mit diesem Benutzer erarbeitet.

Der konzeptuelle Entwurf zeigt alles Wesentliche für die Modellierung der Daten, Funktionen und der Organisation. Er abstrahiert relativ stark. Er ist **Fach- und Informatik-orientiert**. Er ist aber absolut von der EDV-Umsetzung unabhängig.

Der konzeptuelle Entwurf besteht im Wesentlichen aus den folgenden Schritten:

- Systematisch strukturierte Beschreibung der Daten und der Regeln
- Normalisierung (bei Verwendung des relationalen Modells zur konzeptuellen Modellierung, siehe Tabelle auf der nächsten Seite)
- Vervollständigung, wobei rein semantische Vervollständigungen (in der Informationsanalyse "vergessen") über einen "Rücksprung" in die sematische Ebene zu erfolgen haben. Es gibt aber auch Konzepte, z.B. der Datenverteilung, die erst hier abgehandelt werden.

Der konzeptuelle Entwurf muss in einer systematisch strukturierten Form beschrieben werden, die sowohl von einem konzeptionell denkenden (und die Beschreibungssprache verstehenden) **Entwurfs-Benutzer**, z.B. dem Strassendatenbank-Leiter, als auch von einem Informatiker, der dann umsetzen muss, verstanden werden kann.

Der konzeptuelle Entwurf bildet also die Brücke zwischen der Semantik (als benutzernahe Beschreibung der Realität) und der Informatik, also zwischen dem Benutzer, resp. dessen Strassendatenbank-Leiter und dem Informatiker!

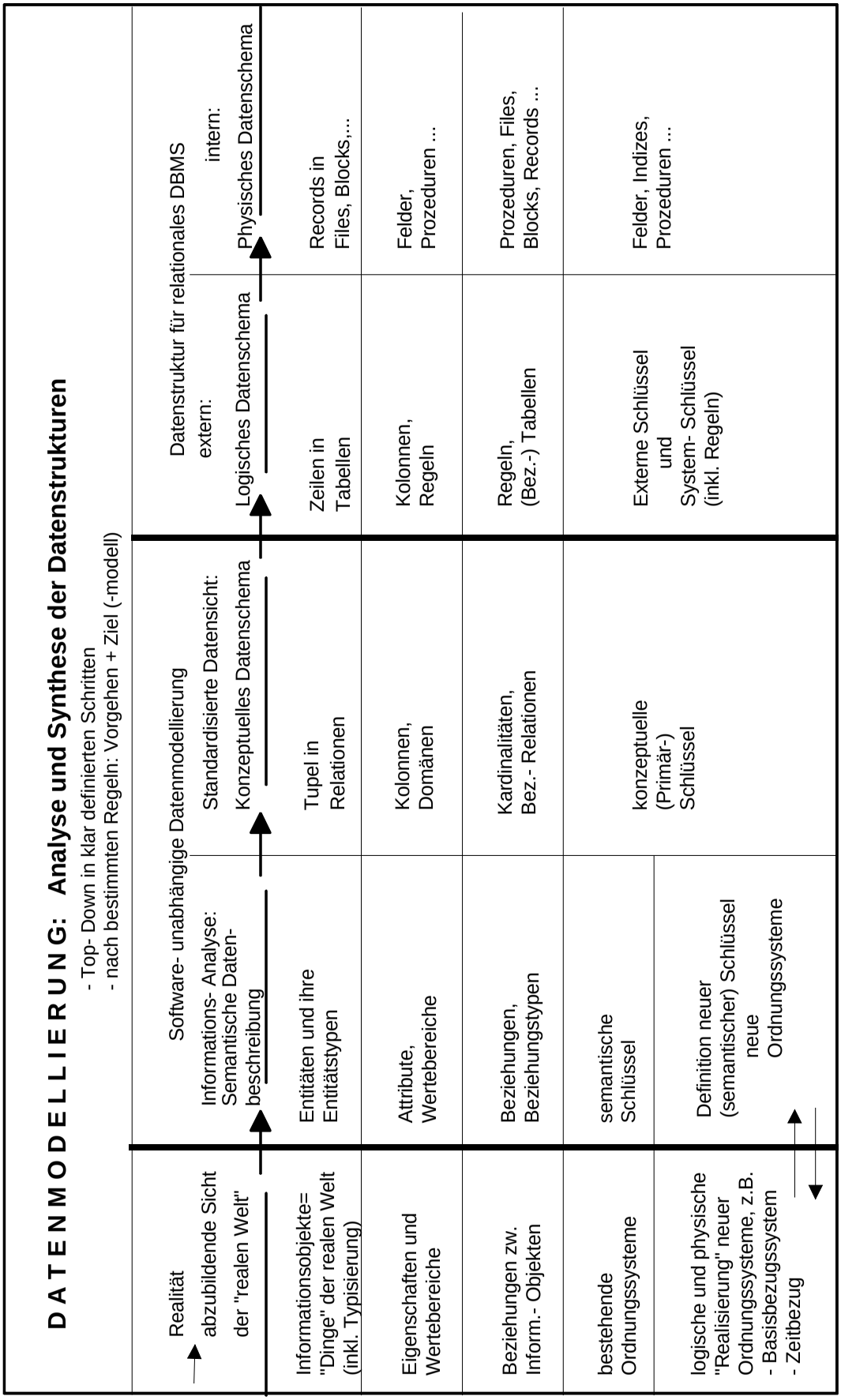


Abbildung 4-8: Die benutzten Begriffe in den unterschiedlichen Entwurfsebenen

4.4.4 Logische Umsetzung (interne Informatik-Sicht)

Die logische Umsetzung setzt den konzeptuellen Entwurf in eine **EDV-Sprache** der System-Entwicklungs-Umgebung SEU um.

Der logische Entwurf besteht im Wesentlichen aus den folgenden Schritten:

- Codieren in die "Programmier-Sprache" der SEU, z.B. SQL, JAVA usw.
- Denormalisieren des Datenmodells (für relationale Schemata), d.h. im Wesentlichen das Einbauen von Redundanzen zur Steigerung der Performance des fertigen Systems oder für technische Zwecke

Die Terminologie beim Übergang von der konzeptuellen zur logischen Ebene ist auf der Tabelle der vorhergehenden Seite dargestellt.

4.4.5 Physische Umsetzung (interne EDV-Sicht)

Die physische Umsetzung setzt den logischen Entwurf, z.B. den Applikations-Quellcode, die DB-Beschreibung oder die CASE-Inhalte, in ausführbare Programme und betreibbare Datenbanken um.

4.5 Die Verwendung der Entwurfsebenen in den SMIS-Phasen

Aus den Phasen der zyklischen Entwicklung des SMIS einerseits und den Entwurfsebenen andererseits kann nun ein konsistentes und zielgerichtetes Vorgehen abgeleitet werden. Die Praxis und die Literatur zeigen, dass jeder der beiden Hauptphasen "Entwurf" und "Umsetzung" zwei Entwurfsebenen zugewiesen werden können. Dies führt zu einem sehr systematischen, strukturierten, nachvollziehbaren und gut dokumentierbaren System-Entwicklungsprozess.

4.5.1 Planung (Strategie) => Von der Realität zum semantischen Entwurf

In der Hauptphase "Planung" wird ein sehr **grober semantischer Entwurf** erarbeitet. Er zeigt nur strategisch relevante Elemente, soweit sie für die Erarbeitung der Resultate der "Planung" erforderlich sind.

4.5.2 Entwurf (Taktik) => Vom semantischen Entwurf zum konzeptuellen Entwurf

In der Hauptphase "Entwurf" wird zuerst ein **vollständiger semantischer Entwurf** erarbeitet. Er umfasst alle Elemente, soweit sie für die Erarbeitung der Resultate zusammen mit den Benutzern erforderlich sind. Dieser Entwurf reicht für die mit allen wichtigen Benutzern durchzuführenden Tätigkeiten aus. Er geht aber auch nicht zu weit mit der Detaillierung.

In der Hauptphase "Entwurf" muss dann das "Pflichtenheft" bezüglich der Systemgestaltung für dessen Umsetzung, ev. auch für eine Ausschreibung, erarbeitet werden. Der **konzeptuelle Entwurf** enthält alle relevanten Elemente dazu. Die Abstraktion ist so, dass sowohl der Entwurfs-Benutzer, z.B. der Strassendatenbank-Leiter, als auch der Informatiker davon profitieren.

Er lässt aber die technischen Aspekte der Umsetzung offen. Diese können, sofern notwendig, in einem technischen Pflichtenheft formuliert werden.

4.5.3 Umsetzung => Vom konzeptuellen Entwurf zur logischen und physischen Umsetzung

In der Hauptphase "Umsetzung" muss das Pflichtenheft des Entwurfs umgesetzt werden. Es werden dabei die Entwurfsebenen der **logischen Umsetzung** und der **physischen Umsetzung** nacheinander durchlaufen. Dies ist die Arbeit der Informatiker.

4.5.4 Nutzung und Betrieb => Vom physischen Betrieb zur logischen und physischen Umsetzung in der Wartung

In der Nutzung und dem Betrieb kommen, bei der Einführung der semantische und der konzeptuelle Entwurf bei der Wartung, die logische und die physische Umsetzung zur Anwendung.

* * *

5 Kommunikationsbedürfnisse im Lebenszyklus des SMIS

5.1 Übersicht

Die Kommunikatonsbedürfnisse beim Entwurf und bei der Umsetzung sowie bei der Nutzung und dem Betrieb können zwei Arten von Information betreffen:

- **Metadaten**, d.h. Daten über die Daten und die Datenstrukturen
- **Strassendaten**, d.h. die eigentlichen, für den Benutzer interpretierbaren Inhalte der Strassendatenbank

Der vorliegende Bericht befasst sich mit der Kommunikation im Bereich der Metadaten.

Die Identifikation der wesentlichsten Informationsbedürfnisse geschieht auf Grund der in den vorhergehenden Kapiteln dargestellten Aktivitäten und der entsprechenden Beteiligten.

5.2 Kommunikationsbedürfnisse im Entwurf

Im Entwurf stehen vor allem die Kommunikationsbedürfnisse zwischen dem Management, den Benutzern, dem Strassendatenbank-Leiter und den Entwerfern im Vordergrund.

Dies verlangt nach **Beschreibungssprachen für den semantischen und den konzeptuellen Entwurf**.

Beispiele:

- Erarbeitung der Datenkatalog-Normen der VSS für Strassendatenbanken.
- Erarbeitung der Norm SIA405 für den Leitungskataster
- Erarbeitung der AV93 und der Datenbeschreibung mit INTERLIS

5.3 Kommunikationsbedürfnisse in der Umsetzung

In der Umsetzung stehen vor allem die Kommunikationsbedürfnisse zwischen den Entwerfern, dem Strassendatenbank-Leiter und den Umsetzern im Vordergrund.

Dies verlangt nach **Beschreibungssprachen für den konzeptuellen Entwurf einerseits, und Beschreibungssprachen für den logischen Entwurf andererseits.**

Beispiele:

- DB-Integration von KUBA-DB und STRADA-DB
- Kopplung von Geo-Informationssystemen und Strassendatenbanken für das Strassenmanagement

5.4 Kommunikationsbedürfnisse im Datenaustausch

Im Datenaustausch stehen vor allem die Kommunikationsbedürfnisse zwischen Benutzern, in der Regel die Strassendatenbank-Leiter, und zwischen Systemen im Vordergrund.

Für die Kommunikation zwischen den Benutzern beim Austausch wird eine **Beschreibungssprache für den semantischen und den konzeptuellen Entwurf** benötigt. Der Austausch zwischen den Systemen erfordert eine **Beschreibungssprache für den logischen Entwurf (Format, Codierung)**.

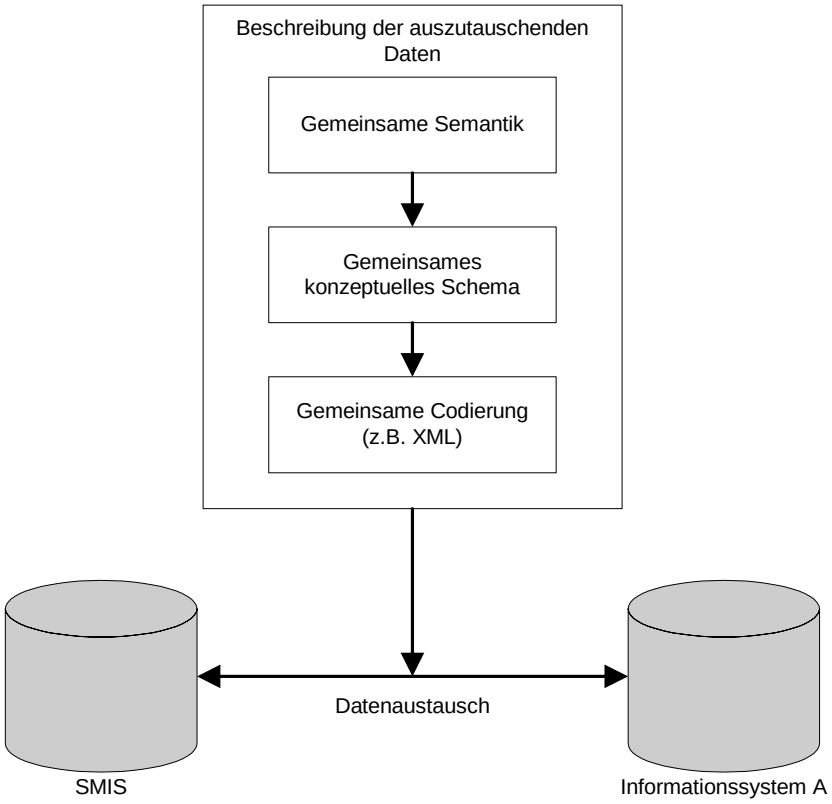


Abbildung 5-1: Voraussetzungen für den Datenaustausch

Wie in Abbildung 5-1 dargestellt, müssen zwei Betreiber von verschiedenen Informationssystemen, die Daten austauschen wollen, eine Einigung auf semantischer, konzeptueller und logischer Ebene bezüglich der auszutauschenden Daten erreichen. Da kaum ein Datenschema einem anderen Informationssystem aufgezwungen werden kann, bedeutet dies, dass ein konzeptuelles Austauschschema erstellt werden muss. Die beiden Informationssysteme müssen ihre auszutauschenden Daten so umstrukturieren, dass sie dem konzeptuellen Schema entsprechen. Die gemeinsam festgelegte Codierung bestimmt die Regeln, wie die Daten auf dem Datenträger anzuordnen sind.

Das konzeptuelle Datenschema (Austauschschema) spielt somit folgende Rolle:

- Dokumentation für den Entwurf des Austauschmechanismus
- Dokumentation der Daten, damit der Empfänger die Daten richtig interpretieren kann und die Qualität der Daten überprüfen kann
- Dokumentation, damit externe Beteiligte in der Austauschgemeinschaft teilhaben können

Falls die Semantik, das konzeptuelle Schema und die Codierung in einem Fachgebiet standardisiert werden (beispielsweise in SIA/VSS-Normen), so können erhebliche Kommunikationsprobleme vermieden werden. Anstatt dass bilaterale Austauschmechanismen entworfen werden, kann der Austausch durch eine standardisierte Schnittstelle sicher gestellt werden.

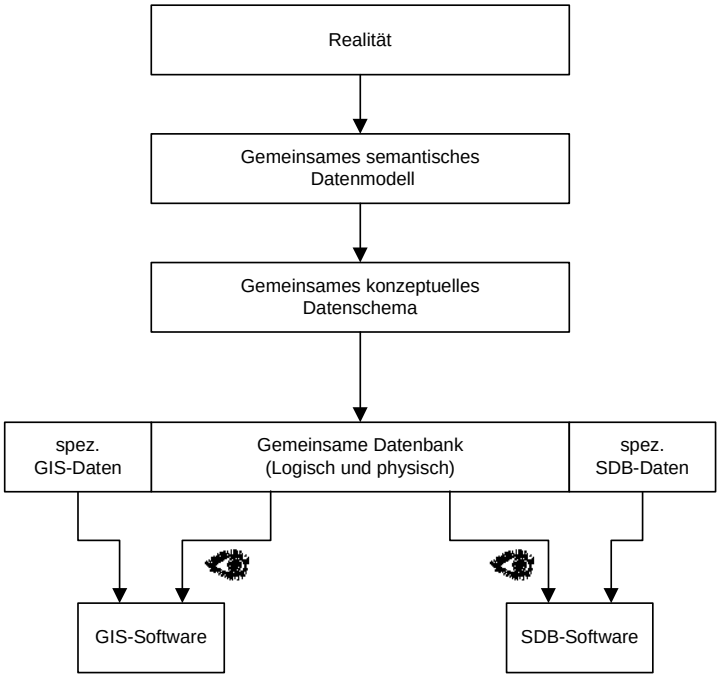


Abbildung 5-2: Kopplung von Geo-Informationssystemen und Strassendatenbanken

Eine andere Möglichkeit, mehreren Benutzern Daten zur Verfügung zustellen, ist der Entwurf einer gemeinsamen physischen Datenbank. Die Benutzer können mit ihren spezifischen Applikationen auf die gemeinsame Datenbank zugreifen. Damit sie die Daten richtig verarbeiten können, ist eine semantische und eine konzeptuelle Beschreibung der Datenstruktur auch in diesem Fall unumgänglich.

5.5 Kommunikationsbedürfnisse beim Erheben und Erfassen

Beim Erheben und Erfassen stehen vor allem die Kommunikationsbedürfnisse zwischen den Erhebern, dem Strassendatenbank-Leiter und den Benutzern im Vordergrund.

Dies verlangt nach **Beschreibungssprachen für den semantischen Entwurf** einerseits, und **Beschreibungssprachen für den konzeptuellen Entwurf** andererseits.

5.6 Kommunikationsbedürfnisse im Hinblick auf die Datenpflege

5.6.1 Semantische Konsistenz (Vollständigkeit, Plausibilität)

Für die Pflege der semantischen Konsistenz stehen vor allem die Kommunikationsbedürfnisse zwischen dem Strassendatenbank-Leiter und den Benutzern im Vordergrund.

Dazu wird eine **semantische und eine konzeptuelle Beschreibungssprache** benötigt.

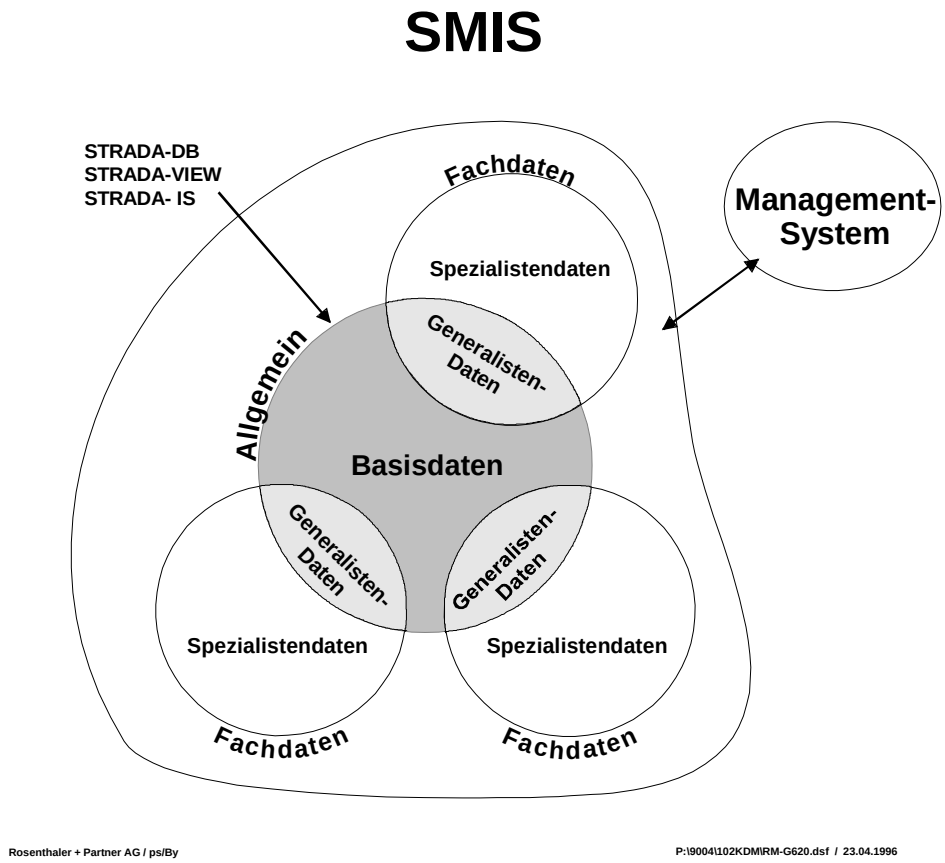


Abbildung 5-3: Fachdaten auf unterschiedlichen Datenbanken (STRADA)

Die Abbildung 5-3 zeigt die Problematik, wenn die semantische Konsistenz nicht sicher gestellt wird. Für jeden Fachprozess werden Daten benötigt, die sich aus Spezialistendaten, Generalistendaten und Basisdaten zusammensetzen. Spezialistendaten sind Daten, die nur einem Fachprozess zur Verfügung gestellt werden müssen, d.h. Fachprozess-spezifisch. Generalistendaten hingegen sind Daten, die von mehreren Fachprozessen benötigt werden. Die Basisdaten sind diejenigen Daten, die, unabhängig von Fachprozessen, allen Benutzern zur Verfügung gestellt werden. Die Gesamtheit aller Generalistendaten und der Basisdaten kann als "Allgemeine Daten" bezeichnet werden.

Hier ist eine zentrale Verwaltung der "Allgemeine Daten" wünschenswert. Eine Unvollständigkeit der Basisdaten oder von Generalistendaten beispielsweise (d.h. Fehler in der semantischen Konsistenz) kann zu falschen Entscheidungen in einem oder mehreren Fachprozessen führen.

Die Erhaltung der semantischen Konsistenz kann durch folgende Massnahmen erreicht werden:

- Festlegung der Zuständigkeiten in der Verwaltung der Basisdaten. Wenn zwei Strassendatenbank-Leiter dieselben Daten nachführen, kann dies zu Widersprüchen führen.
- Bezeichnung der inkonsistenten Daten als solche, beispielsweise durch einen Integritätstatus (z.B konsistent, unvollständig, Plausibilität nicht geprüft usw.).
- Abgabe des Datenschemas der Gesamtheit der Basisdaten an alle Beteiligten, so dass jeder die Konsequenzen einer Nachführung absehen kann

Für die semantische Konsistenz der Spezialistendaten ist jeder Benutzer selbst verantwortlich.

5.6.2 Strukturelle Integrität

Für die Pflege der strukturellen Integrität, insbesondere der Konsistenz, steht ebenfalls die Kommunikation zwischen dem Strassendatenbank-Leiter und den Benutzern im Vordergrund.

Sie benötigen dazu eine **semantische, eine konzeptuelle und, teilweise, auch eine logische Beschreibungssprache.**

5.7 Kommunikationsbedürfnisse im Hinblick auf Abfrage und Auswertung

Für Abfragen und Auswertungen stehen die Kommunikationsbedürfnisse zwischen dem Management, den Benutzern und dem Strassendatenbank-Leiter im Vordergrund.

Sie benötigen dazu eine **semantische und eine konzeptuelle Beschreibungssprache**.

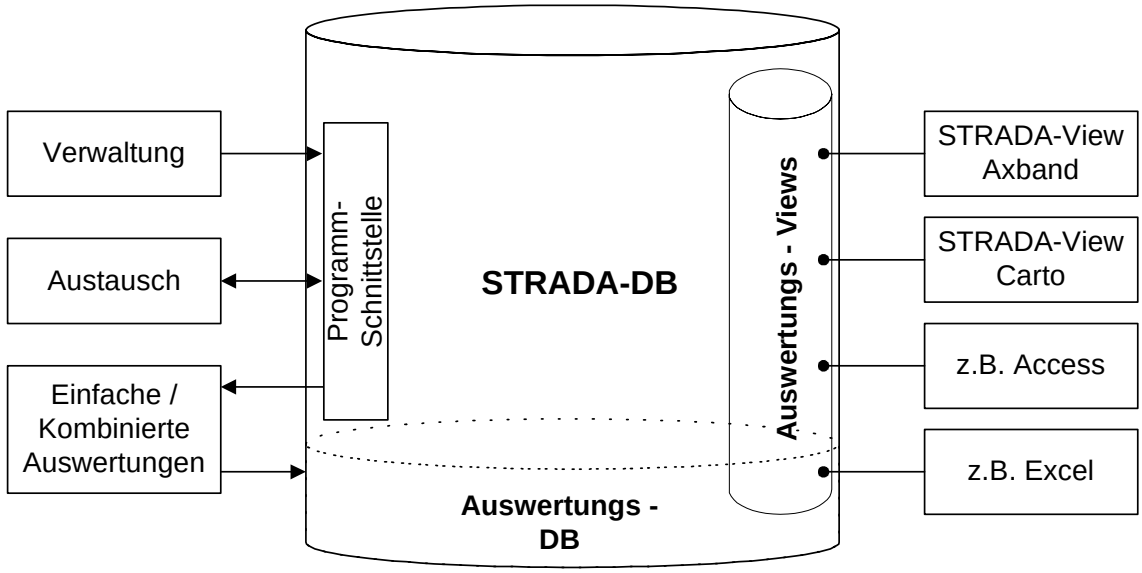


Abbildung 5-4: Die Auswertungswerkzeuge von STRADA-DB

Die Abbildung 5-4 zeigt die Struktur der Strassendatenbank STRADA-DB und der Auswertungsmöglichkeiten, resp. Auswertungswerkzeuge, die zur Verfügung stehen. Die Auswertungswerkzeuge müssen sowohl die semantischen als auch die konzeptuellen Modelle, resp. Schematas kennen, um überhaupt arbeiten zu können.

5.7.1 Kombinierbarkeit

Die Aspekte der Kombinierbarkeit von Strassendaten spielen bei Abfragen und Auswertungen eine für die Nutzung des SMIS ausschlaggebende Rolle. Die Möglichkeit der Kombinierbarkeit und die Verifikation der Kombinierbarkeit können nur auf der Basis der Dokumentation des konzeptuellen Entwurfs gewährleistet werden.

Dazu wird eine **semantische und eine konzeptuelle Beschreibungssprache** benötigt.

5.7.2 Vergleichbarkeit

Die Aspekte der Vergleichbarkeit von Strassendaten spielen bei Abfragen und Auswertungen eine für die Nutzung des SMIS ausschlaggebende Rolle. Die Möglichkeit und die Verifikation der Vergleichbarkeit können nur auf der Basis der Dokumentation des konzeptuellen Entwurfs durchgeführt werden.

Dazu wird eine **semantische und eine konzeptuelle Beschreibungssprache** benötigt.

5.8 Kommunikationsbedürfnisse im Bereich der Raum-/Zeit-Aspekte von SMIS

Die Strassen-spezifischen Aspekte der Raum- und der Zeit-Aspekte begründen auch spezifische Kommunikationsbedürfnisse

5.8.1 RBBS

Beim räumlichen Basis-Bezugssystem RBBS (gemäss SN640910) stehen Kommunikationsbedürfnisse zwischen dem Strassenunterhalt, als Besitzer des RBBS, dem Management, den Benutzern, den Entwerfern und dem Strassendatenbank-Leiter im Vordergrund.

Sie benötigen dazu eine **semantische und eine konzeptuelle Beschreibungssprache**.

5.8.2 Topologien

Im Bereich der Topologien stehen Kommunikationsbedürfnisse zwischen dem Management, den Benutzern, den Entwerfern und dem Strassendatenbank-Leiter im Vordergrund.

Sie benötigen dazu eine **semantische und eine konzeptuelle Beschreibungssprache**.

5.8.3 Geometrien

Im Bereich der Geometrien stehen Kommunikationsbedürfnisse zwischen dem Strassenbauern, den Benutzern, den Entwerfern und dem Strassendatenbank-Leiter im Vordergrund.

Sie benötigen dazu eine **semantische und eine konzeptuelle Beschreibungssprache**.

5.8.4 Zeit und Historisierung

Im Bereich der Zeit und der Historisierung stehen Kommunikationsbedürfnisse zwischen dem Management, den Benutzern, den Entwerfern und dem Strassendatenbank-Leiter im Vordergrund.

Sie benötigen dazu eine **semantische und eine konzeptuelle Beschreibungssprache**.

5.9 Weitere Kommunikationsbedürfnisse

5.9.1 Wissenskataloge

Ein spezieller Teil des SMIS stellt die Verwaltung des Wissens dar. Daraus entstehen vor allem Kommunikationsbedürfnisse zwischen dem Management, den Fachspezialisten – als Träger des Wissens - , den Benutzern, den Entwerfern und dem Strassendatenbank-Leiter im Vordergrund.

Sie benötigen dazu eine **semantische und eine konzeptuelle Beschreibungssprache**.

5.9.2 Darstellungsparameter

Insbesondere im Datenaustausch kommen auch Darstellungsparameter als auszutauschende Daten in Frage. Dann muss auch für diese eher technischen Grössen ein entsprechendes Datenschema zur Verfügung gestellt werden.

5.10 Übersicht über die Kommunikationsbedürfnisse

Die folgende Tabelle zeigt die Verwendung der Modell-(Schema) Metadaten für alle besprochenen Informationsbedürfnisse.

Kommunikations-Bedürfnisse	Semantischer Entwurf	**Konzeptueller Entwurf**	Logischer Entwurf
Entwurf	Informations-Analyse	**Entwurfs-Dokumente**	-
Umsetzung	-	**= Pflichtenheft**	Entwickl.Dok.
Datenaustausch	Benutzer	**Benutzer**	EDV-Systeme
Erheben und Erfassen	Benutzer	**Vorbereitung, Kontrolle**	-
Semantische Konsistenz	Semantik	**Struktur, Inhalte**	-
Strukturelle Integrität	(Struktur)	**Strukturen**	(Umsetzung)
Kombinierbarkeit	Begriffe	**Abfrage, Auswertung**	-
Vergleichbarkeit	Begriffe	**Abfrage, Auswertung**	-
RBBS	Fachsicht	**Konzeptuelle Sicht**	-
Topologien	Fachsicht	**Konzeptuelle Sicht**	-
Geometrien	Fachsicht	**Konzeptuelle Sicht**	-
Zeit und Historisierung	Fachsicht	**Konzeptuelle Sicht**	-
Wissenskataloge	Katalog-Verständnis	**Katalog-Pflege**	-
Darstellungsparameter	Symbol-Bedeutung	**Symbol-Verwendung**	Symbol-Austausch

Die Übersicht zeigt die Notwendigkeit der konzeptuellen Beschreibungssprache deutlich.

* * *

6 Entwurfs- und Umsetzungs-Paradigmen

6.1 Übersicht

Zwei wesentliche Aspektgruppen sind bei der Betrachtung der Entwurfs- und Umsetzungs-Paradigmen (strukturierter Entwurf und Objekt-orientierter Entwurf) zu betrachten:

- Die Behandlung von **Information** (Daten und ihre Strukturen), **Funktionen** (und ihren Strukturen) und der **Rahmenorganisation** (insbesondere Aktivitäten und Abläufe)
- Die **Strukturierung der Schritte** während dem Entwurf und der Umsetzung (Phasenmodelle, zyklische Entwicklung usw.)

Im Folgenden sollen diese beiden Aspektgruppen sowohl für den strukturierten Entwurf als auch für den Objekt-orientierten Entwurf und die entsprechende Entwicklung beschrieben werden.

6.2 Der strukturierte Entwurf und die entsprechende Umsetzung

Der strukturierte Entwurf und dessen Umsetzung trennen wesentliche Aspekte des zu realisierenden Systems und schaffen Unabhängigkeit durch die Trennung von

- Daten,
- Funktionen,
- (Rahmen-) Organisation.

Zum Beispiel erlaubt die **Unabhängigkeit zwischen Daten und Funktionen** gewisse Änderungen des Datenmodells, ohne dass damit die Funktionen betroffen sind.

Der strukturierte Entwurf wird heute in der Regel schrittweise und oft noch in relativ starren Phasenmodellen umgesetzt. Er schliesst aber eine zyklische Entwicklung nicht aus.

Die **zyklische Entwicklung bringt die in Kapitel 4.2.3 erwähnten Vorteile**. Sie bedarf aber einer intensiven Koordination zwischen den relativ stark getrennten Aspekten der Information, der Funktion (d.h. den Applikationen) und der entsprechenden Rahmenorganisation.

6.2.1 Information/Daten

Die Daten werden im strukturierten Entwurf in der Regel wie folgt analysiert und beschrieben:

- **Informationsobjekt-Typen**
- **Attribute und ihre Wertebereiche**
- **Beziehungen und Regeln zwischen Informationsobjekt-Typen und zwischen Attributen**

6.2.2 Funktionen/Applikationen

Die strukturierten Entwurfs- und Umsetzungsmethoden bilden die Funktionen in der Regel mit einer sogenannten "functional decomposition" ab, zu deutsch mit "**Funktionen-Hierarchie**" übersetzbar.

6.2.3 (Rahmen-) Organisation

Die Rahmenorgansiation umfasst verschiedene Aspekte wie:

- **Organisations-Struktur** des Systembetriebs
- **Personelle Ressourcen** für den Systembetrieb
- **Aufgaben und Verantwortungen** der Beteiligten
- **Abläufe und Termine** für den Systembetrieb
- **Sachmittel** für den Systembetrieb
- usw.

6.3 Der Objekt-orientierte (OO-) Entwurf und die entsprechende Umsetzung

Die Objekt-Orientierung (OO) bringt einen gänzlich anderen Ansatz des Entwurfs und der Umsetzung. Insbesondere wird nicht mehr zwischen dem Entwurf der Information und den Funktionen unterschieden. Vielmehr werden in den (Informations-) Objekten sowohl deren Daten als auch deren Funktionen modelliert und auch entsprechend umgesetzt. Teilweise können sogar auch Aspekte der Rahmenorganisation in den Objekt-Entwurf und die Umsetzung integriert werden.

Wesentliche Aspekte der **OO-Modellierung** sind dabei:

- **Kapselung**: Die Daten und die Funktionen sind in der Regel in den Objekttypen (Klassen) gekapselt und stehen nur über Funktionsaufrufe an die Objekte (Nachrichten, engl. messages) zur Verfügung.
- Daten und Funktionen können auch **in den Objekttypen (Klassen) versteckt** werden, d.h. sie stehen von aussen nicht zur Verfügung (engl. information hiding)
- **Vererbung**: Sub- (Objekt-) Typen erben die Eigenschaften ihrer Supertypen, sowohl bezüglich der Daten als auch der Funktionen
- **Overriding**: Gewisse Eigenschaften können in den Sub-Typen geändert werden
- **Polymorphismus**: ein Objekt-Typ kann generische Funktionen zur Verfügung stellen. Je nach dem Übergabeparameter in der Meldung und Objekt führt es eine andere, z.B mathematische Funktion aus.

Der OO-Entwurf eignet sich hervorragend für Transaktions-orientierte Systeme, z.B. Management-Systeme der Strassenerhaltung. Bei reinen Abfragesystemen kommen die Vorteile des OO-Entwurfes nicht so ausgeprägt zur Geltung. Die zu modellierenden Methoden (Funktionen der Objektklassen) beschränken sich auf die Verwaltung der Daten und die primär zur Aufrechterhaltung der Integrität der Datenbank erstellten Regeln.

6.4 Aspekte von Objekten bei beiden Paradigmen

Objekte können in beiden Paradigmen vielfältige Aspekte aufweisen.

Folgende Eigenschaften gelten innerhalb der strukturierten und des OO-Paradigmas:

- Objekte gehören zu einem **Objekt-Typ** (Umsetzung: Klasse).
- Zwischen den Objekten zweier oder mehrerer Objekt-Typen können **Assoziationen** (Beziehungen) definiert sein.
- Objekt-Typen haben innere **Beziehungen zu ihren Attributen**.
- Objekt-Typen können **Spezialisierungen oder Generalisierungen** anderer Objekt-Typen definieren (engl.: Gen-Spec).
- Objekt-Typen können **Aggregationen oder Kompositionen** von anderen Objekt-Typen definieren.
- Objekt-Typen können **Zustände für ihre Objekte** definieren.
- **Operationen** von Objekten zweier oder mehrerer Objekt-Typen können definiert werden.
- Objekte können in der Umsetzung **geographisch verteilt** werden.
- Objekte mehrerer Objekt-Typen können in der Umsetzung zu **Paketen** "zusammengebaut" werden.

Folgende Eigenschaften, neben den in 6.3 genannten, gelten nur innerhalb des OO-Paradigma:

- Objekt-Typen definieren **private, vererbbare oder öffentliche Funktionen** (Umsetzung: Methoden).
- Zwischen Objekten zweier oder mehrerer Objekt-Typen können **Interaktionen** definiert sein: z.B. Funktions-Sequenzen, Kollaboration usw.

* * *

7 Beschreibungssprachen für den Entwurf eines Informationssystems

7.1 Anforderungen an Sprachen für die Datenmodellierung

Im Rahmen der CEN TC 287 wurde der Bericht CR 287005 veröffentlicht, der anhand einer Liste von Anforderungen unterschiedliche Sprachen evaluierte. Diese Forschungsarbeit wurde von der ISO TC 211 fortgeführt.

Der daraus entstandene Bericht unterscheidet folgende Anforderungen :

1. **Formale Sprache** auf konzeptueller Ebene : Die Sprache muss formal und auf die konzeptuelle Ebene anwendbar sein.
2. **Elemente des konzeptuellen Schemas :**
   - **2.1 Struktur** : Sie muss Datentypen, Konsistenzbedingungen, Wertebereiche, Entitätstypen, Attribute und Relationen darstellen können.
   - **2.2 Verhalten** : Die Sprache sollte OO-Technologien berücksichtigen. Es sollte möglich sein, Mehrfachvererbung und Methodenbeschreibung zum Ausdruck bringen zu können.
   - **2.3 Bedingungen** : Konsistenzbedingungen sollten durch die Sprache dargestellt werden können.
3. **Graphische Notation** : Die Sprache sollte eine graphische Notation beinhalten oder zumindest einfach mit einer graphischen Notation verbunden werden können.
4. **Computer-Verarbeitbarkeit** : Sie soll durch Computer interpretierbar sein.
5. **Modularität** : Sie soll Unterschemata definieren können und auf vordefinierte Unterschemata und Strukturen zurückgreifen können.
6. **Geometrische Aspekte** : Sie liefert Methoden zur Beschreibung geometrischer und topologischer Elemente und Strukturen.
7. **Transfer-Schemata** : Sie soll durch einfache Regeln kodierbar sein.
8. **Dokumentation** : Die Beschreibungssprache soll gut dokumentiert sein.
9. **Internationaler Standard** : Sie sollte ein offizieller, wenn möglich international akzeptierter Standard sein.
10. **Unterstützende Software** : Es sollten genügend Software-Programme existieren, die diese Sprache verarbeiten können.
11. **Erfahrung und Support** : Sie sollte getestet und in der Praxis schon benutzt worden sein.

Die Autoren des vorliegenden Forschungsberichts erachten diese Forderungen als zielführend, auch wenn es vermutlich (noch) keine Beschreibungssprache gibt, die alle Forderungen erfüllt.

7.2 Datenmodellierung: Methoden (conceptual Modeling)

7.2.1 Übersicht über Modellierungsmethoden

Eine Modellierungsmethode besteht in der Regel aus

- einer sachlichen, z.B. mathematischen Grundlage für die Modellierung selbst (d.h. ein entsprechendes Meta-Modell für die Modellierung),
- einer Sprache zur Beschreibung der Resultate der Methode, und
- den Prozessen, die die Methode zur Modellierung verwendet.

Wir beschränken uns hier auf die **relationale** und die **OO-Modellierung**.

Die nachfolgende Figur zeigt eine Darstellung aus dem Arbeitspapier N222 der ISO/TC211/WG1, die die Zusammenhänge zwischen der **Modellierung (-smethode)** und der **Beschreibung (-sprache)** sehr schön zeigt. Das Modell ist ein geistiges Produkt, das erst durch die **Darstellung in einem Schema**, unter Verwendung einer entsprechenden Sprache (graphisch oder textuell) kommuniziert werden kann. Die Sprachen für die Datenmodellierung werden im Kapitel 7.3 erläutert.

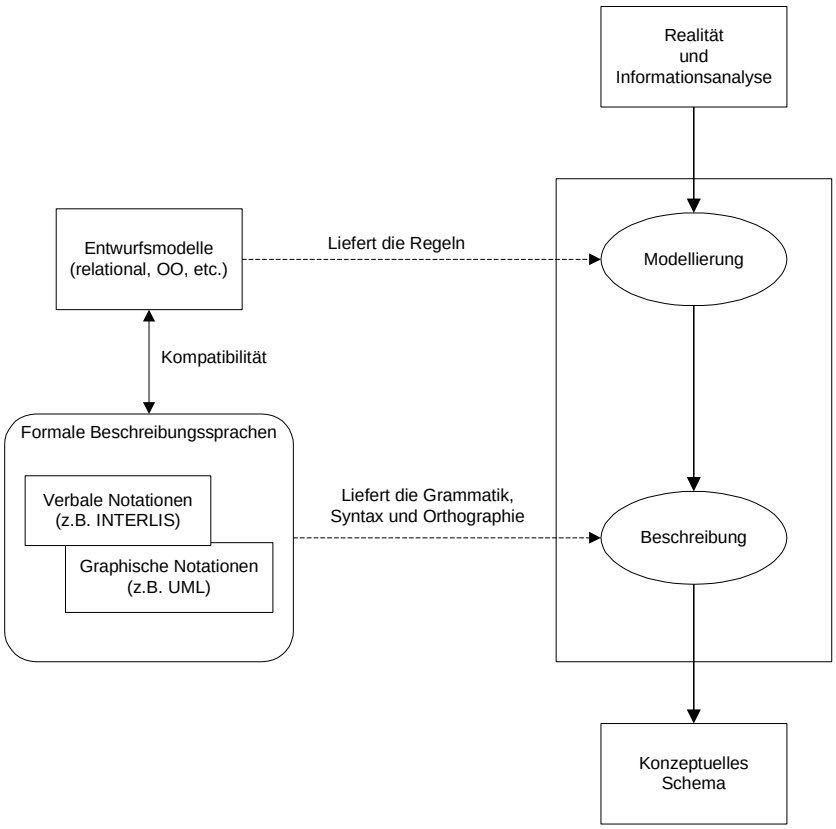


Abbildung 7-1: Metamodell der Datenmodellierung

7.2.2 Relationale Modellierung

Die relationale Modellierung wurde vom amerikanischen Informatiker E. Codd 1971 entwickelt und in den siebziger und achziger Jahren eingeführt. Sie löste hierarchische und netzwerkartige Modellierungsmethoden ab.

Sie stützt sich auf die Relationen-Algebra der Mengen-Mathematik ab. **Relationen** sind dort "Mengen" von möglichen Ausprägungen von Kreuzprodukten mehrerer Attribut-Domänen. Darum enthält ein relationales Modell die Daten in Datensätzen (Zeilen, engl. rows) in Tabellen (engl. tables) mit fest definierten Attributen (Spalten, engl. columns).

Ein sauber relational modelliertes Datenmodell soll mindestens in dritter Normalform sein. Dies bedeutet:

- Die Attribut-Ausprägungen sind atomar, d.h. keine multiplen Felder oder Periodenbrüche (1. Normalform)
- Die Nicht-Schlüssel-Attribute sind vom gesamten Schlüssel abhängig (2. Normalform)
- Die Nicht-Schlüssel-Attribute sind nicht transitiv voneinander abhängig d. h. sie sind nur vom Schlüssel abhängig (3. Normalform)

Das relationale Modell führt zu sehr **Redundanz-armen Datenmodellen**, die zudem für den Benutzer sehr einfach verständlich sind. Deshalb wird in der Norm [SN640909] postuliert, dass die Datenmodelle für Strassendatenbanken mit dem relationalen Modell zu entwerfen sind.

7.2.3 OO-Modellierung

Das in Kap. 6.3 über den OO-Entwurf gesagte gilt ebenso für die OO-Modellierung.

7.3 Sprachen für die Datenmodellierung

Im Folgenden werden die gängigsten Modellierungssprachen vorgestellt:

- Entity-Relationship-Model (ERM) und ER-Diagramm (ERD)
- Formale Algebra
- EXPRESS (STEP)
- INTERLIS
- Unified Modeling Language (UML)

7.3.1 ERM und ERD

Das Entitäten-Relationship Modell (ERM) wurde 1975 von Peter P. Chen erstmals vorgestellt. Chen stellte fest, dass zwischen den Entwurfs- oder Umsetzungsmodellen wie beispielsweise die relationale, die netzartige oder die hierarchische Modellierung, und der semantischen Beschreibung der Informationsobjekte eine zu grosse Lücke bestand. Durch die Einführung des ERM beschrieb er erstmals eine Entwurfsmethode, die diese Lücke füllte.

Diese ERM Entwurfsmethode wurde im Laufe der Zeit ständig weiterentwickelt (was unter anderem zu MERISE führte). Die Gesamtheit der Entwurfsmethoden, die sich auf ERM stützen, wurde in diesem Bericht mit dem Begriff "Strukturierte Entwurfsmethoden" zusammengefasst.

Wird heute der Begriff ERM gebraucht, wird meistens das Entity-Relationship-Diagram (ERD) gemeint. Das ERD wurde von Chen entwickelt, um eine von Umsetzungsmethoden unabhängige Darstellung der Informationsobjekte zu ermöglichen, d.h. ohne das man sich für eine relationale, hierarchische oder netzartige Umsetzung entschieden hat.

Es gilt zu erwähnen, dass damals die Umsetzungsmethoden (relationales, hierarchisches oder netzartiges Modell) gleichzeitig auch Entwurfsmethoden waren, weil das zeitliche Vorgehen diskussionslos linear (Wasserfall-Modell) verlaufen musste und weil die zyklischen Phasenmodelle noch nicht üblich waren.

7.3.2 Formale Algebra

Die formale Algebra wird auch Prädikaten-Kalkül oder relationale Algebra genannt. Sie beschreibt eine Anzahl von Operatoren, mit der unterschiedliche Tabellen in Beziehung gebracht werden können. So ist es möglich, eine Abfrage mit Hilfe der formalen Algebra auszuführen und das entsprechende Resultat in eine temporäre Tabelle abzulegen.

Man kann zwei Arten von Operatoren unterscheiden :

- Die Operatoren, die aus einer mathematischen Theorie stammen, wie beispielsweise die Vereinigung, die Verschneidung, die Differenz oder das Produkt (Boolsche Operatoren).
- Die Operatoren, die für relationale Datenbanken eigens festgelegt wurden, wie beispielsweise die Projektion, die Verbindung, die Division und die Umbenennung.

Die relationale Modellierungsmethode stützt sich auf die formale Algebra, und somit auf ein sauberes mathematisches Fundament.

7.3.3 EXPRESS (STEP)

STEP (Standard for the Exchange of Product Model Data) findet seinen Ursprung in der Produktdatentechnologie. Darin wird die Interoperabilität zwischen verschiedenen IT-Systemen gesucht, um ein sogenanntes "Symultaneous Engineering" überhaupt zu ermöglichen. STEP wurde insbesondere unter Berücksichtigung der gängigen Formate der CAD bzw. CAM-Systeme entwickelt, um Produktdaten wie Entwurfsdaten, Simulationsdaten usw. austauschen zu können.

Um die Lesbarkeit dieses Formats zu erleichtern, wurde die Beschreibungssprache EXPRESS von ISO 10303 übernommen und mit graphischen Komponenten ergänzt (EXPRESS-G). EXPRESS ermöglicht die verbale Beschreibung eines Produkt- oder Datenmodells.

7.3.4 INTERLIS

INTERLIS ist eine Beschreibungssprache, die 1991 definiert wurde, um in der föderalistischen Schweiz eine standardisierte Schnittstelle in der amtlichen Vermessung einzuführen. Durch die Einführung der Technischen Verordnung der Amtlichen Vermessung TVAV 1994 wurde das erste Datenmodell mit INTERLIS beschrieben. Damit wurde zugleich seine Benutzung innerhalb des Vermessungswesens gewährleistet. INTERLIS ist heute auch ausserhalb der Vermessung ein Standard in der Schweiz geworden, weil er sich als einzige Beschreibungssprache in der Schweiz in der Praxis bewährt hat.

7.3.5 UML (OO)

Ende der achziger Jahre begann man das "Objekt"-Konzept, welches in Programmiersprachen wie C++ oder SmallTalk genutzt wurde, als Analyse-Methode anzuwenden

UML 1.0 wurde im Januar 1997 dank der gemeinsamen Bemühungen von Booch, Rumbaugh und später von Jacobson von der Firma Rational Software veröffentlicht.

UML ist eine Modellierungssprache (nicht eine Methode), die auf den schon vor dreissig Jahren existierenden "Objekt-orientierten" Konzepten beruht. UML vereinigt frühere OO-Modellierungs-konzepte und definiert sie (formale Definitionen).

Die Grundkonzepte der OO-Programmierung bleiben in UML bestehen: **Klasse, Objekt, Kapselung, Vererbung, Overriding** und **Polymorphismus**.

7.4 Meta-Modelle von Sprachen für die Datenmodellierung

7.4.1 Elemente von Metamodellen

Die zur vollständigen Beschreibung erforderlichen Elemente einer Sprache können in die folgenden Aspekte gegliedert werden:

7.4.1.1 Semantik

Die Semantik beschreibt die Bedeutung der einzelnen Sprachelemente, der "Wörter". Dies können tatsächlich umgangssprachliche Wörter sein. Es können aber auch graphische Symbole verwendet werden.

7.4.1.2 Syntax

Die Syntax beschreibt die "Satzstellung". Diese spielt insbesondere bei textuellen Beschreibungssprachen eine grosse Rolle. Sie zeigt aber auch bei graphischen Notationen die möglichen und erlaubten Kombinationen der in der Semantik definierten Elemente.

7.4.1.3 Grammatik

Eine eigentliche Grammatik existiert bei Beschreibungssprachen für Datenmodelle nicht. Es wird weder konjugiert, noch dekliniert usw.

7.4.1.4 Orthographie

Die Orthographie stellt die Rechtschreibung sicher. Die Orthographie spielt auch für Datenmodellierungssprachen eine grosse Rolle. Die Sprachelemente müssen korrekt verwendet werden, um eine korrekte Aussage zu ermöglichen.

7.4.2 Das Metamodell von INTERLIS

Das Metamodell von INTERLIS ist im Anhang E beschrieben.

Eine Beschreibung in INTERLIS wird "objekt-relational" bezeichnet. Sie ist "relational", weil die generelle Struktur der INTERLIS-Beschreibung diesem Metamodell entspricht. Sie ist "objekt", weil sie gewisse OO-Konzepte verwendet.

7.4.3 Das Metamodell von UML

Die Komplexität des UML-Metamodells Version 1.0 ist enorm. Es umfasst 90 Metaklassen und 100 Meta-Assoziationen, und darin werden bis zu 50 Stereotypen beschrieben. Um diese Komplexität zu strukturieren, wird das Metamodell in Gruppen unterteilt. Alle stark abhängigen Metaklassen werden in einer Gruppe beschrieben, während dem die lose gebundenen Metaklassen in einer anderen Gruppe beschrieben werden.

Es werden drei Hauptgruppen unterschieden:

- Methoden-Beschreibungselemente
- Grundelemente wie beispielsweise Wertebereichstypen
- Modell-Management

Im Anhang G wird das UML-Metamodell stark vereinfacht beschrieben.

7.5 Notationen für die Präsentation der Sprachelemente für die Datenmodellierung

7.5.1 Übersicht über Notationen

Zu jeder konzeptuellen Beschreibungssprache wurde eine entsprechende Notation eingeführt. Falls eine Beschreibungssprache keine Notation vorschlägt, muss bezweifelt werden, ob die Sprache auf konzeptueller Ebene anzusiedeln ist. Die Notation ermöglicht die eindeutige Darstellung gewisser Aspekte eines Datenmodells. Nach Ansicht der Autoren sind verbale Notationen dann zu benutzen, wenn die Vollständigkeit eines konzeptuellen Datenschemas im Vordergrund steht. Graphische Notationen werden dann benutzt, um dem Entwerfer einen Überblick über die wesentlichen Aspekte eines Datenmodells geben zu können.

Beispiele **graphischer Notationen** sind: Entity Relationship Diagramm (ERD) von Chen, NIAM-Diagramme, OMT von Rumbaugh, Syntropy, UML, EXPRESS-G, MADS.

Beispiele **verbaler Notationen** sind: INTERLIS, EXPRESS-G.

Verbale Notationen auf konzeptueller Ebene gibt es nur sehr wenige. Die meisten Beschreibungssprachen sind auf logischer Ebene anzusiedeln, wie beispielsweise SQL-DDL.

7.5.1.1 Aufgabe der Notation

Eine Notation umfasst alle Beschreibungselemente, die ein Modell (Datenmodell, Auswertungsmodell, Organisationsmodell oder sonstiges) widerspruchsfrei und eindeutig darstellen kann.

Beispiele für Notationen sind Programmiersprachen, die eine Funktion oder ein Datenmodell darstellen. Programmiersprachen ermöglichen eine Interpretation einer Darstellung durch Spezialisten und deren Umsetzung durch ein Informatikwerkzeug.

Für die konzeptuelle oder semantische Datenbeschreibung steht die Lesbarkeit im Vordergrund, nicht die Verarbeitbarkeit durch Informatikwerkzeuge

Häufig wird eine konzeptuelle Beschreibungssprache benutzt, um Daten bei einem Datentransfer zu beschreiben.

7.5.1.2 Graphische Notationen

Graphische Notationen werden zur Vermittlung des statischen bzw. des dynamischen Modells eines Informationssystems genutzt. Die Kommunikation zwischen den Beteiligten steht im Vordergrund. Die graphische Notation stellt einen "Plan" des zukünftigen Informationssystems dar, um die Diskussion der Systemanalytiker und –modellierer zu fördern. Eine vereinfachte Notation erlaubt sogar, Benutzer in die Systemmodellierung einzubeziehen (sei es zur Analyse des gegenwärtigen Informationssystems, das sog. Business Modeling, oder für das zukünftige Informationssystem mit integrierter IT-Lösung). Die graphische Notation ermöglicht eine Übersicht über das ganze System, indem Elemente weggelassen werden, die zur logischen Architektur nur wenig beitragen.

Zur vollständigen Übermittlung des Modells eignet sich eine graphische Notation nur wenig, ohne dass sie durch etliche verbale Kommentare ergänzt wird. Solche Ergänzungen erschweren die Lesbarkeit des Modells und die Computer-Verarbeitbarkeit erheblich. Dafür sollte eine verbale Notation in Betracht gezogen werden.

7.5.1.3 Verbale Notationen

Verbale Notationen haben eine IT-verarbeitbare, vollständige und widerspruchsfreie Modellbeschreibung zum Ziel. Sie werden dazu gebraucht, um System-Strukturen in IT-Systemen einzulesen, wie beispielsweise die Übergabe einer Struktur in ein CASE-Tool, die automatische Strukturerstellung in einer Datenbank, oder die Interpretation der Daten durch eine Zieldatenbank bei einem Datentransfer.

Die Einbindung von Datenmodellen, beispielsweise in Normen, wird mit Hilfe von verbalen Notationen dank deren eindeutigen Beschreibungsmöglichkeiten überhaupt erst ermöglicht.

Verbale Notationen sind jedoch als Diskussionbasis für alle Beteiligten während der Systementwicklungsphase schlecht geeignet.

7.5.2 Die Notation von INTERLIS (Anhänge D,E,F,M)

Die INTERLIS-Notation wurde bereits für die Beschreibung der Schnittstelle der amtlichen Vermessung genutzt, die in der technischen Verordnung (TVAV) festgelegt ist. Diese Notation zeichnet sich bezüglich ihrer einfachen Lesbarkeit und Struktur (im Gegensatz zu EXPRESS) aus. Zudem können geographische "Features" wie etwa Punkt, Linie oder Fläche leicht beschrieben werden. Die Version 2 von INTERLIS wurde durch Begriffe der OO-Datenmodellierung ergänzt. Was INTERLIS aber weiterhin vermissen lässt, ist die Formalisierung der Konsistenzbedingungen (u.a. topologische Konsistenzbedingungen), wie auch die Methoden, die unabdingbar für eine IT-Verarbeitbarkeit sind. Zwar kann jede Konsistenzbedingung durch einen Kommentar wiedergegeben werden, aber das System kann diese Information nicht verwerten.

Dennoch kann INTERLIS besonders für den Datenaustausch in Betracht gezogen werden, vor allem für IT-Systemen, bei denen die Daten-Qualität durch den Benutzer garantiert werden kann, indem er die Beschreibung konsultiert.

7.5.3 Die Notation von UML (Anhang G)

UML ist eine graphische Notation, die noch vermehrt zum Einsatz kommen wird. Die OO-Modellierung wird in Zukunft auf UML nicht mehr verzichten können. Sie unterteilt die Modellierungskonzepte in Diagramme, die unterschiedliche Sichten eines Benutzers auf ein Informationssystem wiedergeben. Diese Diagramme sind zum Teil stark redundant, so dass die Nachführung dieser Diagramme schwierig ist. Wie jede graphische Notation vereinfacht sie die Kommunikation zwischen den verschiedenen Beteiligten in einem IT-Entwicklungsteam. Es scheint aber schwierig, ein vollständiges UML-Modell an Benutzer zu vermitteln, da UML doch wesentlich komplexer ist, als beispielsweise ein Entitäten-Relationen-Diagramm für die Darstellung eines Datenmodells (dieses kann auch von der etwas umfassenderen OO-Modellierung her erklärt werden).

7.5.4 Die Notation von NIAM

NIAM (Nijssen's Information Analysis Method) ist eine Entwurfsmethode, die 1974 von G.M. Nijssen eingeführt wurde. Die NIAM-Methode beinhaltet zwei graphische Notationen: IFD (Information Flow Diagrams) für die Darstellung von Informationsfluss innerhalb der Funktionen und ISD (Information Structure Diagram) für die Darstellung der Datenstruktur. Heute spricht man von NIAM-Diagrammen und meint damit die mit Hilfe von ISD erstellten Schemata.

Die NIAM-Notation wird in den GDF-Normen der CEN TC 278 und der ISO TC 204 gebraucht.

* * *

8 Die Beschreibungssprache für das SMIS, resp. die Strassendatenbank

8.1 Einleitung

Zwei Wege der Beschreibungssprache für SMIS, resp. für die Strassendatenbank sind heute in der Schweiz vorgezeichnet:

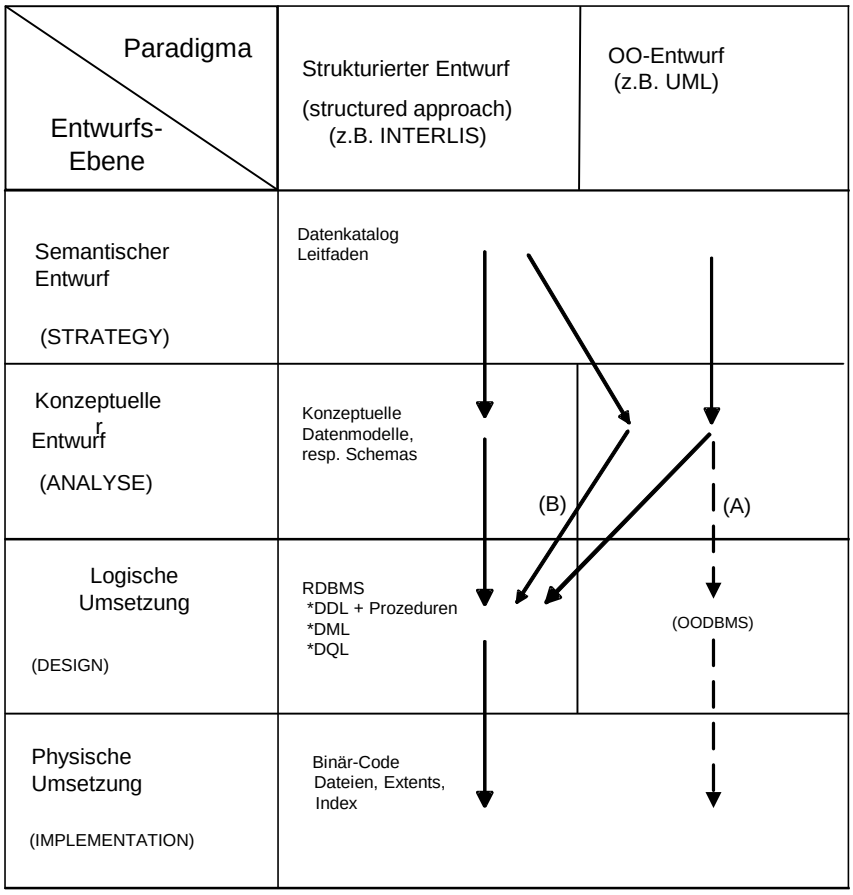
- Der **strukturierte Weg mit INTERLIS Version 1**: Dieser schweizerische Standard wird heute sowohl in der amtlichen Vermessung als auch in anderen Gebieten der raumbezogenen Daten verwendet. Er eignet sich, nach einigen notwendigen Ergänzungen, zur Beschreibung der Daten in der Strassendatenbank.
- Der **Objekt-orientierte Weg mit UML und INTERLIS Version 2**: UML entwickelt sich zur Zeit zu einem Weltstandard, ist aber erst ein De-Facto-Standard. Er ist von den internationalen Normierungsgremien noch nicht akzeptiert worden. UML wird in Zukunft zusammen mit INTERLIS 2 (und Folgende) eine schweizerische Standardlösung darstellen.

Die Autoren des Forschungsberichts schlagen für die Datenmodellierung der Strassendatenbank das folgende Vorgehen vor:

Für die reine Datenmodellierung, und als logische Fortsetzung der Datenkataloge in den VSS-Normen [SN640940 und Folgende], soll der **strukturierte Weg mit INTERLIS** eingeschlagen werden. Dieses Vorgehen ist auch kohärent mit dem SIA, der für die Informationen des Leitungskatasters ebenfalls diesen Weg eingeschlagen hat. **Damit sind alle Informationen im Strassenraum in derselben Art modelliert**. Dies ist für die gemeinsame Nutzung dieser Datenmodelle sicher von hohem Wert.

Im Anschluss an das vorliegende Forschungsvorhaben soll aber die "Objekt-orientierte" Modellierung des SMIS ebenfalls untersucht und dargestellt werden. Dies wird es erlauben, zu untersuchen, welche Probleme und Fragen zu bearbeiten sind, um den ganzheitlichen Entwurf mit dem Objekt-orientierten Paradigma durchzuführen.

Das folgende Schema zeigt mögliche Wege des Datenentwurfs. Damit soll auch das Interesse an einer Objekt-orientierten Modellierung begründet werden.



P:\9004\102KDM\RM-G810.dsf

Abbildung 8-1: Mögliche Wege des Datenentwurfs (Datenmodellierung)

Die Abbildung 8-1 beschreibt, wie die zwei Entwurfsparadigmen (strukturiert und "Objekt-orientiert") die Entwurfsebenen für die Datenmodellierung durchlaufen. Auf semantischer Ebene werden in beiden Entwurfsparadigmen ein Objekttypen- resp. ein Datenkatalog (oder ein Inventar) erstellt. Diese Kataloge werden jeweils nach den Regeln der Modellierung auf eine konzeptuelle Ebene gebracht. Im strukturierten Entwurf werden dann die auf konzeptueller Ebene beschriebenen Daten in einem gängigen Datenbankmanagementsystem DBMS logisch umgesetzt, beispielsweise in ein RDBMS (Relationales DBMS), wohl das am häufigsten genutzte DBMS überhaupt. Die auf konzeptueller Ebene gemäss OO entworfene Datenstruktur kann logisch auch in ein OODBMS umgesetzt werden (A). Da sich die OODBMS bis heute nicht durchgesetzt haben, werden die "OO-Datenstrukturen" häufig ebenfalls in RDBMS abgebildet (B).

Dieser Paradigmenwechsel findet bei der Umsetzung von Funktionen (d.h. von der Prozessbeschreibung zu einem Programm) nicht statt, denn die OO-Programmiersprachen, wie beispielsweise C++ oder Java, beinhalten bereits die Grundkonzepte der OO-Modellierung. Mit Hilfe dieser Grundkonzepte werden innerhalb der OO-Entwicklungsumgebung die Programme auf die physische Ebene (Binärcode, Dateien, usw.) abgebildet.

8.2 Der "strukturierte Weg": INTERLIS Version 1

8.2.1 Übersicht

Siehe Anhang D "INTERLIS – ein Austauschmechanismus".

Die Normierung der Fachdaten ist wohl die wichtigste Tätigkeit im Bereich des SMIS, weil die Integrität des nationalen Datenpools SMIS sicher gestellt werden muss, um Vergleichbarkeit, Kombinierbarkeit und Austauschbarkeit auf nationaler Ebene zu ermöglichen. Es ist weniger die Aufgabe der Normierungsgremien, standardisierte Funktionen (Auswertungen, Darstellungen) einzuführen. Die Normen müssen Datenstrukturen widerspruchsfrei und für Fachspezialisten verständlich darstellen. Die INTERLIS-Beschreibungssprache ist ein Mittel, dieses Ziel zu erreichen. Die in INTERLIS formulierte Datenstruktur sollte folgende Aufgaben erfüllen:

- Konzeptuelle Beschreibung der Datenstruktur für das SMIS
- Verbindung der Semantik mit bestehenden, in Datenbanken umgesetzten Datenstrukturen
- Information an Datenlieferanten: Daten so liefern, wie für die Integration in das SMIS nötig.
- Hilfe zu ersten Qualitätskontrollen: Plausibilität und Integritätskontrollen innerhalb eines Datensatzes des SMIS

INTERLIS ist in erster Linie eine Beschreibungssprache, die ein Datenmodell auf konzeptueller Ebene beschreiben kann. Obwohl die Version 2 einige Elemente der OO-Modellierung integriert hat, insbesondere die Vererbung, wird INTERLIS wohl nie die Gesamtheit der für die OO-Modellierung erforderlichen Modelle beschreiben können (Vergleiche 8.3.4). INTERLIS beschreibt ausschliesslich Daten und nicht Aktivitäten oder Organisationen.

In den bisher konzipierten Informationssystemen im Bereich des Managements der Strassenverkehrsanlagen wurde das Gewicht vor allem auf den Entwurf der Datenstrukturen gelegt. Dies ist einerseits begründet durch ihre Langlebigkeit im Vergleich zu den Funktionen und andererseits durch das kostspielige Erheben der Daten. Der Entwurf von Datenstrukturen gewinnt noch mehr an Bedeutung, wenn eine Normierung eines Datenkataloges angestrebt wird. Normen sollen ja so langlebig wie nur möglich sein. Organisatorische Elemente und Funktionen werden in der Regel wegen ihrer bescheidenen Lebensdauer nicht normiert.

Für die Beschreibung eines Daten-Entwurfs genügt INTERLIS vollumfänglich. EXPRESS wurde absichtlich nicht als Beschreibungssprache gewählt, weil sie erstens in der Schweiz kaum angewendet wird (INTERLIS wird als Standard eingesetzt) und zweitens die Lesbarkeit einer solchen Beschreibung doch schwierig ist. Die Lesbarkeit einer Beschreibung durch Anwender ist für die Kommunikationsbedürfnisse im Strassenbereich ein wichtiges Kriterium.

8.2.2 Erläuterung eines Beispiels

a) Beispiel

```
TABLE Cross_Section_Usages =
    !! ( D:‘Fahrbahn-Nutzung‘ F:‘Usage de la chaussée‘, SN640942/GEN-SN640940)
      CSU_BaseID:                       BASEID                  !! PK             SN-GEN-I1
      CSU_Version:                      VERSION                 !! PK             SN-GEN-I2
      CSU_AXE_CKO_Owner:                OWNER       //Kernel//; !! UKC(1)->AXE    SN-I1.1
      CSU_AXE_CK:                       CK3         //Kernel//; !! UKC(1)->AXE    SN-I1.2
      CSU_AXE_POS_Code:                 POSCODE     //Kernel//; !! UKC(1)->AXE    SN-I1.3
      CSU_AXE_VRS_Code:                 VERSCODE    //Kernel//; !! UKC(1)->AXE    SN-I1.4
      CSU_RPT_BaseID_1:                 BASEID      //Kernel//; !! FKL(2)->RPT    SN-I1.10
      CSU_RPT_CK_1:                     CK2         //Kernel//; !! UKC(2)->RPT    SN-I1.11
      CSU_RPT_VRS_Code_1:               VERSCODE    //Kernel//; !! UKC(2)->RPT    SN-I1.12
      CSU_RPDistBegin:                  U_ABS       //Kernel//; !! UKC            SN-I1.21
      CSU_RPT_BaseID_2:                 BASEID      //Kernel//; !! FKL(3)->RPT    SN-I2.0
      CSU_RPT_CK_2:                     CK2         //Kernel//; !! UKC(3)->RPT    SN-I2.1
      CSU_RPT_VRS_Code_2:               VERSCODE    //Kernel//; !! UKC(3)->RPT    SN-I2.2
      CSU_RPDistEnd:                    U_ABS       //Kernel//; !! UKC            SN-I2.11
      CSU_PosBegin:                     NUM2_2      //Kernel//; !! UKC            SN-I3
      CSU_PosEnd:                       NUM2_2      //Kernel//; !! UKC            SN-I4
      CSU_POS_Code:                     POSCODE     //Kernel//; !!                SN-A1
      CSU_Width:                        NUM2_2      //Kernel//; !!                SN-A2
      CSU_SCA_UST_CTK_BaseID:           BASEID      //Kernel//; !! FKL(4)->SCA    SN-A3.0
      CSU_SCA_UST_CTK_CKO_Owner:        OWNER       //Kernel//; !! FKC(4)->SCA    SN-A3.1
      CSU_SCA_UST_CTT_LNG_Code:         LANGCODE    //Kernel//; !! FKC(4)->SCA    SN-A3.2
      CSU_SCA_UST_CTT_TextID:           TEXTID      //Kernel//; !! FKC(4)->SCA    SN-A3.3
      CSU_TypAdText:       OPTIONAL     ADTEXT      //Kernel//; !!                SN-A4
      CSU_PRO_BaseID:      OPTIONAL     BASEID      //Kernel//; !! FKL(5)->PRO    SN-A5.0
      CSU_PRO_CKO_Owner:   OPTIONAL     OWNER       //Kernel//; !! FKC(5)->PRO    SN-A5.1
      CSU_PRO_CK:          OPTIONAL     CK4         //Kernel//; !! FKC(5)->PRO    SN-A5.2
      CSU_PRO_VRS_Code:    OPTIONAL     VERSCODE    //Kernel//; !! FKC(5)->PRO    SN-A5.3
      CSU_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
      CSU_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
      CSU_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
      CSU_EndValidity:     OPTIONAL     DATEDOM     //Kernel//; !!                SN-GEN-D2
      CSU_Comment:         OPTIONAL     COMMENT     //Kernel//;
      CSU_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
      CSU_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
      CSU_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
      CSU_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
      CSU_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
      CSU_ChangeDate:      OPTIONAL     DATEDOM;                !!                SN-GEN-A9
      CSU_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
      CSU_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
      CSU_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
      CSU_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
      CSU_XST_XfrSetID:    OPTIONAL     BASEID;                 !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
         CSU_BaseID, CSU_Version; !! PK
         CSU_AXE_CKO_Owner, CSU_AXE_CK, CSU_AXE_POS_Code, CSU_AXE_VRS_Code,
         CSU_RPT_CK_1, CSU_RPT_VRS_Code_1, CSU_RPDistBegin,
         CSU_RPT_CK_2, CSU_RPT_VRS_Code_2, CSU_RPDistEnd, CSU_PosBegin, CSU_PosEnd; !! UKC
    END Cross_Sect_Usage;
```

b) Schlüsselwörter und Symbole

Das in a) aufgeführte Beispiel, das die Struktur der Fahrbahnnutzung darstellt, enthält einige Schlüsselwörter der INTERLIS Beschreibungssprache (IDDL: INTERLIS Data Description Language):

- TABLE : Anfang einer Beschreibung eines Informationsobjekttyps
- OPTIONAL: Das mit diesem Schlüsselwort bezeichnete Attribut muss nicht unbedingt einen Wert enthalten
- CONSTRAINT: Anfang der Beschreibung der Konsistenzbedingungen eines Informationsobjekts
- UNIQUE: Konsistenzbedingung der Eindeutigkeit (eines Schlüssels) oder einer anderen Attributkombination.
- END: Ende einer Beschreibung eines Informationsobjekttyps

Zudem definiert IDDL folgende Symbole:

- //……// : Erläuterung, eine formalisierte oder nicht formalisierte Konsistenzbedingung, die nur ein Attrbut betrifft.
- !!: Alle nach diesem Symbol auf gleicher Zeile erfolgten Beschreibungen werden als Kommentar angesehen.

Für die Beschreibung der Strassendaten wurden folgende Abkürzungen hinzugefügt:

- PK: Primary Key, der Systemschlüssel
- UKC: Unique Key (conceptual), der konzeptuelle, sprechende Schlüssel
- FKL: Foreign Key (logic): der logische Fremdschlüssel
- FKC: Foreing Key (conceptual), der konzeptuelle, sprechende Fremdschlüssel
- //Kernel//: bezeichnet alle Attribute, die von einem Datenlieferant berücksichtigt werden müssen
- SN: Referenz zur Schweizerischen Norm
- SN-GEN: Referenz zu den generischen Attributen der Schweizerischen Normenvereinigung

c) Erklärungen

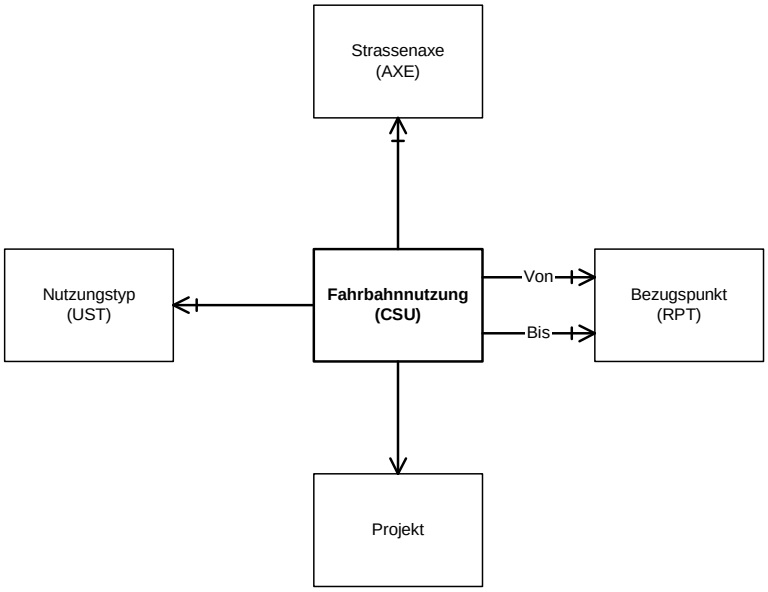


Abbildung 8-2: Darstellung der Beziehungen der Informationsobjekte "Fahrbahnnutzung" zu anderen Informationsobjekten

Ein Informationsobjekttyp wird zuerst mit einem eindeutigen Namen bezeichnet, hier mit Cross_Section_Usages (CSU). Andere Namen in anderen Sprachen wurden als Kommentar in die Beschreibung integriert. Zudem werden die Normen SN 640'942 für die spezifischen Attribute und die Norm SN 640'940 für die generischen Attribute angegeben. Falls der Benutzer, der eine INTERLIS-Beschreibung konsultiert, die Semantik des Informationsobjekts oder der Attribute nicht kennt, kann er die entsprechende Definitionen in den Normen nachschlagen.

Darunter werden die Attribute aufgelistet. Pro Attribut werden folgende Daten angegeben:

- Name: z.B CSU_BaseID
- OPTIONAL oder nicht
- Wertebereich, z.B BASEID
- //Kernel//
- !! (Kommentar)
- Spezielle Kennung, z.B PK für primary key
- Abkürzung des referenzierten Informationsobjekttyps, z.B AXE
- Referenz zu einem in der Norm beschriebenen Attribut, z.B. SN-GEN-I1

Zuletzt werden die Attribute bezeichnet, die den Primary Key oder den Unique Key bilden, um eine Eindeutigkeit überprüfen zu können.

d) Für SMIS notwendige Änderungen gegenüber der Beschreibungssprache INTERLIS Version 1

Schichten

INTERLIS ermöglicht durch sogenannte Topics, die im GIS-Bereich den Layers oder Schichten entsprechen, die Strukturierung des Datenmodells. Diese Topics müssen von einander unabhängig sein. In der Beschreibung der Datenstruktur wurden bei INTERLIS alle Informationsobjekte in ein Topic integriert, weil die Bedingung nach Unabhängigkeit der identifizierten Themen sonst nicht eingehalten werden konnte.

Systemschlüssel

In einer INTERLIS-Beschreibung werden die Systemschlüssel nicht angegeben, weil nach Ansicht der INTERLIS-Konzipierer diese Schlüssel vor dem Benutzer versteckt gehalten werden sollten. Eine solche Beschreibung ist nicht vollständig. Ein Benutzer, der in seinem System Daten erfasst, wüsste die Struktur des erwarteten Systemschlüssels nicht. Für die in der Schweiz eingesetzten Strassendatenbanken werden die Systemschlüssel aus der Datenbank-Kennung, einer Nummer und der Versionsnummer zusammengesetzt.

Fremdbeziehungen

Die Fremdbeziehung wird mit Hilfe eines in der Schweiz eindeutigen sprechenden Schlüssels oder mit einem eindeutigen Systemschlüssel gebildet. In der standardisierten INTERLIS-Beschreibung wird nur das Informationsobjekt referenziert ohne den sprechenden Schlüssel anzugeben. Um die Lesbarkeit der Daten und die Anforderungen an die Datenerheber besser formulieren zu können, wurde die Referenzierung durch den sprechenden Schlüssel erlaubt und auch beschrieben.

INTERLIS sollte beide Möglichkeiten anbieten können, die Fremdbeziehungen mit dem logischen Schlüssel oder mit dem sprechenden Schlüssel zu beschreiben.

Räumliche Referenzierung

Die räumliche Referenzierung wird in INTERLIS durch X-Y (Z) Koordinaten sichergestellt. Diese Referenzierungsart ist für die meisten Fachprozesse im Strassenunterhalt ungeeignet. Sie wird durch die Referenzierung mit Hilfe von Distanzen entlang der Strassenaxe und seitlichen Abständen ersetzt. Somit werden diese zur Referenzung benötigten Daten wie normale Attribute beschrieben. GIS-Softwarehersteller bieten einen ersten Ansatz zu den linearen Bezugssystemen, indem sie die "lineare Segmentierung" eingeführt haben. INTERLIS sollte diese Entwicklung miteinbeziehen.

//Kernel//

Attribute, die für den Datenlieferant wesentlich sind, werden mit dem Kennwort //Kernel// beschriftet. Das Kennwort OPTIONAL bezeichnet diejenigen Attribute, die für das Informationssystem als Ganzes nicht unbedingt einen Wert verlangen. Attribute, die nicht OPTIONAL sind und die sich nicht im Kernel befinden, müssen durch das System nachgeführt werden.

8.3 Der Objekt-orientierte Weg: UML und INTERLIS 2

8.3.1 Übersicht

Die Entwurfs- und Umsetzungs-Sprachen sind einem ständigen Wandel unterworfen. Sie folgen einerseits den Entwurfs- und Umsetzungs-Paradigmen. Andererseits werden sie auch durch die Möglichkeit der automatischen Generierung von Datenbanken und Applikationen aus der Beschreibungssprache heraus beeinflusst.

Zur Zeit steht vor allem ein Paradigma im Fokus der Entwicklung und wird die Zukunft zumindest mitbestimmen: die Objekt-Orientierung (Siehe Ziffer 6).

Im Zuge der vermehrten Nutzung des Objekt-orientierten Paradigmas werden auch entsprechende Beschreibungssprachen vermehrt Verwendung finden (Siehe Ziffer 7).

UML zum Beispiel wird zur Zeit weiterentwickelt und mit einer Entwurfs- und Umsetzungsmethode ergänzt (Rational Unified Process, Anhang H).

8.3.2 UML

UML eignet sich sehr gut für den Entwurf einer Applikation. Sie ermöglicht die Kommunikation innerhalb eines Projektteams. UML ist aber weniger geeignet, um Benutzer oder Fachspezialisten über den Stand eines IT-Projektes zu informieren; dies, da gewisse Kenntnisse über die etwas komplexe OO-Datenmodellierung und über die UML-Notation auch vorhanden sein müssen, um UML-Diagramme interpretieren zu können.

Um jedoch die Fachprozesse mit dem Benutzer definieren und strukturieren zu können, kann die UML-Notation (Use-Case Diagramme) unabhängig vom Modellierungsparadigma (OO oder strukturiert) benutzt werden.

Beispielsweise wurden die aktuellen Arbeiten im Rahmen von STRADA-View/ Axband von der Modellierung bis hin zur Entwicklung unter dem OO-Paradigma durchgeführt. Darin spielt UML eine zentrale Rolle. Es hat sich gezeigt, dass vor allem das Klassen- und das Sequenzdiagramm wesentlich und hilfreich sind.

Die zentrale Eigenschaft der Wiederverwendbarkeit oder "reusability" von UML bewirkt, dass der Code derart strukturiert wird, dass er vielfältig einsetzbar wird. UML spornt die Softwareentwickler somit an, generische Programme zu erstellen. Deren Erstellung wird eher teurer sein, als herkömmlicher spezifischer Code. Durch die mehrfache Verwendbarkeit wird dieses Vorgehen auf lange Sicht dennoch wirtschaftlicher sein.

UML beschreibt jedoch nicht die Datenstruktur einer Datenbank. Für die Konzipierung einer Datenbank sind die herkömmlichen Methoden und Beschreibungssprachen anzuwenden.

8.3.3 INTERLIS Version 2

INTERLIS 2 wurde ein erstes Mal 1998 publiziert. Im Jahre 2000 wurde eine neue Version des Handbuchs von der Eidgenössischen Vermessungsdirektion herausgegeben, dass wesentliche Änderungen bezüglich der 1998 herausgegebenen Version enthält. INTERLIS 2 ist eine Sprache, die die Grundkonzepte der Objekt-orientierten Datenmodellierung wiedergeben kann. INTERLIS 2 liegt noch nicht in Form einer Norm vor. Es werden zur Zeit Anstrengungen unternommen, um sie auf internationaler Ebene bei der ISO TC 211 durchzusetzten, da sie die einzige Sprache ist, die die konzeptuelle Datenbeschreibung zum Ziel hat. Da INTERLIS 2 noch kein Standard ist, wird sie sicherlich noch einige Änderungen erfahren (z B in der Formalisierung von Methoden). Es ist zusätzlich notwendig, dass Schema-externe Konsistenzbedingungen formal beschrieben werden können. OCL (Object Constraint Language) stellt eine solche formale Sprache für Konsistenzbedingungen dar, die in UML genutzt wird.

8.3.4 Beispiele

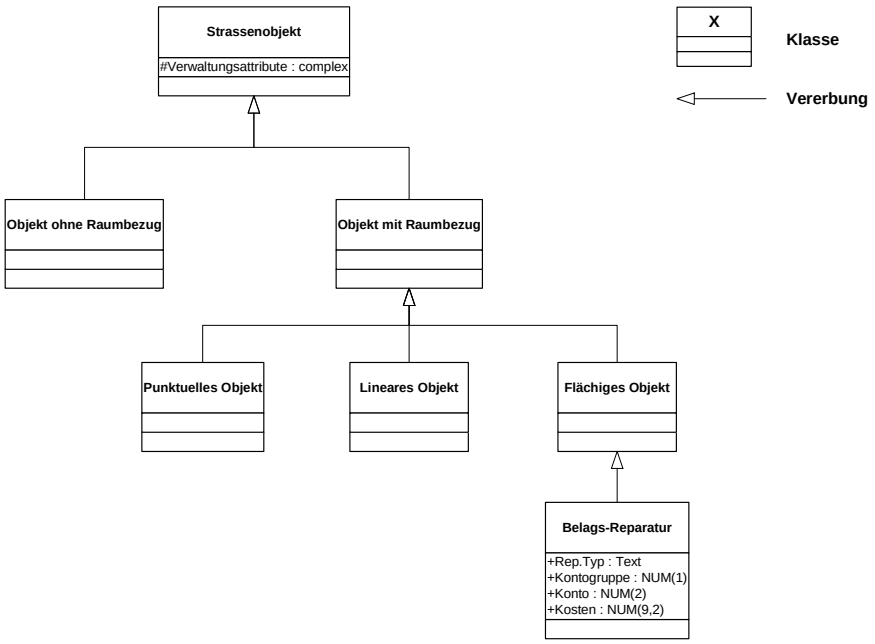


Figure 8-1 : Beispiel einer Vererbung im UML-Klassendiagramm

Das Beispiel zeigt eine Vererbung in einem UML-Klassendiagramm. Alle Objekte sind Strassenobjekte. Diese Strassenobjekte können einen oder keinen Raumbezug haben. Die Objekte mit Raumbezug sind entweder punktuelle, lineare oder flächige Objekte. Als Beispiel eines flächigen Objekts wird hier die Belagsreparatur gezeigt. Attribute, wie beispielsweise die Verwaltungsattribute, die für jedes Strassenobjekts gültig sind, können für die Klasse "Strassenobjekt" definiert werden. Diese Eigenschaften werden auf die Objekte der entsprechenden Unterklassen vererbt, d.h. der Fahrbahnaufbau erbt beispielsweise die Eigenschaften eines Strassenobjekts. Er ist ja eine Ausprägung eines Strassenobjekts. In der Objekt-orientierten Modellierung können Methoden oder Funktionen an Klassen geheftet werden. Die Funktion "stelle als Polygon dar" beispielsweise kann an die Klasse "Flächiges Objekt" integriert werden und sie kann automatisch durch die Vererbung für die Fahrbahnaufbau genutzt werden. Der Einfachheit halber wurden keine Methoden oder Funktionen dargestellt.

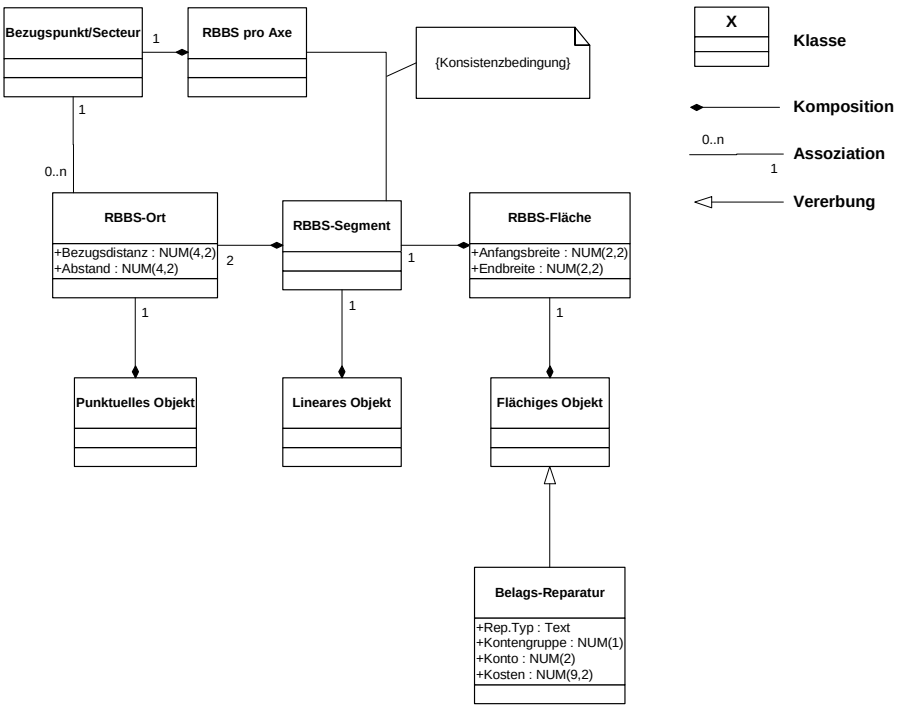


Figure 8-2: UML-Klassendiagramm des Raumbezugs

In der Abbildung 8-3 wird der Raumbezug dargestellt. Das RBBS wird bekanntlich aus Axe und einer Menge von Bezugspunkten dargestellt. Es kann durch eine Komposition von Objekten der Klasse Bezugspunkt in der OO-Modellierung dargestellt werden. Ein RBBS-Ort bezieht sich auf genau einen Bezugspunkt (also Assoziation in der OO-Modellierung), und enthält eine Bezugsdistanz und einen Abstand. Ein RBBS-Segment wird von zwei RBBS-Orten abgegrenzt und die RBBS-Fläche ergänzt ein RBBS-Segment durch eine Anfangs- und eine Endbreite. Der Ort eines punktuelles Objekts wird immer durch einen RBBS-Ort beschrieben, das lineare Objekt immer durch ein RBBS-Segment und das flächige Objekt durch eine RBBS-Fläche. RBBS-Ort, RBBS-Segment und RBBS-Fläche sind nicht Klassen mit eigenständigen Objekten, d.h ein RBBS-Ort kann nicht existieren, wenn nicht ein entsprechendes punktuelles Objekt oder RBBS-Segment existiert. Diese Unselbstständigkeit wird durch die Komposition abgebildet. Der Benutzer wird somit nie einen RBBS-Ort ohne entsprechenden Kontext manipulieren können, d.h. er kann ein RBBS-Ort verändern, wenn er von einem punktuellen Objekt aus oder von einem linearen bzw. flächigen Objekt über RBBS-Segment bzw. RBBS-Fläche auf ihn zugreifen kann. Es gilt zu erwähnen, dass beispielsweise ein RBBS Segment aus zwei RBBS-Orten besteht (also Komposition). Diese Beziehung kann nicht als Vererbung dargestellt werden, weil ein RBBS-Segment semantisch nicht die gleiche Bedeutung hat wie zwei RBBS-Orte. Eine Agregation bzw. Komposition kann durch das Verb "haben" qualifiziert werden, während dem die Vererbung mit dem Verb "sein" qualifiziert werden kann. Ein RBBS-Segment "hat" zwei RBBS-Orte und nicht RBBS-Segment "ist" zwei RBBS-Orte.

Das UML-Beispiel wird wie folgt in die INTERLIS 2 Beschreibungssprache umgesetzt:

```
TRANSFER Roads;

MODEL STRADA =
…
TOPIC STRADA/UR-A

  CLASS Strassenobjekt =                    !! Verwaltungsattribute
      ODO_Owner:     MANDATORY      CKO;
      OrigSubDBID:   MANDATORY      TEXT*4;
      DAO_Owner:     MANDATORY      CKO;
      DataOwner:     MANDATORY      CKO;
      CreateDate:    MANDATORY      DATUM;
      ChangeDate:                   DATUM;
      CHO_Owner:     MANDATORY      CKO;
      ChangeUser:    MANDATORY      TEXT*32;
      ITS_Code:      MANDATORY      TEXT*2;
      IntegrityDate: MANDATORY      DATUM;
      XST_XferSetID:                TEXT*20;
  END Strassenobjekt;

  CLASS Objekt_ohne_Raumbezug EXTENDS Strassenobjekt =
  END Objekt_ohne_Raumbezug;

  CLASS RBBS_pro_Axe EXTENDS Objekt_ohne_Raumbezug =
    Sektoren:        MANDATORY        LIST{2..*} OF Bezugspunkt/Sektor; !! Komposition
  End RBBS_pro_Axe;

  CLASS Bezugspunkt/Sektor EXTENDS Objekt_ohne_Raumbezug =
    SektorLänge:     MANDATORY        DIM4_2;
  End Bezugspunkt/Sektor;

  CLASS Objekt_mit_Raumbezug EXTENDS Strassenobjekt =
  End Objekt_mit_Raumbezug;

  STRUCTURE RBBS_Ort =
    Bezugspunkt:     MANDATORY        -> Bezugspunkt/Sektor; !! Assoziation
    Bezugsdistanz:   MANDATORY        DIM4_2;
    Abstand:         MANDATORY        DIM4_2;
  MANDATORY CONSTRAINT
    Bezugsdistanz < Bezugspunkt.SektorLänge;
  END RBBS_Fläche;

  STRUCTURE RBBS_Segment =
    Axe:             MANDATORY        -> Axe;
    RBBS-Ort:        MANDATORY        LIST{2..2} OF RBBS_Ort;  !! Komposition von 2 Elementen
  MANDATORY CONSTRAINT
    Axis == RBBS-Ort.Axis;
  END RBBS_Fläche;

  STRUCTURE RBBS_Fläche =
    RBBS-Bezug:      MANDATORY        OBJECT OF RBBS_Segment;
    Anfangsbreite:   MANDATORY        DIM2_2;
    Endbreite:       MANDATORY        DIM2_2;
  END RBBS_Fläche;

  STRUCTURE Flächiges_Objekt EXTENDS Objekt_mit_Raumbezug =
    Raumbezug                         -> RBBS_Fläche;
  END Flächiges_Objekt;

  CLASS Belags-Reperatur EXTENDS Flächiges_Objekt =
    Reparaturtyp:    MANDATORY      TEXT*72;
    Kontengruppe:    MANDATORY      DIM1;
    Konto:           MANDATORY      DIM2;
    Kosten:                         DIM9_2;
  END Fahrbahnaufbau;

END STRADA/UR-A;
END STRADA;
END Roads.
```

Eine "INTERLIS 2"-Beschreibung respektiert i.a. die im "INTERLIS 1"-Handbuch beschriebene Struktur. Sie wird des weiteren nicht weiter erläutert. Augenfällig sind folgende Schlüsselwörter:

- Class: selbstständige Klassen
- Structure: unselbstständige Klassen deren Objekte nicht einzeln identifiziert werden können. Sie kommen innerhalb von Kompositionen vor oder als Klasse deren Objekte nur temporär existieren.

Assoziations-Beziehungen werden mit einem Pfeil "->" dargestellt, wie beispielsweise die Beziehung von RBBS-Segment auf die Axe. Aggregations-Beziehungen werden mit einem kleinen Rhombus "<>-" dargestellt, wie beispielsweise die Beziehung vom Abschnitt zum Knoten (nicht dargestellt im oben aufgeführten Schema). Kompositionsbeziehungen werden mit Hilfe der Kennwörter LIST (geordnete Menge), BAG (ungeordnete Menge) und OBJECT (Menge mit einem Element) dargestellt. Kompositionen innerhalb einer Klasse können nur auf Strukturen aufgebaut werden.

Konsistenzbedingungen werden mit den Kennwort CONSTRAINT eingeführt. Mit MANDATORY CONSTRAINTS werden alle Konsistenzbedingungen bezeichnet, die obligatorisch von allen Objekten der entsprechenden Klasse erfüllt werden müssen. Fakultative Konsistenzbedingungen können mit dem Prozentsatz der Objekte, die diese Konsistenzbedingung erfüllen, ergänzt werden.

8.3.5 Die Zukunft

Die konzeptuelle Modellierung von Strassendatenbanken (mit Normen) wird auch in Zukunft nach den Regeln des strukturierten Entwurfs erfolgen, da eine systematische Analyse für den Entwurf von langlebigen Datenstrukturen zwingend notwendig ist.

Das Vermeiden einer Fehlentwicklung eines IT-Projektes führt aber zusätzlich dazu, dem Benutzer ausführbare Applikationen in kleinen, jeweils Nutzen bringenden Schritten (Modulen) übergeben zu wollen. Jedes Modul muss rasch entworfen und umgesetzt werden. Dazu ist eine iterative Entwurfsmethode Voraussetzung. Der OO-Entwurf mit UML eignet sich für dieses Vorgehen. Voraussetzung ist aber eine konzeptionell saubere Gesamtarchitektur, sowohl bezüglich der Daten als auch bezüglich funktionaler Aspekte. UML wird darum vor allem in der Softwareentwicklung für das SMIS Anwendung finden

In diesem Zusammenhang muss darauf hingewiesen werden, dass mit Hilfe von OO-Konzepten realisierte Software auch in Zukunft oft auf relationale Datenbanken zugreifen muss. Es muss also eine Konvertiersoftware zwischen die OO-Software und die relationale Datenbank eingefügt werden. Um in Zukunft eine homogenere Integration zwischen Datenbank und Anwendungssoftware zu ermöglichen, müssen sich die Datenbankverwaltungssysyteme in Richtung der Objektorientierung entwickeln. Sowohl die Datenstrukturen als auch die Datendefinitions- und manipulationssprachen müssen die objektorientierten Konzepte umsetzen können. Solche Arbeiten sind im Gange, aber noch nicht abgeschlossen.

Die genaue Abgrenzung von strukturiertem und objektobjektorientiertem Entwurf für das Informationssystems SMIS muss in einem Folgeprojekt untersucht werden. Zudem soll dieses Projekt die Entwurfsmethoden mit den entsprechenden Notationen nicht nur auf die Datenmodellierung beschränken, sondern zumindest auf Funktionen erweitern. INTERLIS 2 wird sich sicherlich noch entwickeln und auch diese Resultate sollen im Folgeprojekt untersucht werden. Der vorliegende Bericht beschreibt die Modellierung aus der Sicht der Normierung; im Folgeprojekt soll die Sicht der Informationssystem-Entwicklung mehr zum Tragen kommen. Ein Beispiel einer komplette OO-Modellierung eines Verwaltungsprozesses des SMIS und eines Fachprozesses aus dem MSE sollen zur Illustration mit UML und, sofern möglich, mit INTERLIS 2 dargestellt werden. Zudem soll die Entwicklung im Bereich von XML einbezogen werden. Dies gilt sowohl für den Datenaustausch als auch für die Datenablage.

* * *

A Abbildungsverzeichnis

B Bibliographie

TC287/1996 CEN/TC287; 1996; Geographic Information, Data Desription, Conceptual Schema, Language.

ISO TC 211 WG1 – WI3 (1996): Conceptual Schema Language.

INTERLIS/1998 EJPD/BRP/EVD; 1998; INTERLIS: Ein Datenmodellierungs- und Austauschmechanismus für Geo-Informationssysteme

INTERLIS/1997 Germann/Dorfschmid; 1997; Inkrementelle Nachlieferung mit INTERLIS

Projektleitung RAV (1991): INTERLIS, ein Daten-Austausch-Mechanismus für Land-Informations-Systeme.

Bundesamt für Raumplanung, Eidgenössische Vermessungsdirektion, Kompetenzzentrum INTERLIS / AVS (1998): INTERLIS, ein Daten-Austausch-Mechanismus für Land-Informations-Systeme; Version 2; Revision 1.

Bundesamt für Raumplanung, Eidgenössische Vermessungsdirektion, Kompetenzzentrum INTERLIS / AVS (1998): Hinweise zu Draft INTERLIS Version 2; Version 2, Revision 1.

Bundesamt für Landestopographie, Eidgenössische Vermessungsdirektion (2000): INTERLIS Version 2.0, Referenzhandbuch. Ausgabe vom 2000-04-18 (deutsch).

SIA405 Norm SIA 405 inkl. Merkblätter 1015 (Informationsbeschreibung) und 1016 (Datenaustausch für Leitungsinformationen)

SN640900 Norm VSS/SN 640'900; 1989; Management der Strassenerhaltung, Grundsätze

SN640901 Norm VSS/SN 640'901; 1990; Zielsystem (für das MSE)

SN640909 Norm VSS/SN 640'909; 1990; Strassendatenbanken, Grundlagen

SN640940 Norm VSS/SN 640'940, 1993; Katalog für Strassendaten, Grundsätze

SN640941 Norm VSS/SN 640'941; 1993; Katalog für Strassendaten, Raumbezug

Forschungsbericht Nr. 191 (1988): Die Datenintegrität bei Strassendatenbanken im Hinblick auf den Datenaustausch. Im Auftrag der Vereinigung Schweizerischer Strassenfachleute (Forschungsauftrag 15/88).

Forschungsbericht Nr. 295 (1994): STRADA-DB: Strassendatenbank für das MSE, Leitfaden für die Einführung und den Betrieb. Im Auftrag der Vereinigung Schweizerischer Strassenfachleute (Forschungsauftrag 03/91).

http://wwwis.cs.utwente.nl:8080/dmrg/MEE97/misop032/niam.htm

Bröhl, A.-P und Dröschel W. (1993): Das V-Modell : der Standard für die Softwareentwicklung mit Praxisleitfaden. Oldenburg.

Bundesamt für Informatik (1995): HERMES, conduite de déroulement de projets informatiques, OCFIM; Berne.

Booch Grady, Rumbaugh James, Jacobson Ivar (1999): The Unified Modeling Language; User's Guide. Object Technology Series; Addison Wesley; Massachusetts, USA.

Kruchten Philippe (1999): The Rational Unified Process, An Introduction. Addison Wesley Longman Inc.; Massachusetts; USA.

Kahlbrandt, Bernd (1998): Software-Engineering, Objektorientierte Software-Entwicklung mit der Unified Modeling Language; Springer Verlag, Berlin, Heidelberg.

C Glossar

1. Informationssysteme, Datenbanken:

D:	**Information**	Eine Aussage über einen Teil der Realität als Grundlage für Kommunikation und Dokumentation von menschlichem Wissen. Information kombiniert Daten und ihre Bedeutung. (Wissen kombiniert Information und ihren Kontext.)
F:	**Information**	
E:	**Information**	
D:	**Daten**	Strukturierte Elemente zur Abbildung von Information, z. B. in einer Datenbank.
F:	**Données**	
E:	**Data**	
D:	**Informationssystem**	Ein System, meist informatisiert, zur Speicherung, Verwaltung und Nutzung von Information, hat insbesondere die Fähigkeit, durch entsprechende Regeln aus Daten wieder nutzbare Information zu kombinieren.
F:	**Système d'information**	
E:	**Information System**	
D:	**Datenbank (physisch)**	Kernelement eines Informationssystems, dient zur Speicherung, Verwaltung und Nutzung der Daten; physikalische Einheit.
F:	**Banque de données**	
E:	**Database**	
D:	**Datenbank (konzeptionell)**	Konzeptionelle Gesamtheit aller Datenbanken, die der Verwaltung der Daten eines Informationssystems dienen.
F:	**Base de données**	
E:	**Database**	
	Data Warehouse	Eine Datenbank und zugehörige Applikationen, die (nur) lesend auf Daten verschiedener Herkunft mit unterschiedlicher Abstraktion zugreifen können.

2. Verarbeitungsfunktionen:

D:	**Aggregation**	Die Übertragung punktueller Messungen einer Eigenschaft auf ein flächiges oder lineares Objekt (z. B die Umwandlung punktueller Ebenheitsmessungen in eine repräsentative Ebenheit eines Fahrbahnobjekts).
F:	**Agrégation**	
E:	**Aggregation**	
D:	**Einfache Abfrage**	Eine Abfrage aller Informationsobjekte eines Typen, die bestimmte Kriterien erfüllen.
F:	**Traitement simple**	
E:	**Simple Querying**	
D:	**Kombinierte Auswertungen**	Eine Abfrage, resp. Auswertung mit Kombination verschiedener Informationsobjekttypen, mit sachlichem, zeitlichem und räumlichem "Verschneiden".
F:	**Traitement combiné**	
E:	**Combined Querying**	
D:	**Kombinierbarkeit**	Die Möglichkeit, über die nötige Kompatibilität, Daten verschiedener Quellen oder Objekte zu vereinigen, um Informationen ableiten zu können.
F:	**Combinaison**	
E:	**Combination**	

3. Qualitätskontrolle einer Datenbank:

D:	**Integrität (Daten)**	Die Daten-Integrität umfasst die Merkmale Konsistenz (Widerspruchsfreiheit), Vollständigkeit (der Aussage), Datensicherheit und Datenschutz.
F:	**Intégrité (données)**	
E:	**Integrity (Data)**	
D:	**Qualität (Daten)**	Die Qualität der Daten wird durch das Bedürfnis nach Integrität, Vollständigkeit (quantitativ), Genauigkeit, Präzision und Aktualität innerhalb eines Fachprozesses bestimmt.
F:	**Qualité (données)**	
E:	**Quality (Data)**	
D:	**Metadaten**	Daten über Daten; Daten, die den Benutzer z.B. über die Qualität der verfügbaren Daten unterrichten.
F:	**Méta-données**	
E:	**Meta data**	
D:	**Semantische Konsistenz**	Die Qualität der Daten bezüglich ihrer Vollständigkeit und ihrer Widerspruchsfreiheit. Es muss sicher gestellt werden, dass der Datenlieferant die semantische Objekt- bzw. Attributfestlegungen richtig interpretiert hat.
F:	**Cohérence sémantique**	
E:	**Semantic consistency**	
D:	**Vollständigkeit**	Der Prozentsatz der gelieferten Daten bezüglich der verlangten Daten - kann ein Mass der Möglichkeit der Bildung von Information aus Daten sein.
F:	**Complétude**	
E:	**Completeness**	
D:	**Plausibilität**	Nachvollziehbarkeit und Erklärbarkeit in einem gegebenen Kontext.
F:	**Plausibilité**	
E:	**Plausibility**	
D:	**Strukurelle Integrität**	Strukturelle Übereinstimmung der gelieferten Daten mit dem konzeptuellen Datenmodell, resp. Datenschema.
F:	**Intégrité structurelle**	
E:	**Structural integrity**	
D:	**Redundanz**	Mehrfaches Vorkommen derselben Daten oder Information.
F:	**Rédondance**	
E:	**Redondancy**	

4. Entwerfen eines Informationssystems

D:	**Entwerfen**	Identifizieren und Strukturieren der für das Informationssystem relevanten Daten und Funktionen aus dem Fachbereich.
F:	**Modéliser**	
E:	**Modeling**	
D:	**Funktionen-Hierarchie**	Eine hierarchische Strukturierung der Funktionen, als Teil des klassischen strukturierten Entwurfs.
F:	**Hiérarchie de fonctions**	
E:	**Function hierarchy**	

D:	**Konzeptionell**	Nicht technisch, ohne Details. Eine systemische Analyse kann als konzeptionell bezeichnet werden, weil funktionale Einheiten und deren Interaktionen analysiert werden, ohne genau zu wissen, wie die Einheiten funktionieren. Beim Datenentwurf wird auf deutsch auch der Begriff "konzeptuell" verwendet.
F:	**Au niveau du concept**	
E:	**At a conceptual level**	
D:	**Paradigmen**	Die Gesamtheit aller wesentlichen, elementaren und unbeweisbaren Rahmenbedingungen, auf die sich eine Theorie (eine Philosophie oder ein Glaube) stützt, z.B : Axiome für die Mathematik. Ein Paradigmenwechsel führt unvermeidlich zu einer anderen Theorie.
F:	**Paradigme**	
E:	**Paradigm**	
D:	**Entwurfs- und Umsetzungsparadigmen**	Der strukturierte Entwurf und der Objekt-orientierte Entwurf als Vorgehens-Paradigmen.
F:	**Paradigme de modélisation et de mise en oeuvre**	
E:	**Modeling and implementation paradigms**	
D:	**Strukturierter Entwurf**	Ein Entwurfs- und Umsetzungs-Paradigma, das auf dem getrennten Entwurf von Daten, Funktionen und Organisation beruht.
F:	**Conception structurée**	
E:	**Structured conception**	
D:	**OO-Entwurf**	Ein Entwurfs- und Umsetzungs-Paradigma; Datenbankentwurf, der die Grundkonzepte der Objekt-orientierten Programmierung für die Modellierung benutzt, wie Vererbung, Polymorphismus etc.; vereinigt Daten und Funktionen (Methoden) in Klassen.
F:	**Conception OO**	
E:	**OO-Conception**	
D:	**Abstraktion**	Die erste Entwurfsphase, in der nur die für die Fachprozesse wesentlichen Aspekte der "realen" Welt betrachtet werden.
F:	**Abstraction**	
E:	**Abstraction**	
D:	**Abstraktions-Niveaus**	Ein Begriff aus der Methode MERISE, bezeichnet die verschiedenen Datenmodelle, die nötig sind, um einen Ausschnitt der realen Welt in eine physikalische Datenstruktur umzuwandeln. Dieser Begriff soll mit Vorsicht verwendet werden, da eine eigentliche Abstraktion nur in einer frühen Phase durchgeführt wird.
F:	**Niveaux d'abstraction**	
E:	**Abstraction levels**	
D:	**Semantischer Entwurf**	Die Entwurfsebene der Abstraktion, um die für den Fachprozess relevanten Informationen bzw. Daten zu strukturieren. Aus dem semantischen Entwurf resultiert ein semantisches Daten- , Funktions- und Organisationsmodell; dieses ist Software-unabhängig.
F:	**Modélisation sémantique**	
E:	**Semantical Modeling**	

D:	**Konzeptueller Entwurf**	Die Entwurfsebene der Formalisierung der semantischen Modelle, um Redundanzen zu reduzieren (Normalisierung) und Rahmenbedingungen zu formalisieren. Aus dem konzeptuellen Entwurf resultieren konzeptuelle Modelle; diese sind Software-unabhängig.
F:	**Modélisation conceptuelle**	
E:	**Conceptual Modeling**	
D:	**Logische Umsetzung**	Die Entwurfsebene der Umsetzung der resultierenden Modelle in eine EDV-Sprache der Entwicklungsumgebung SEU. Aus dieser Umsetzung entstehen Programme in SQL, JAVA u.ä. und interpretierbare Schemata, beispielsweise in einem CASE-Tool.
F:	**Mise en œuvre logique**	
E:	**Logical Implementation**	
D:	**Physische Umsetzung**	Die Entwurfsebene der Interpretation der logischen Schemata und Programme durch Compiler und Speicherzuteilungen.
F:	**Mise en œuvre physique**	
E:	**Physical Implementation**	
D:	**Phasenmodell**	Eine Methode, die die verschiedenen Lebenszyklen einer logischen Datenbank strukturiert. Sie bringt die verschiedenen Entwurfsmodelle mit der zeitlichen Abfolge des Entwurfs und der Umsetzung einer Datenbank in Zusammenhang.
F:	**Modèle de phases**	
E:	**Phase model**	
D:	**Wasserfallmodell**	Ein Phasenmodell, das jede Phase nur einmal durchlaufen lässt.
F:	**Modèle de cascades**	
E:	**Model "waterfall"**	

5. Entwurfsmethoden

RUP	Rational Unified Process: eine auf UML basierende Entwurfsmethode, die von Rational Rose vertrieben wird.
MERISE	Eine auf dem relationalen Entwurfsmodell basierende Entwurfsmethode.
MEGA	Eine von MERISE abgeleitete Methode, die unter anderem Informatik-Werkzeuge für den Entwurf zur Verfügung stellt.
HERMES	Eine Methode des Bundesamtes für Informatik und Telekommunikation für Informatik-Projekte.

6. Datenmodellierung

6.1 Allgemeines

D:	**Datenmodell**	Ein Konzept der Datenstrukturierung, bestehend aus Informationsobjekttypen und Assoziationen. Datenmodelle werden mit Hilfe von Datenschemata kommuniziert.
F:	**Modèle de données**	
E:	**Data model**	

D:	**Datenschema**	Ein mit Hilfe einer verbalen oder graphischen Notation dargestelltes Datenmodell.
F:	**Schéma de données**	
E:	**Application schema**	

D:	**Konzeptuelles Datenschema**	Ein mit einer geeigneten Notation erstelltes Datenschema, das das Datenmodell für Entwerfer und für ausgebildete Fachleute widerspruchsfrei darstellt; "Brücke" zwischen Fachsicht und Informatik, resp. EDV.
F:	**Schéma de données conceptuel**	
E:	**Conceptual data schema**	

D:	**Informationsobjekt**	Semantische Abstraktion eines Objektes der realen Welt.
F:	**Objet d'information**	
E:	**Information object**	

D:	**Informationsobjekttyp**	Zusammenfassung ähnlicher Informationsobjekte durch Typisierung.
F:	**Type d'objet d'information**	
E:	**Information object type**	

6.2 Modellierung

D:	**Entwurfsmodell**	Eine Anleitungen zum Entwurf; Es können zwei Hauptentwurfsmodelle unterschieden werden : - Relationales Entwurfsmodell - "Objekt-orientiertes" Entwurfsmodell
F:	**Modèle de conception**	
E:	**Formalism**	

D:	**Relationale Modellierung**	Entwurfsmodell, das auf der formalen Algebra beruht.
F:	**Modélisation relationnelle**	
E:	**Relational Modeling**	

D:	**Formale Algebra**	Relationale Algebra, mathematischer Bereich, der verschiedene Operatoren zur Tabellenvereinigung beschreibt.
F:	**Algèbre formelle**	
E:	**Formal Algebra**	

D:	**OO-Modellierung**	Entwurfsmodell, das OO-Ansätze verwendet.
F:	**Modélisation OO**	
E:	**OO-Modeling**	

D:	**Generalisierung**	Die Gruppierung ähnlicher Informations-Objekte in einen Objekttyp, in welchem alle für die Fachprozesse wesentlichen Eigenschaften aller Objekten vorhanden sind.
F:	**Généralisation**	
E:	**Generalisation**	

D:	**Spezialisierung**	Bezeichnet die Ausgliederung ähnlicher Objekte aus einem generalisierten Objekttyp, um sie detaillierter beschreiben zu können.
F:	**Spécialisation**	
E:	**Specialisation**	

D:	**Assoziation**	Eine Menge von Verbindungen zwischen Objekten von zwei oder mehreren Klassen. Eine Assoziation beschreibt eine Menge von Verbindungen mit gemeinsamer Struktur und Semantik.
F:	**Association**	
E:	**Association**	
D:	**Aggregation**	Eine Form der Assoziation, eine Ganzes-Teil Beziehung, worin die Objekte an einem Ende als "Ganzes" und die am anderen Ende als "Teile" bezeichnet werden.
F:	**Agrégation**	
E:	**aggregation**	
D:	**Komposition**	Eine Form der Aggregation mit stark ausgeprägter Eigentümerschaft des Ganzen über die Teile und übereinstimmender Lebensdauer von Ganzem und Teile (z.B. ein Rad, eine Kupplung usw. sind Bestandteile des Autos)
F:	**Composition**	
E:	**Composition**	
D:	**Kapselung**	Ein Konzept der Objekt-orientierten Modellierung; in der auf Daten eines Objekts einer Klasse nur über die entsprechenden Methoden des Objekts zugegriffen werden können. Daten sind a priori in den Objekten versteckt (Geheimnisprinzip).
F:	**Encapsulation**	
E:	**Encapsulation**	
	Information hiding	Ein Konzept der Objekt-orientierten Modellierung; in der Objekte Daten enthalten können, die von aussen (von ausserhalb des Objekts) nicht zugegriffen werden können. Diese Daten werden ausschliesslich für die Ausführung von internen Methoden benutzt. Einige Autoren setzen das Konzept des information hiding dem Kapselungsprinzip gleich.
D:	**Vererbung**	Ein Konzept der Objekt-orientierten Modellierung; die Übergabe aller öffentlichen Daten einer Klasse an einer anderen Klasse.
F:	**Héritage**	
E:	**Inheritage**	
	Overriding	Ein Konzept der Objekt-orientierten Modellierung; eine übergebene Methode einer Klasse an eine andere Klasse, kann von dieser durch eine eigene Methode spezifiert oder verändert werden.
D:	**Polymorphismus**	Polymorphismus bedeutet, dass eine Nachricht unterschiedliche Operationen auslösen kann,je nachdem zu welcher Klasse die Objekte gehören, an die sie geschickt wird..
F:	**Polymorphisme**	
E:	**Polymorphism**	

7. Aufbau einer Sprache

D:	**Metamodell**	Ein Datenmodell einer Beschreibungssprache, es enthält alle Sprachelemente.
F:	**Méta modèle**	
E:	**Meta model**	
D:	**Semantik**	Die Bedeutung von Sprachelementen (Wörter).
F:	**Sémanique**	
E:	**Semantic**	

D:	**Syntax**	Die Stellung eines Sprachelementes innerhalb eines Satzes.
F:	**Syntaxe**	
E:	**Syntax**	

D:	**Grammatik**	Regeln, wie die verschiedenen Sprachelemente (Wörter) in einem Satz aufeinander abgestimmt werden sollen. Eine eigentliche Grammatik existiert bei Beschreibungssprachen nicht. Es wird weder dekliniert, noch konjugiert. In der Literatur wird vielfach die Syntax als Bestandteil der Grammatik angesehen. Im vorliegenden Bericht wird sie jedoch unterschieden.
F:	**Grammaire**	
E:	**Grammar**	

D:	**Orthographie**	Rechtschreibung
F:	**Orthographie**	
E:	**Orthography**	

8. Datenbeschreibungssprachen und Notationenen

D:	**Beschreibungssprache**	Sprachelemente, die ein Daten-, Funktions- oder Organisationsmodell widerspruchsfrei beschreiben, um sie den entsprechend ausgebildeten Beteiligten zu vermitteln.
F:	**Language de description**	
E:	**Description Language**	
D:	**Verbale Notation**	Verbale Sprachelemente, die eine Daten- (Funktions- oder Organisations-) struktur für ausgebildete Beteiligte wiederspruchsfrei widergeben.
F:	**Notation textuelle**	
E:	**Lexical Notation**	
D:	**Graphische Notation**	Graphische Elemente die eine Daten-(Funktions- oder Organisations-) struktur für ausgebildete Beteiligte widerspruchsfrei wiedergeben.
F:	**Notation graphique**	
E:	**Graphical Notation**	
D:	**ERD**	(Entity Relationship Diagram) Eine graphische Datenbeschreibungssprache auf semantischer und konzeptueller Ebene.
F:	**diagramme E-A**	
E:	**ERD**	
	EXPRESS	Datenbeschreibungssprache auf konzeptueller Ebene, Standard der CEN TC 287. EXPRESS-G ist ein Subset für geografische Modelle.
	INTERLIS	Datenbeschreibungssprache auf konzeptueller Ebene, Standard in der Schweiz.
	UML	Unified Modeling Language: Graphische Sprachelemente, die die Dokumentierung der "Objekt-orientierten" Modellierung ermöglichen.
	SQL	Selected Query Language: eine Sprache zur Daten-Definition, -Manipulation und -Abfrage; basiert auf relationalem Modell.

9. Umsetzung eines IS

D:	**CASE-Werkzeuge**	Informatik-Werkzeuge, die den Entwurf unterstützen.
F:	**Outil AGL**	
E:	**CASE Tools**	
D:	**SEU (Software-Entwicklungs-umgebung)**	Informatik-Werkzeuge, die die Softwareentwicklung ermöglichen und vereinfachen: Programmiersprachen, GUI-Builder, usw.
F:	**Environnement de développement**	
E:	**Software development environment**	

D:	**SBP (Software-Betriebsplattform)**	Stellt der SEU die grundlegenden Hardware- und Software-Funktionalitäten zur Verfügung.
F:	**Plate-forme d'exploitation**	
E:	**Operation platform**	
D:	**DBMS (Datenbank-Management System)**	Eine Software, die den Zugang zur physischen Datenbank gewährleistet. Durch sie können Datenbankmanager auf Daten zugreifen oder neue Datenmodelle definieren.
F:	**SGBD (système de gestion de la base de données)**	
E:	**DBMS (data base management system)**	
	GUI-Builder	Informatik-Werkzeuge, für die Erstellung der grafischen Benutzer-Schnittstelle (GUI, grafic user interface).

10. Management von raumbezogenen Daten

D:	**GIS**	Ein Informationssystem, das Daten kartographisch darstellt und das zusätzlich geographische, geometrische und topologische Bearbeitungs- und Auswertungs-Funktionen enthält.
F:	**SIRS**	
E:	**GIS**	
D:	**LIS**	Landinformationssystem, ein GIS, das geographische Daten massstabsunabhängig verwaltet. Überlappungsfreie, flächendeckende Modellierung sind vielfach Eigenschaften eines LIS. Beispiel eines LIS: amtliche Vermessung. Die Unterscheidung zwischen LIS und anderen GIS ist heute nicht mehr relevant. Sie werden unter dem Begriff GIS zusammengefasst.
F:	**SIT**	
E:	**LIS**	
D:	**Geo-Viewer**	Ein Informatikmittel zur kartographischen Visualisierung von anderswo abgespeicherten Daten. Enthält nur wenige Auswertungsmöglichkeiten (z.B ArcExplorer).
F:	**Outil de représentation cartographique**	
E:	**Geo-Viewer**	
D:	**Topologie**	Beschreibt die relative räumliche Beziehung zu anderen Objekten, unabhängig von ihrer exakten Position und Form. In Netz-artigen Strukturen wird die Topologie häufig mit Knoten und Kanten beschrieben.
F:	**Topologie**	
E:	**Topology**	
D:	**Geometrie**	Beschreibt die Form eines Objektes mit Hilfe geometrischer Grössen (Länge, Winkel) oder mit Hilfe von Koordinaten.
F:	**Géométrie**	
E:	**Geometry**	

D: ***F:*** ***E:***	**Raumbezugssystem** **Système de repérage** **Referencing System**	Definiert ein geometrisches Gebilde (z.B. Fläche oder Netz) und bestimmt genau identifizierbare Objekte (z. B Fixpunkte), die dazu benötigt werden, die Position anderer Objekte zu beschreiben. Beispielsweise um ein globales Bezugssystem einzuführen, wurde ein Ellipsoid (WGS84) definiert, das die Erdoberfläche annähert, und als Fixpunkte wurden einige genau positionierte Stationen festgelegt. GPS benützt Satelliten als Fixpunkte; Satelliten sind zwar nicht fix, aber deren Position in Funktion der Zeit ist sehr genau bekannt.
D: ***F:*** ***E:***	**(Räumliches) Basis-Bezugssystem** **Système de repérage de base** **Standard Referencing System**	Das primäre Raumbezugssystem, das im Strassenunterhalt benutzt wird. Es wird definiert durch eine Menge von Stassenachsen, die durch Bezugspunkte in Sektoren unterteilt werden.
D: ***F:*** ***E:***	**RBBS** **SRB** **BSRS**	Siehe (Räumliches) Basis-Bezugssystem
D: ***F:*** ***E:***	**(Schweizer) Landes-koordinaten** **Coordonnées nationales (suisse)** **Swiss National coordinates**	Ein Koordinatensystem, das von der amtlichen Vermessung und für die Kartographie (z.B. Militär, LK) benutzt wird. Diese Koordinaten beruhen auf dem Bessel-Ellipsoid.
	STRADA-View	Eine Zusatzapplikation zu STRADA-DB, die Resultate graphisch darstellen und weitere Verarbeitungen durchführen kann. STRADA-View besteht aus zwei Modulen : STRADA-View/Carto und STRADA-View/Axband.
	STRADA-View/Carto	Ein Werkzeug zur kartographischen Darstellung von STRADA-DB Daten.
	STRADA-View/Axband	Ein Werkzeug zur graphischen Darstellung von STRADA-DB Daten. Die Strassenaxse wird als eine Gerade dargestellt und die Daten werden darauf projiziert.
	STRADA-DB/JOKER	Eine Zusatzapplikation zu STRADA-DB, die es den Benutzern ermöglicht, eigene Raum-/Zeit-bezogene Informationsobjekttypen in die Datenbank einzubinden.

11. Strassenmanagement-Systeme:

SMIS	Informationssystem des "Managements der Strassenverkehrsanlage".

D:	**MSE (Management der Strassenerhaltung)**	Die Führung, die Koordination und die Überwachung der gesamten Tätigkeiten im Hinblick auf die Strassenerhaltung in der Schweiz.
F:	**SGE (Système de gestion de l'entretien routier)**	
E:	**MSE**	
D:	**MSE-Teilsystem**	Eines der im MSE definierten Teilsysteme (z. B. Fahrbahn, Kunstbauten).
F:	**Système partiel SGE**	
E:	**Subsystem MSE**	
	STRADA-DB	ist die Datenbank für die allgemeinen Strassendaten (Kerndaten). Die allgemeinen Daten setzen sich zusammen aus den Basis- und den Generalistendaten. Die Spezialistendaten sind in der Regel nicht in STRADA-DB. Sie können aber als Spezialmodule in STRADA-DB integriert werden oder an STRADA-DB gekoppelt werden.
	STRADA	ist der Sammelbegriff für alle Informatik-Module für Strassendaten im RHISS: die Strassendatenbank STRADA-DB, das Präsentationswerkzeug STRADA-View, die Selektions- und Auswertungsmodule STRADA-INFO und die Berechnungs- und Statistik-Module STRADA-CALC.
	STRADA-IS	ist ein auch in den Kantonen betreibbarer Teil des RHISS: das Informationssystem "Strassendaten" als Datenfundament, unter anderem für das MSE und die MSE-Applikationen (STRADA-MS). Es umfasst alle STRADA-Teile und weitere Informationssysteme, z.B. Geo-IS, die mit STRADA mehr oder weniger eng gekoppelt sind.
	RHISS	ist das "Informationssystem Strasse" des ASB als Gesamtsystem im Sinne des Data-Warehouse-Gedankens. Wesentliche Fundamente sind STRADA/KUBA, ISA, Büroautomation usw.
	PMS	Pavement Management System, ein Managementsystem, für die Mehrjahresplanung des Fahrbahnunterhalts.
	BMS	Bridge Management System, ein Managementsystem für die Mehrjahresplanung des Unterhalts der Kunstbauten.
	EMS	Electromechanical Management System, ein Managementsystem, für die Mehrjahresplanung des Unterhalts der elektronischen Anlagen.

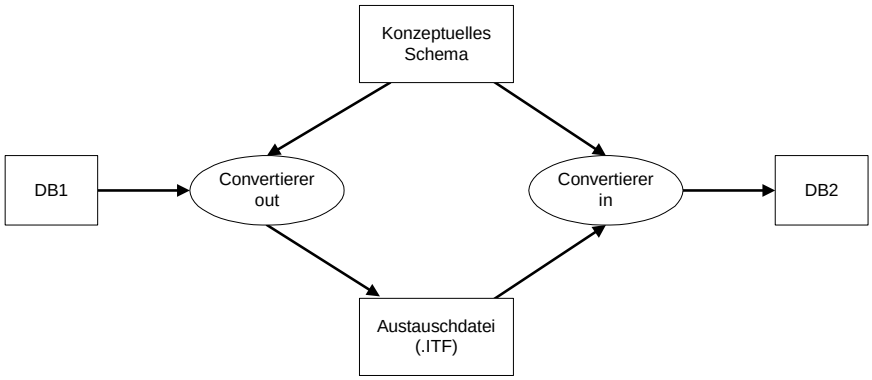
D INTERLIS – ein Austauschmechanismus

1 Prinzip

In den Referenzdokumenten von INTERLIS werden folgende Aspekte beschrieben :

- Ein Austauschmechanismus
- Eine Datenbeschreibungssprache
- Ein Datenformat

INTERLIS ermöglicht standardisierte Schnittstellen, die den Austausch zwischen mehreren Austauschpartnern sicherstellen. Diese bestehen je aus einer Beschreibung und einer Austauschdatei. Die Beschreibung ist nötig, um einen strukturellen Konsens aller Beteiligter auf konzeptueller Ebene zu finden. Diese Beschreibung nennt man konzeptuelles Schema und sie beinhaltet die Spezifikation aller Daten, auf die sich die Beteiligten einigen konnten. Diese soll zusätzlich automatisch verarbeitbar sein. Dazu muss das konzeptuelle Schema in einer formellen Sprache beschrieben sein, die von einem Konvertierer lesbar ist. Die von INTERLIS vorgeschlagene Sprache wird INTERLIS Data Description Language, kurz IDDL, genannt. INTERLIS stellt somit das Konzept einer offenen Schnittstelle dar, da ein Benutzer das konzeptuelle Schema mit eigenen Datenbeschreibungen definieren kann und damit seine eigenen Daten transferieren kann. Das konzeptuelle Schema muss dem Zielsystem bekannt sein, um die Daten in der Austauschdatei interpretieren zu können. Das Format der Austauschdatei wird in der 1. Version INTERLIS Transfer Format (ITF) genannt, währenddem sich die 2. Version auf das XML-Format stützt.



Austauschmechanismus von INTERLIS.

2 Die Beschreibungssprache von INTERLIS (IDDL)

IDDL ist eine Sprache, die Daten auf konzeptueller Ebene beschreibt. Sie beinhaltet folgende Elemente :

- Modelle
- Layer
- Tabellen
- Konzeptuelle Schlüssel
- Fremdschlüssel
- Konsistenzbedingungen (Wertebereich usw.)
- Geometrie

IDDL ist eine Sprache die alle Grundkonzepte der relationalen Datenmodellierung darstellen kann. Obwohl das Datenmodell im konzeptuellen Schema hierarchisch sein muss (Forderung von INTERLIS) kann mit geeigneter Modellierung praktisch jedes relationale Modell in ein hierarchisches umgewandelt werden.

IDDL ermöglicht die Beschreibung der geometrischen Form eines Entitätstypen als einfaches mehrwertiges Attribut. INTERLIS wird daher auch als 'Objekt'-relationales Modell betrachtet. Geometrietypen wie Punkte, Polylinien, überlappende Flächen und überlappungsfreie Flächen sind darstellbar.

Konsistenzbedingungen sind in der Beschreibung unerlässlich, wenn das Zielsystem die Qualität der erhaltenen Daten überprüfen soll. INTERLIS stellt einige Standard-Konsistenzbedingungen zur Verfügung, ermöglicht aber dem Benutzer zusätzliche eigene Konsistenzbedingungen einzufügen. Eine Formalisierung der Konsistenzbedingungen wird durch INTERLIS nicht vorgeschlagen.

3 IDDL – im internationalen Vergleich (Siehe Bericht, Kap. 7.1):

	Anforderungen	**INTERLIS (Version 1)**
1	Formale Sprache ...	(+)
2	Elemente des konzeptuellen Schemas	(+)
2A	Struktur	(+)
2B	Verhalten	(-)
2C	Konsistenzbedingungen	(+/-)
3	Graphische Notation	(-)
4	Automatische Verarbeitbarkeit	(+)
5	Modularität	(-)
6	Geometrie	(+)
7	Transfer-Schema	(+)
8	Dokumentation	(+)
9	IT Standard	(+/-)
10	Unterstützende Software	(+)
11	Erfahrung und Support	(+)

IDDL besitzt gemäss CEN/TC-287 Mängel bezüglich seinem Verhalten, den Konsistenzbedingungen, der Modularität, der graphischen Notation und der Standardisierung. Die Version 2 von INTERLIS kann die Mängel im Verhalten, bei den Konsistenzbedingungen und bei der Modularität teilweise aufheben.

4 Zusätze von INTERLIS Version 2

Die Erfahrungen der letzten Jahre zeigen, dass durch eine gute Dokumentation der Daten ein Missverständnis zwischen Datensender und –empfänger vermieden werden kann. Die Sprache von INTERLIS wurde in erster Linie in der amtlichen Vermessung angewandt. Heute kann sie als schweizerischer Transferstandard betrachtet werden.

Die Version 2 berücksichtigt neue Technologien in der Datenmodellierung und neue Bedürfnisse, die die erste Version nicht abdeckt.

Die Version 2 beinhaltet zusätzlich Folgendes :

1. Erweiterungen zur OO-Modellierung
2. Einführung von Sichten
3. Modell zur graphischen Darstellung
4. Inkrementelle Nachführung

4.1 Erweiterungen zur OO-Modellierung

INTERLIS Version 2 beinhaltet folgende OO-Konzepte:

- Vererbung (in INTERLIS Handbuch Erweiterung genannt)
- Beziehungstypen wie Assoziation, Agregation und Komposition

Das Bedürfnis nach OO-Konzepten in der INTERLIS Beschreibungssprache wurde vor allem in der Vermessung festgestellt. In der Schweiz liefert der Bund in der Vermessung eine Datenstruktur, die die Bedürfnisse des Bundes zum Ausdruck bringt und die den Kern für kantonale Reglemente und Datenbanken bilden soll. Diese Struktur des Bundes kann von den kantonalen Verwaltungen darart erweitert, dass sie auch deren Bedürfnisse erfüllen können. Durch das OO-Konzept der Vererbung bzw. Erweiterung, kann die kantonale Verwaltung seine zusätzliche Datenstruktur beschreiben ohne dass sie diejenige des Bundes nochmals darstellen muss. Ähnlichkeiten und Unterschiede zwischen Bund und kantonalen Datenstrukturen können damit klarer aufgezeigt werden.

Im folgenden Beispiel wird die Vererbung anhand des Bezugspunktes aufgezeigt:

```
Transfer Roads

MODEL STRADA =
…
TOPIC STRADA/UR-A

  STRUCTURE Verwaltung =
      ODO_Owner:     MANDATORY      CKO;
      OrigSubDBID:   MANDATORY      TEXT*4;
      DAO_Owner:     MANDATORY      CKO;
      DataOwner:     MANDATORY      CKO;
      CreateDate:    MANDATORY      DATUM;
      ChangeDate:                   DATUM;
      CHO_Owner:     MANDATORY      CKO;
      ChangeUser:    MANDATORY      TEXT*32;
      ITS_Code:      MANDATORY      TEXT*2;
      IntegrityDate: MANDATORY      DATUM;
      XST_XferSetID:                TEXT*20;
  END Verwaltung;

  CLASS Reference_Point EXTENDS Verwaltung=
      BaseID:        MANDATORY      MSK;
      Version:       MANDATORY      VERS;
      Axe_BaseID     MANDATORY      ->Axis
      CK:            MANDATORY      CK;
      Ret_BaseID:    MANDATORY      ->ComplexText
      TypAdText:                    TEXT*72;
      Name:                         TEXT*32;
      SortKey:       MANDATORY      NUM8;
      SectorL:       MANDATORY      NUM4_3;
      SLPrecision:                  NUM2_3;
      LatDist:       MANDATORY      NUM2_3;
      Km_Number:                    NUM4_3;
      CoorY:                        NUM6_3;
      CoorX:                        NUM6_3;
      PreciXY:                      NUM2_3;
      CoorZ:                        NUM4_3;
      PreciZ:                       NUM2_3;
      PlateAxisDist:                NUM2_2;
      PlateRefDist:                 NUM2_2;
      VRS_Code:      MANDATORY      TEXT*2;
      RefDate:       MANDATORY      DATUM;
      BeginValidity: MANDATORY      DATUM;
      EndValidity:                  DATUM;
      Comment:                      COMMENT;
  CONSTRAINT
    UNIQUE
      BaseID, Version;  !!Logical key
      Axis, CK, RefDate !! Conceptual Key;
    END Reference_Point;
```

In diesem Beispiel wird eine generische Struktur 'Verwaltung' beschrieben, die die Verwaltungsattribute enthält. Diese Attribute sind für jeden anderen Objekttyp zu beschreiben. In der Version 1 müsste man also die Gesamtheit der Verwaltungsattribute für jeden Entitätstyp wiederholt beschreiben. Dank der Objektorientierung der Version 2 können diese Attribute anderen Entitätstypen vererbt werden. Da die Verwaltungsattribute immer im Zusammenhang mit anderen Entitätstypen vorkommen, wird die Verwaltungstabelle als Struktur (generische Tabellenstruktur) definiert.

Es ist zudem auch möglich, Wertebereiche zu vererben bzw. zu erweitern, beispielsweise eine Aufzählung:

- Art: (Natur, Kultur, Geschichte)
- Art EXTENDED : (Geschichte(Kirche, Museum, Theater)) ;

In INTERLIS 2 können zudem noch Assoziationen, Agregationen und Kompositionen beschrieben werden. Solche Beziehungen können mit folgenden Eigenschaften versehen werden:

- Kardinalität: minimale und maximale Anzahl der in Beziehung stehenden Objekte
- Stärke: der zu erwartende Prozentsatz der Objekte der beiden beteiligten Klassen, die wirklich in Beziehung gebracht werden.
- Rolle: Beschreibung der Rolle der in einer Beziehung gebrachten Objekte.

4.2 Einführung von Sichten

Neben den Klassen, die die eigentlichen Sachdaten enthalten, können auch Sichten beschreiben werden, d.h virtuelle Klassen, deren Objekte aus anderen Objekten anderer Klassen abgeleitet werden. Sichten gehören eigentlich nicht zu einer konzeptuellen Datenbeschreibung. Die Beschreibung von Sichten kann aber sehr nützlich sein, um Funktionen vereinfacht darzustellen.

Eine Sicht verbindet somit mehrere Klassen oder Sichten (sog. Basismengen) mit Hilfe einer Verbindungsvorschrift. Folgende Vorschriften werden unterschieden:

- Verbindung (JOIN): kartesische Produkt der Basismengen, z.B. die Verbindung der Menge A mit den Elementen (a ,b ,c) und der Menge B mit den Elementen (x , y, z) ergibt eine Sicht mit den Elementen (ax, ay, az, bx, by, bz, cx, cy, cz)
- Vereinigung (UNION): Vereinigungsmenge der Basismengen, z.B. die Vereinigung der Menge A mit den Elementen a ,b ,c und der Menge B mit den Elementen x , y, z ergibt eine Sicht mit den Elementen (a, b, c, x, y, z)
- Dekompositionen (DECOMPOSITION): Darstellung der Elemente einer Komposition als Objekte, z.B. eine Dekomposition einer Ganglinie, die mehrere Wertepaare (Zeitpunkt, Verkehrswert) enthält, ergibt eine Sicht mit jedem Wertepaar als eigenes Objekt.

Mit Hilfe von Selektionen (welche Objekte sollen dargestellt werden?) und Projektionen (welche Attribute der selektierten Objekte sollen dargestellt werden?) können zusätzlich die Daten gefiltert werden.

```
Beispiel:
VIEW ReferencePoints_on_NationalRoads
  JOIN Reference_Point, Axis;
  WHERE Reference_Point.Axis == Axis and Axis.AxisType == "National Road";
ATTRIBUTE
  ALL OF Axis
  RPTNumber: TEXT*10 := Reference_Point:CK;
END ReferencePoints_on_NationalRoads;
```

4.3 Beschreibung der graphischen Darstellung

Die konzeptuelle Beschreibung der graphischen Darstellung hat zum Ziel:

- Die Formulierung eines Darstellungsschemas in IDDL, die eine Beschreibung einer applikationsneutralen Symbolik ermöglicht.
- Die Festlegung einer Darstellungssprache, die einem Datentypen eine Symbolik zuspricht.

Eine saubere Beschreibung der graphischen Darstellung benötigt folgende Modelle (gemäss Definition im INTERLIS-Handbuch):

- Daten- bzw. Sichtenmodell: beinhaltet Objekte, die dargestellt werden sollen.
- Signaturenmodell: beinhaltet verschiedene Symbole, Texte, Linien- und Flächentypen, die für die Darstellung benutzt werden können.
- Graphikmodell: teilt einem darzustellenden Objekt eine Signatur zu.

Das Signaturmodell muss nicht unbedingt in der Beschreibung enthalten sein. Es genügt, wenn auf das Signaturenmodell verwiesen wird.

4.4 Inkrementelle Nachlieferung

Unter inkrementeller Nachlieferung wird die Aktualisierung einer externen Datenbank von einer Hauptdatenbank verstanden, indem man nur die modifizierten Objekte liefert. Die Nachführung einer Hauptdatenbank wird ausgeschlossen.

Die inkrementelle Nachlieferung verhindert den Austausch von ganzen Datensätzen, nur um einige wenige Objekte nachzuführen. Nur das Inkrement wird transferiert, d. h. der Unterschied zwischen zwei Datenbankzuständen.

Diese Nachlieferung kann nur erreicht werden, wenn ein Objekt eindeutig über Datenbanken hinweg und Zeit-unabhängig identifiziert werden kann. In INTERLIS wird dieser Identifikator GID (Global Identification) genannt.

Die inkrementelle Nachlieferung zieht keine Veränderungen in der Beschreibungssprache nach sich. Nur im Transferformat müssen entsprechene Metadaten eingeführt werden, um die Abstammung der Daten zu übermitteln.

Folgende Operationen werden unterstützt:

- Einfügen neuer Objekte
- Nachführung alter Objekte
- Löschen alter Objekte

E Das Metamodell von INTERLIS

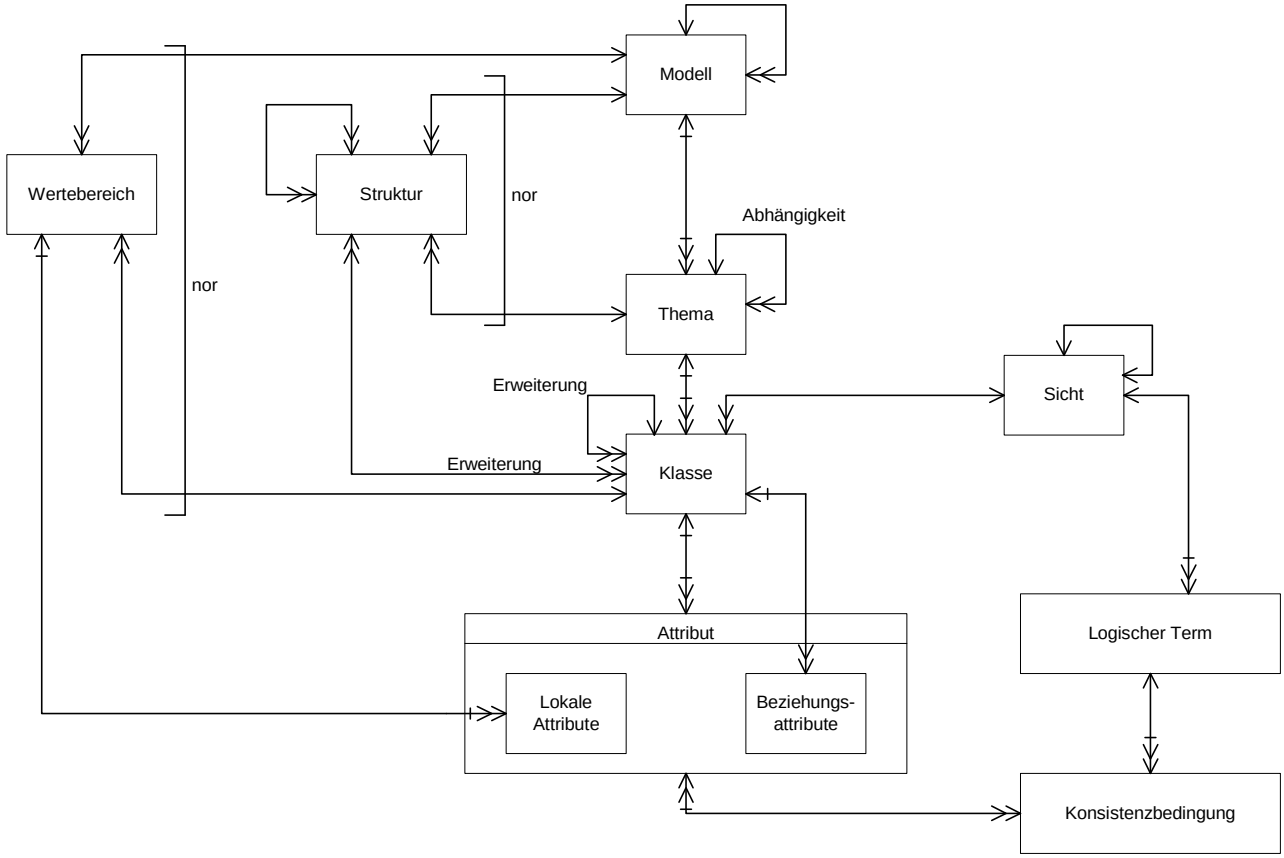
INTERLIS beschreibt grundsätzlich ein relationales Datenmodell (ausgenommen die geographischen Attribute). Die Version 2 von INTERLIS wurde unter anderem mit Begriffen aus der OO-Modellierung ergänzt.

1 Bibliographie

Eidgenössische Vermessungsdirektion (1998): INTERLIS, Ein Datenmodellierungs- und Austauschmechanismus für Geo-Informationssysteme. Version 2 Revision 1.

Eidgenössische Vermessungsdirektion (2000): INTERLIS Version 2.0, Referenzhandbuch. Ausgabe vom 2000-04-18 (deutsch).

2 Übersicht über das Metamodell



Die oben aufgeführte Darstellung stellt nur die wesentlichen Grundkonzepte von INTERLIS Version 2 dar. Sie kann folglich nicht als komplettes Metamodell verstanden werden.

2.1 Beschreibung der Informationsobjekttypen

Das Kernmodell wird aus den folgenden vier Informationsobjekttypen gebildet :

- Modell : Das Datenmodell.
- Thema : Ist eine Sammlung von Tabellen ("Topic")
- Klasse : Ist eine Sammlung von Attributen, die einen Informationsobjekttyp beschreiben
- Attribut : Ist eine Eigenschaft, die die Informationsobjekte eines Typen teilen.

Durch die Ergänzung von INTERLIS mit Begriffen aus der Objekt-orientierten Datenmodellierung wird es nun erlaubt, dass ein Modell (bzw. ein Thema) auf ein anderes Modell (bzw. Tabelle) referenziert wird. Dies heisst, dass die Datenstruktur eines Modells an ein anderes Modell vererbt wird. Zudem wurde ein sogenanntes Template eingeführt, eine abstrakte Tabelle, die selbst keine Werte enthalten darf, aber die an verschiedene Tabellen vererbt werden kann.

Die Attribute können in zwei Kategorien eingeteilt werden :

- Beziehungsattribute : Sind Attribute, die die Beziehung zu anderen Tabellen erstellen.
- Lokale Attribute : Alle anderen Attribute. Sie werden durch ein Wertebereich (DOMAIN) ausgezeichnet, der alle möglichen Werte des Attributs beschreibt.

Jedes Attribut kann mit zusätzlichen Konsistenzbedingungen ausgezeichnet werden, seien es nun Beziehungsattribute oder lokale Attribute.

Sichten können über mehrere Klassen oder andere Sichten gebildet werden. Objekte, die in der Sicht integriert werden müssen, können über logische Terme selektiert werden.

Genauere Angaben zu INTERLIS 2 finden Sie im Anhang D.

F Amtliche Vermessung

1 Text im Hauptbericht

Die AV 93 definiert in der technischen Verordnung TVAV den Datenkatalog der amtlichen Vermessung: Er enthält folgende Informationsebenen : Fixpunkte, Bodenbedeckung, diverse Objekte/ lineare Elemente, Nomenklatur, Altimetrie, Liegenschaften, Leitungen und administrative bzw. technische Gebietseinteilungen.

Der Strassenraum wird als eine Fläche in der Bodenbedeckungsebene geführt. Die Semantik entspricht nicht den im Strassenmanagement gebräuchlichen Festlegungen.

Die Daten der amtlichen Vermessung können mit Hilfe der in INTERLIS definierten Schnittstelle ausgetauscht werden.

2 Anhang

Le catalog de données se compose de couches d'information suivants :

1. Points fixes
2. Couverture du sol
3. Objets divers / Elements linéaires
4. Altimétrie
5. Nomenclature
6. Biens-fonds
7. Conduites
8. Divisions administratives et techniques

Les cantons peuvent définir d'autres couches d'information et d'autres information dans ces couches selon leurs exigences supplémentaires.

2.1.1 Points fixes

« Les points fixes sont des points de rattachement pour la mensuration officielle qui sont déterminés par des mesures et des méthodes de compensation dans le système de référence de la mensuration nationale et qui sont matérialisés durablement sur le terrain par la pose de repères fixes » (Ordonnance technique de la mensuration officiel art. 46).

Deux types de points fixes sont distingués : Les points fixes planimétriques (PFP) et les points fixes altimétriques (PFA). Pour les PFP, les coordonnées planimétriques (Y-X) ainsi que l'altitude sont déterminées, pendant que pour les PFA l'altitude et avec une moindre précision les coordonnées planimétriques sont définis.

Parmi ce deux classes de points fixes, trois catégories sont introduit :

- PFP1 (PFA1) : les points fixes provenant de la mensuration nationale
- PFP2 (PFA2) : les points fixes provenant de la mensuration officielle. Ses coordonnées sont dérivées directement des PFP1 (ou PFP2 ?).
- PFP3 (PFA3) : les points fixes provenant de la mensuration officielle. Ses coordonnées sont dérivées des PFP1, PFP2 ou PFP3.

Le précision exigée des points de repère (PFP1 et PFA1 exclut car ils sont définis par la mensuration nationale) dépend de la région dans laquelle un point de repère se trouve. Dans la mensuration officielle cinq niveaux de tolérance (NT) ont été introduits et chaque canton attribut des zones à ces niveaux.

- NT 1: régions urbaines
- NT 2: régions construites et zones à bâtir
- NT 3: régions agricoles et forestières d'exploitation intensive
- NT 4: régions agricoles et forestières d'exploitation extensive
- NT 5: régions alpestres et improductives.

La précision des points de repère sont définie par dans l'ordonnance technique de la mensuration officielle (OTEMO art. 28).

Prenons l'exemple de points de repère dans une région du niveau de tolérance 3. De plus, nous admettons que 2 PFP2 sont écartés d'un kilomètre. La distance moyenne entre deux PFP3 est 250 m. La précision exigée pour les PFP2 est 2.7 cm et pour les PFP3 4.65 cm en planimétrie, en altimétrie 4.4 cm pour les PFP2 et 7 cm pour les PFP3.

2.1.2 Couverture du sol

La couche d'information « couverture du sol » comprend les éléments suivants :

- Bâtiments
- Surfaces à revêtement dur subdivisées en :route/chemin, trottoir, îlot, chemon de fer, place d'aviation, bassin et autre surface à revêtement dur
- Surface vertes subdivisées en : pré/champ/pâturage, culture intensive (elle-même subdivisée en vignes et autres), jardin, tourbière et autre surface verte
- Eaux subdivisées : en eau stagnante, cours d'eau et roselière
- Surfaces boisée subdivisées en : forêt dense et autre surface boisée.
- Surfaces sans végétation subdivisées en : rocher, glacier/névé, éboulis/sable, gravière/décharge et autre surface sans végétation.

Les type d'information qui nous intéresse plus particulièrement sont les routes/chemins qui sont traitées comme des éléments surfaciques. Une route/chemin est une surface remplissant une fonction de desserte pour la circulation des piétons ou des véhicules, telles que routes (y compris les bandes de stationnement), chemin agricoles, forestiers et de débardage, sentier (surface en terre battue) et leurs écoulements, à savoir caniveaux et bordures en pierre. Les chemin de jardins qui ne sont pas d'intérêt public ne sont pas levés.

La précision planimétrique d'un points d'une bordure d'une telle surface dépend de sa définition claire le terrain.

	Précision d'un point clairement défini sur le terrain	Précision d'un point non défini exactement sur le terrain
NT2	10 cm	20 cm
NT3	20 cm	50 cm
NT4	50 cm	100 cm
NT5	100 cm	200 cm

2.1.3 Objets divers / Element linéaire

Cette couche d'information peut contenir les éléments suivants :

Mur, bâtiment souterrain, autre corps de bâtiment, eau canalisée souterraine, escalier important, tunnel/passage inférieur/galerie, pont/passerelle, fontaine, réservoir, pilier, couvert indépendant etc.

Pour ces objets d'information les mêmes exigences de précision que pour la couche « Couverture du sol » sont valables.

2.1.4 Nomenclature

La nomenclature doit comprendre les noms locales, noms de lieu et les lieux-dits.

2.1.5 Altimétrie

La couche altimétrie contient :

- Point côté,
- Arrête de terrain subdivisée en :ligne de rupture (cassure) et ligne de structure (douce).

Pour un côte bien définie dans une zone de NT2, son imprécision ne doit pas excédér 20 cm. Pour les points mal définis dans NT2 et les points de NT3 et 4 la tolérance est $1+3.5\ tg(\alpha)$ mètres. Par exemple, pour un point de NT3 dans une pente de 50 grads, la tolérance est 4.5 m. Les points de NT 5 ne doivent pas excéder l'écart de MNT25 (modèle numérique de terrain 25 du service de la topographie).

2.1.6 Biens-fonds

Cette couche contient :

- Les biens-fonds
- Les droits distincts et permanents subdivisée en droit de superficie et droit de source
- Points limite

Les précisions exigées sont les suivantes :

	Précision d'un point clairement défini sur le terrain	Précision d'un point non défini exactement sur le terrain
NT2	3.7 cm	20 cm
NT3	7 cm	35 cm
NT4	15 cm	75 cm
NT5	35 cm	150 cm

2.1.7 Les conduites

Cette couche comprend :

- Les conduite de pétrole, de gaz et autres conduites régies par la législation sur les installations de transport par conduites de combustibles ou carburants liquides ou gazeux
- Signal indiquant la situation des conduites

La précision exigée est la même que pour les biens fonds.

2.1.8 Division administratives et techniques

Cette couche comprend :

- Commune
- Point de limite territoriale (district, canton, pays)
- Répartition des plans
- Répartition des niveaux de tolérance
- Zones de glissement selon l'article 660a CC
- État de la réalisation
- Données sur le titre du plan du registre foncier

Seuls les exigences de la limite communale est mentionnées dans l'ordonnance technique de la mensuration officiel. Elles correspondent à celles des biens-fonds.

2.1.9 Interface de la Mensuration Officiel (IMO)

Les échanges se réalise par l'interface de la mensuration officiel (IMO). L'IMO comprend un catalogue de données de la mensuration officiel exprimé dans le langage de données « INTERLIS » et le format de transfert correspondants selon la documentation INTERLIS. Un récepteur de données de la mensuration officiel a le droit d'exiger que les données lui soit fourni en INTERLIS. C'est donc l'IMO qui définit la structure des données définies dans l'ordonnance technique OTEMO.

G Das UML-Metamodell

1 Einleitung

UML (Unified Modeling Language) wurde aus drei früheren Methoden von OO-Technologien heraus vereinheitlicht: Booch, Jacobsens Object Oriented Software Engineering (OOSE) und Ramboughs Object Modeling Technique (OMT). In den letzten Jahren hat sich UML als das Werkzeug für die OO-Modellierung herausgestellt.

Die Erläuterung des kompletten UML-Metamodells sprengt den Rahmen dieses Projekts. Daher wird in diesem Anhang vor allem auf die grundsätzlichen Aspekte der UML-Modellierungssprache eingegangen.

2 Bibliographie

Booch Grady; Rumbaugh James; Jacobson Ivar (1999) : The Unified Modeling Language; User Guide. Object Technology Series; Addison Wesley; Massachusetts, USA.

3 Übersicht über UML

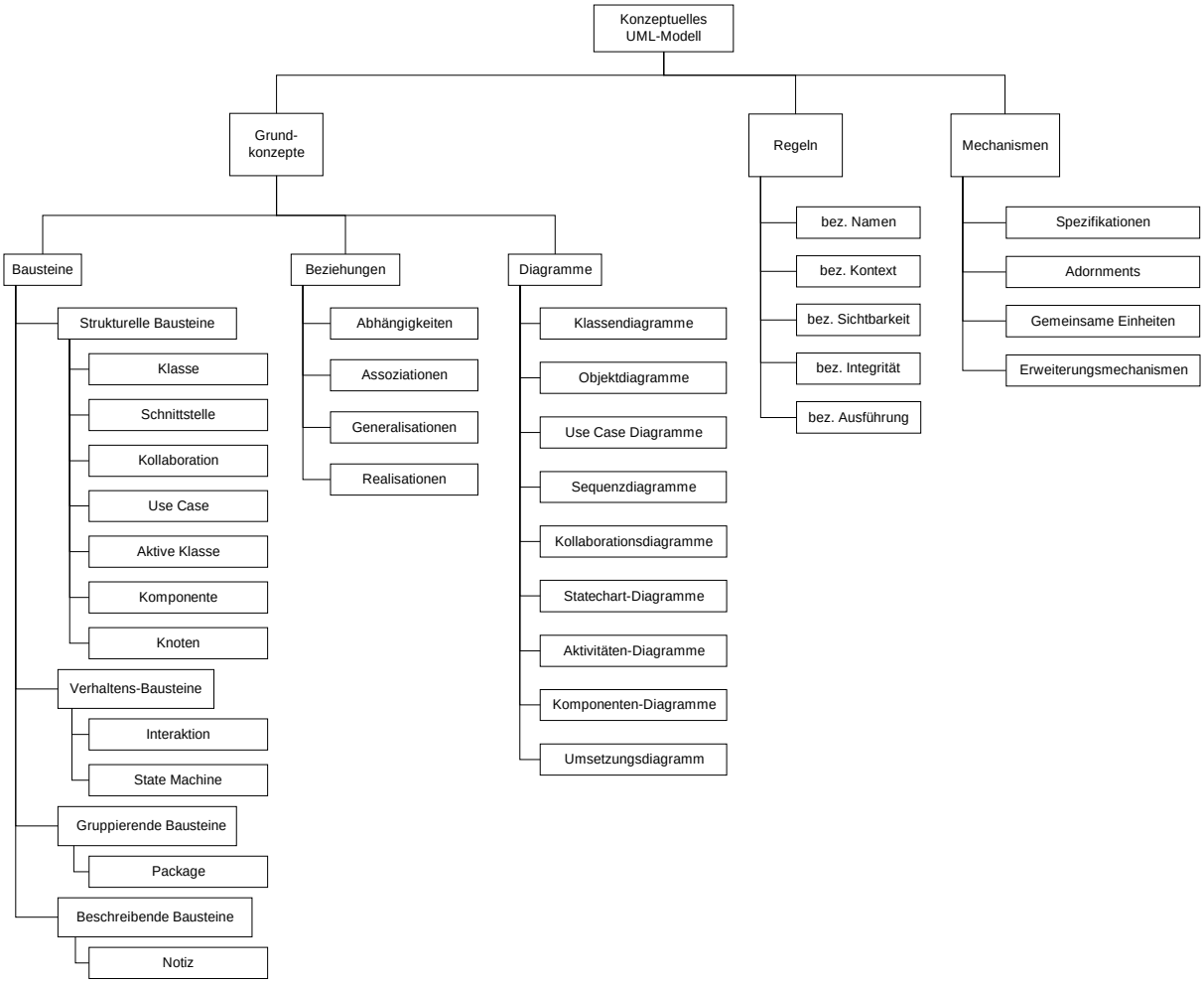


Figure 8-4 : Übersicht über die Grundkonzepte von UML

4 Beschreibung des Modells

4.1 Grundkonzepte

4.1.1 Bausteine

a) Strukturelle Bausteine

Strukturelle Bausteine sind die Tragelemente eines jeden UML-Modells. Sie sind mehrheitlich statischer Natur. Sie können entweder eine konzeptuelle oder eine physische Einheit darstellen.

Diese Bausteine können in folgende Kategorien eingeteilt werden :

- ***Klasse*** : eine Beschreibung einer Menge von Objekten, die gemeinsame Eigenschaften und eine gemeinsame Semantik besitzen. Um Objekte in Klassen zusammenzufassen, müssen die Objekte in ähnlicher Weise mit anderen Objekten in Beziehung stehen. Eine Klasse wird mit einem eindeutigen Namen identifiziert. Attribute und Methoden charakterisieren eine Klasse.
- ***Schnittstelle*** : eine Menge von Operationen, die ein bestimmtes Verhalten einer oder mehrerer Objekte einer Klasse (oder einer Komponente) auslösen oder die von einem oder mehreren Objekten ausgelöst werden. Nur mit Hilfe dieser Schnittstellen kann auf die in Klassen zusammengefassten Objekte zugegriffen werden. Ohne Schnittstelle kann der Inhalt eines Objekts nicht konsultiert werden. Das Verhalten eines Objektes, das mit einer gewissen Schnittstelle ausgelöst wurde, bezeichnet die Rolle des Objekts innerhalb eines Prozesses.
- ***Kollaboration*** : die Gruppierung mehrerer Klassen, die zusammen ein kooperatives Verhalten auslösen können. Dieses Verhalten kann viel mehr sein als die einfache Summe der Verhalten der Gruppen-Objekte.
- ***Use case*** : zusammengehörige Tätigkeiten, deren Ergebis einen Mehrwert für einen Akteur darstellen.
- ***Aktive Klasse*** : eine Klassen, deren Objekte eine oder mehrere Methoden besitzen, die Kontrollaktivitäten auslösen, um konkurrenzierende Prozesse aufeinander abstimmen zu können.
- ***Komponente*** : eine physische und ersetzbare Systemeinheit, die eine oder mehrere Schnittstellen realisiert. Ein COM+ oder ein JavaBeans ist beispielsweise eine Komponente.
- ***Knoten*** : ist eine physische Einheit, die eine gewisse Rechner-Kapazität ausweist, also eine Hardware-Einheit, die eine logische Komponente ausführt.

b) Verhaltens-Bausteine

Sie beschreiben die Dynamik in einem UML-Modell.

Es werden zwei Typen dieser Bausteine unterschieden:

- ***Interaktionen*** : ein Verhalten ein oder mehrerer Objekte, aus einer oder mehreren Klassen, das stets als eine Menge von ausgetauschten Meldungen ausgedrückt wird und einem bestimmten Zweck dient.
- ***State Machine*** : eine Sequenz aller Zustände eines Objektes oder einer Interaktion während ihres Lebens. Ein Zustand kann entweder "ruhig" (das Objekt wartet auf eine Meldung), aktiviert (das Objekt ist bereit, etwas auszuführen, wartet nur bis einige Bedingungen erfüllt sind) oder aktiv sein (das Objekt führt eine Methode aus).

c) Gruppierende Bausteine

Diese Bausteine ermöglichen die Strukturierung eines UML-Modells in Untermodelle. Sie werden nur aus organisatorischen Gründen eingeführt. Man spricht meistens von ***Package***.

d) Beschreibende Bausteine

Diese Bausteine werden gebraucht, um UML-Modelle erklären zu können. Sie werden in Form von ***Notizen*** in das UML-Modell eingebunden. Konsistenzbedingungen beispielsweise werden auch mit Hilfe von Notizen in das UML-Modell eingebracht. Der Inhalt einer Notiz kann in formalisierter Form (OCL siehe Anhang M) oder nicht formalisierter Form vorliegen.

4.1.2 Beziehungen

Die Beziehung können in folgende Kategorien eingeteilt werden:

- ***Abhängigkeiten*** : semantische Beziehungen zwischen zwei Bausteinen, in der eine Änderung des unabhängigen Bausteins eine semantische Änderung des abhängigen Bausteins mit sich bringen kann.
- ***Assoziationen*** : strukturelle Beziehungen zwischen Objekten. Die Aggregation beispielsweise ist ein Sonderfall einer Assoziation, die eine Klasse mit ihren Teilklassen in Beziehung bringt.
- ***Generalisationen*** : ermöglichen die Vererbung einer Datenstruktur bzw, der Methoden einer generalisierten Klasse ("Mutter"-Klasse) auf ihre spezialisierten Klassen ("Kinder"-Klassen). Eine Klasse "Möbel" kann beispielsweise die Attribute "Baujahr" und "Hersteller" an eine Klasse "Tisch" vererben.
- ***Realisationen***: Beziehungen zwischen Bausteinen, in der ein Baustein bestimmt, was für eine Methode ein anderer Baustein zu implementieren hat. Eine Schnittstelle beschreibt beispielsweise Methoden, die andere Objekte aus anderen Klassen zu implementieren haben.

4.1.3 Diagramme

Diagramme haben zum Ziel, gewisse Aspekte eines UML-Modells zu kommunizieren. Sie sind graphische Darstellungen, die eine Sicht des UML-Modells wiedergeben. Ein UML-Modell besteht aus folgenden Diagrammen.

- ***Klassendiagramme*** : bestehen aus Klassen, Schnittstellen und Kollaborationen. Sie beschreiben den statischen Aspekt eines UML-Modells.
- ***Objektdiagramme*** : bestehen aus einer Menge von Objekten mit ihren Beziehungen. Diese Diagramme ermöglichen eine detaillierte Sicht auf konkrete Objekte (im Gegensatz zu den eher abstrakten Klassen) zu einem bestimmten Zeitpunkt.
- ***Use Case Diagramme*** : zeigen bestimmte existierende oder erwünschte Vorgehen der verschiedenen am Informationssystem Beteiligten. Sie stellen eher eine statische Sicht der Aktivitäten dar (d.h. Prozeduren)
- ***Sequenzdiagramme*** : zeigen dynamische Sichten eines UseCase, d.h. unter anderem den Austausch der Meldungen zwischen den Objekten, in denen die sequentielle Abfolge der auszuführenden Methoden und Meldungen im Vordergrund stehen. Sequenzdiagramme können ohne Informationsverlust in Kollaborationsdiagramme übergeführt werden.

- ***Kollaborationsdiagramme***: zeigen dynamische Sichten eines UseCase, d.h. unter anderem den Austausch der Meldungen zwischen den Objekten, wobei die strukturelle Anordnung der implizierten Objekte im Vordergrund steht. Kollaborationsdiagramme können ohne Informationsverlust in Sequenzdiagramme übergeführt werden.
- ***State Chart Diagramme*** : zeigen eine dynamische Systemsicht, die sog. State Machines darstellt. Die State Machines beinhalten Objektzustände, Ereignisse, Transformationen und Aktivitäten.
- ***Aktivitäten-Diagramme*** : zeigen eine dynamische Systemsicht, die den Informationsfluss zwischen den unterschiedlichen Aktivitäten zeigt.
- ***Komponenten-Diagramme*** : zeigen die Organisation der Komponenten
- ***Umsetzungsdiagramme*** : zeigen die Kombination der Knoten und Komponenten.

4.1.4 Regeln

Die Regeln werden benötigt, um aus den verschiedenen Konzepten ein semantisch korrektes UML-Modell zu erstellen. Sie können sich auf folgende Elemente beziehen :

- **Namen** : welche Namen für die Bausteine, Beziehungen und Diagramme erlaubt sind.
- **Kontext** : der Kontext, in dem ein Namen verwendet wird, und welche semantische Bedeutung diesem Namen dadurch gegeben wird.
- **Sichtbarkeit** : wie diese Namen durch andere Benutzer sichtbar gemacht werden können.
- **Integrität** : wie die verschiedenen Bausteine miteinander verbunden werden, damit ein konsistentes Modell entstehen kann.
- **Ausführung** : wie ein dynamisches Modell ausgeführt wird und wie es zu interpretieren ist.

4.1.5 Mechanismen

Um konsistente UML-Datenmodelle zu erstellen, muss sich der Modellierer auf verschiedene vorgezeichnete Modelle stützen können. Die aus diesen Modellen hervorgegangenen Mechanismen werden in folgende Kategorien unterteilt :

- ***Spezifikationen*** : in UML wird zum Beispiel eine Klasse mit einem graphischen Symbol dargestellt. Damit die Semantik und die Syntax beschrieben werden können, kann eine verbale Spezifikation an ein Symbol angeheftet werden.
- ***Adornments*** : bestimmen die graphische Notation der Grundkonzepte.
- ***Gemeinsame Unterteilungen*** : verschiedene Bausteine können unterteilt werden, wie beispielsweise Klassen in Objekte, Schnittstellen in Implementierungen usw.
- ***Erweiterungsmechanismen*** : UML ist ein offenes Modell und kann durch den Benutzer erweitert werden, beispielsweise durch ***Stereotypen*** (Erweiterungen des UML-Vokabulars), ***gezeichnete Werte*** (Erweiterungen der Eigenschaften der Bausteine) oder ***Konsistenzbedingungen*** (Erweiterungen der Semantik eines Bausteins).

5 Einsatz der Grundkonzepte in den Diagrammen

5.1 Nutzung der Grundkonzepte in den Diagrammen

		Diagramme								
		Statisch			***Dynamisch***				***Statisch***	
Grundkonzepte		**Klassendiagramm**	**Objektdiagramm**	**Use Case Diagramm**	**Sequenzdiagramm**	**Kollaborations-diagramm**	**Statechart Diagramm**	**Aktivitätendia-gramm**	**Komponenten-diagramm**	**Umsetzungs-diagramm**
Strukturelle Bausteine	**Klasse**	+	-	-	-	-	-	-	-	-
	- Akteure	-	-	+	-	+	-	+	-	-
	- Objekte	-	+	-	+	+	-	+	-	-
	Schnittstelle	+	-	+	+	+	-	-	+	-
	Kollaboration	+	-	-	-	+	-	-	-	-
	Use Case	-	-	+	-	-	-	-	-	-
	Aktive Klasse	+	-	-	-	-	-	-	-	-
	- Aktive Objekte	-	+	-	+	-	-	-	-	-
	Komponente	-	-	-	-	-	-	-	+	-
	Knoten	-	-	-	-	-	-	-	-	+
Verhaltens-Bausteine	**Interaktion**	-	-	-	+	+	-	-	-	-
	State Machine	-	-	-	-	-	+	-	-	-
Grup. Bausteine	**Package**	+	-	-	-	-	-	-	-	-
Beschr. Bausteine	**Notiz**	+	+	+	+	+	+	+	+	+
Beziehungen	**Abhängigkeiten**	+	+	+	-	+	-	-	+	+
	Assoziationen	+	+	+	-	+	-	-	+	+
	Generalisationen	+	+	+	-	+	-	-	+	-
	Realisationen	+	+	+	-	+	-	-	+	-

Tabelle 8-1: Die Zuordnung der Grundkonzepte zu den Diagrammen.

Die oben aufgeführte Tabelle zeigt, welche Grundkonzepte für welche Diagramme benötigt werden. Dabei fällt auf, dass die verschiedenen Grundkonzepte in mehreren Diagrammen dargestellt werden. Die verschiedenen Diagramme stellen unterschiedliche Sichten des UML-Modells dar. Je nach den im IT-Entwicklungsprozess Beteiligten (Projektleiter, Modellierer, Programmierer), muss eine andere Sicht des UML-Modells gezeigt werden. Dem Projektleiter können beispielsweise nicht dieselben Aspekte gezeigt werden, wie den Programmierern.

Für die IT-Entwicklung ist es nicht unbedingt nötig, alle diese Diagramme zu erstellen. Die Grösse und die Komplexität (aus Sicht des IT-Systems oder aus organisatorischer Sicht) einerseits, und das Kommunikationsbedürfnis andererseits, bestimmen, welche Diagramme den System-Entwicklungsprozess vorantreiben oder nicht. Grundsätzlich kann davon ausgegangen werden, dass das UseCase-Diagramm und das Klassendiagramm immer dargestellt werden.

5.2 Abfolge der Diagramme

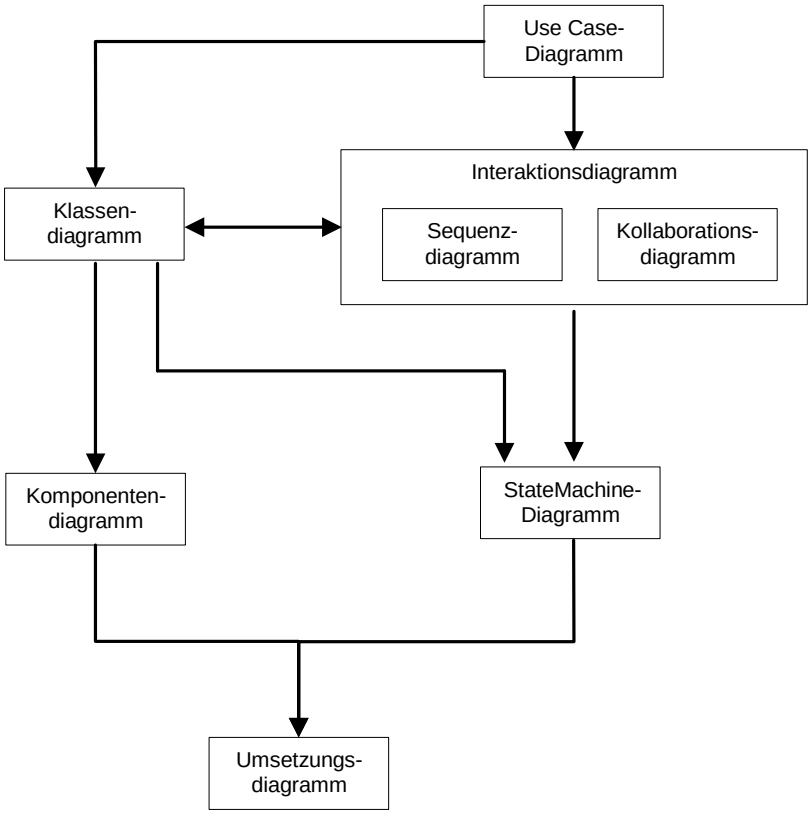


Abbildung 8-5 . Informationsfluss zwischen den Diagrammen

Das Ursprungsdiagramm ist eigentlich das UseCase-Diagramm. Es stellt die Fachprozesse dar, die im neuen Informationssystem abgebildet werden müssen. Aus ihm entsteht das Klassendiagramm, für den statischen Teil der UseCases, und dann ein Interaktionsdiagramm für den dynamischen Teil, wobei die Erstellung des Klassendiagramms stets vorangeht. Das Sequenzdiagramm und das Kollaborationsdiagramm sind eigentlich völlig redundant, es kann ohne Informationsverlust eines in das andere übergeführt werden, ohne das andere Informationsquellen dabei konsultiert werden. Der Entscheid, welches der beiden Diagramme wohl zutreffender ist, wird vom Blickwinkel der Modellierer bestimmt. Die Interaktionsdiagramme werden dann in State Machine-Diagramme (StateChart) übergeführt, die die Aktivitäten innerhalb eines Objekts beschreiben. Aus dem statischen Teil (Klassendiagramm) wird ein Komponentendiagramm erstellt. Schliesslich wird das Umsetzungsdiagramm mit Hilfe der statischen und dynamischen Beschreibung des UML-Modells angefertigt. Die Gesamtheit aller für ein Projekt wesentlichen Diagramme bilden das UML-Modell.

H Rational Unified Process (RUP)

1 Einleitung

Rational Unified Process (RUP) ist eine Software-Entwurfsmethode, die von der Firma Rational vertrieben wird. Ziel dieser Methode ist, eine qualitativ hochstehende Software zu erstellen, die den Bedürfnissen der Endbenutzer genügt. RUP soll den Rahmen bilden, in dem eine Softwareentwicklung innerhalb eines voraussehbaren Budgets und Terminplans durchgeführt werden kann.

RUP beschreibt ein iteratives Vorgehen, weil

- es selten möglich ist, die Anforderungen zu Beginn des Projekts vollständig definieren zu können, um sie dann zu entwickeln und umzusetzen;
- die laufend dazugewonnenen Kenntnisse aller Beteiligten direkt in die Entwicklung einfliessen können;
- die Risiken schneller identifiziert und minimiert werden können;
- den neuen Bedürfnissen der Fachleute schneller Rechnung getragen werden kann;
- der Kunde durch eine möglichst schnelle Veranschaulichung durch einen Prototypen die weiteren Anforderungen besser stellen kann.

Dieses iterative Vorgehen wird mit einem sorgfältigen Management der Bedürfnisse kontrolliert, um die Kunden über eventuelle durch neue Bedürfnisse entstandene Budget- oder Terminplanüberschreitungen informieren zu können.

RUP unterstützt OO-Entwurfsmodelle und benutzt zur Formulierung die Beschreibungssprache UML.

2 Bibliographie

Kruchten, Philippe (1999) : The Rational Unified Process, An Introduction. Addison Wesley Longman Inc., Massachusetts.

On-line Dokumentation der Firma Rational

3 Prozessarchitektur

RUP unterscheidet zwei Dimensionen in der Softwareentwicklung :

- Die zeitliche Abfolge eines Projektes : Gliederung in ***Phasen***
- Die inhaltliche Gliederung eines Projektes : Gliederung in ***Prozesse***

Die Gliederung in Prozesse betrachtet den statischen Aspekt der Softwareentwicklung und die Gliederung in Phasen den dynamischen Aspekt.

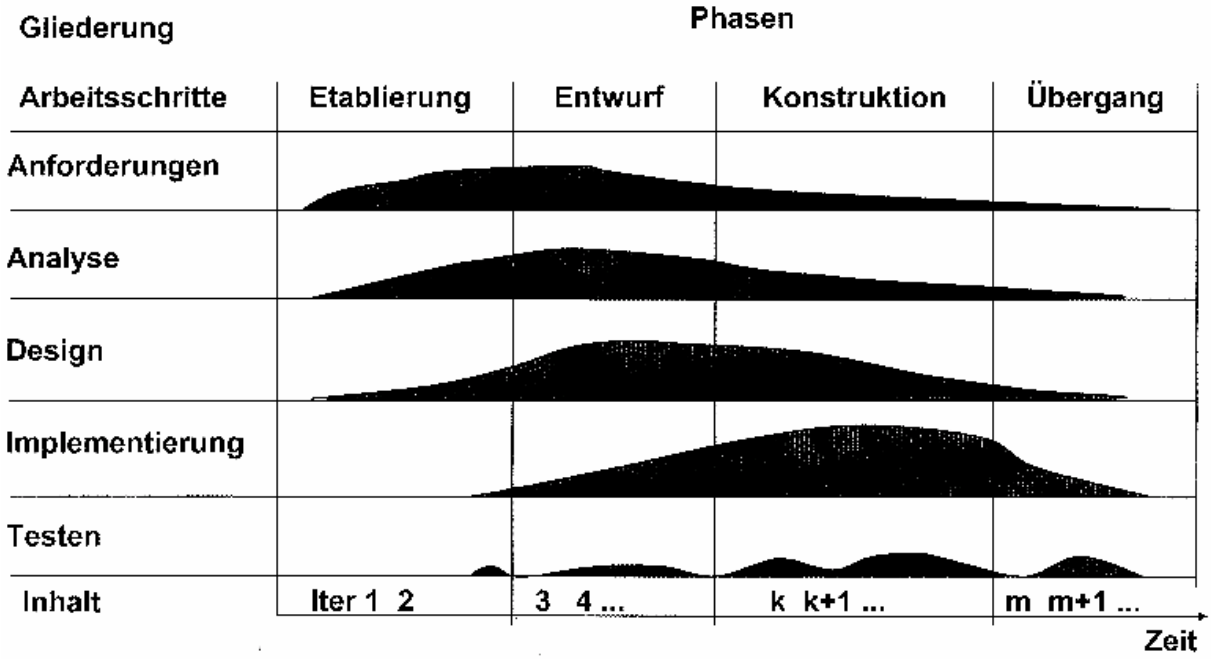


Figure 8-6 : Zuteilung der Prozesse an die Phasen eines Projektes. (Schema aus der Dokumentation RUP)

Die Figure 8-6 zeigt, dass mehrere Prozesse gleichzeitig laufen können, und dass sie nicht eindeutig einer Phase zugeordnet werden müssen. Dies ist vor allem durch die Iterationen zu erklären. Jede Iteration muss mehrere Prozesse durchlaufen, die die neuen Anforderungen des Kunden in die Software umsetzt.

4 Die statische Struktur : Die neun Hauptprozesse

Die verschiedenen Prozesse werden mit Hilfe folgender Grundeinheiten beschrieben :

- Arbeiter: eine Rolle, die eine oder mehrere Personen in einem Fachbereich innehaben
- Aktivität: eine Arbeitseinheit, die ein Arbeiter ausführen kann
- Artefakt: eine Informationseinheit, die benötigt oder produziert wird, und die einen gewissen Verantwortungsbereich definiert; kann ein Schema oder ein Dokument sein.
- Workflows (Arbeitsfluss): Eine Folge von Aktivitäten in einem Fachbereich, deren Ergebnis einen Arbeiter von Interesse ist.

In jedem hier aufgeführten Hauptprozess werden die nötigen Arbeiter und ihre Aufgaben, die zeitliche Abfolge der Arbeiten (Workflow) und die zu erwartenden Resultate (Artefakts) von RUP beschrieben.

4.1 Die Modellierung der Fachprozesse

Die Modellierung der Fachprozesse ist vor allem nützlich, um die Struktur und die Dynamik einer Organisation besser verstehen zu können. Es ist vor allem wichtig, dass sich der Datenbank-Entwerfer, die Kunden und die Endbenutzer auf ein gemeinsames Verständnis der Organisation einigen können. Dieses Verständnis und das daraus resultierende gemeinsame Vokabular vereinfachen die einheitliche und eindeutige Formulierung der Systemanforderungen.

Der Kunde oder andere interessierte Akteure können hier zum ersten Mal mit formalisierten Schemen wie zum Beispiel Use Cases konfrontiert werden. Es ist für sie einfacher, diese Schemen an bekannten Fachprozessen anzuwenden und sie dadurch verstehen zu lernen, als an abstrakten Anforderungen.

4.2 Formulierung der Anforderungen

Die Formulierung der Anforderungen muss unmissverständlich sein, so dass alle Beteiligten sich ohne jegliche Vorbehalte damit einverstanden erklären können. Sie muss auf dem gemeinsamen Vokabular beruhen und schematisiert werden. Durch diese Anforderungen werden einerseits die Ziele und andererseits die Grenzen des zu entwickelnden Informationssystems (oder einer Software) festgelegt.

RUP besteht (zu Recht) darauf, dass die inhaltliche Festlegung der Iterationen und die nötigen Benutzerschnittstellen in diesem Hauptprozess bestimmt werden.

4.3 Analyse und "Design"

Die Analyse und das "Design" werden in einem Hauptprozess zusammengefasst. Die Analyse ist der Entwurf aller funktionalen Systemanforderungen in einer für den Entwerfer eindeutige Sprache. Der Entwerfer stellt sicher, dass keine Benutzeranforderungen ausser Acht gelassen werden. Funktionale Anforderungen werden von Kunden und vom System-Analysten formuliert. Das Design befasst sich zudem mit den nicht funktionalen Anforderungen, wie beispielsweise Wiederverwendbarkeit, Zuverlässigkeit, Performance und Portabilität. Die Unterscheidung zwischen Analyse und Design wird von RUP nicht als fundamental betrachtet.

Aus diesem Entwurfsprozess geht eine Spezifikation hervor, die die Anforderungen an die Systemgegebenheiten (Programmier-Tool, Hardware etc.) berücksichtigt.

4.4 Implementierung

Ziel der Implementierung ist die Erarbeitung des Codes. Der Code kann entweder von einem Programmierer erstellt werden, oder aber auch von Case-Tools generiert werden. Demzufolge muss die Erstellung des Codes je nach Kenntnisse der Entwickler oder der zur Verfügung stehenden Tools organisiert werden. RUP schlägt die Unterteilung des Gesamtcodes (der Klassen und der Objekte) in Komponenten vor, derart dass sie mehrfach genutzt werden können (reuse). Die verschiedenen Komponenten werden getestet, sobald sie erstellt worden sind und schliesslich in ein lauffähriges Gesamtsystem integriert.

4.5 Test

Der Testprozess untersucht das Gesamtsystem als solches. Die richtige Integration der Softwarekomponenten werden kontrolliert und deren Interaktionen aufgezeigt. Zudem wird das Gesamtprodukt mit den Anforderungen des Kunden konfrontiert, um zu vermeiden, dass Anforderungen nicht implementiert worden sind. Dieser Prozess stellt sicher, dass alle Fehler im Gesamtsystem behoben werden, bevor es beim Kunden eingeführt wird.

4.6 Zusätzliche Prozesse

4.6.1 Change-Management

Ziel des Change-Management ist die ständige Überwachung aller Einheiten, seien es Software-Einheiten oder Konzeptionsdokumente, um deren Integrität während ihrem Lebenszyklus zu gewährleisten.

Das Change-Management besteht aus folgenden drei Haupttätigkeiten:

- Es beinhaltet die sachliche Gruppierung von Konfigurationseinheiten. Man gruppiert Softwareeinheiten und deren Versionen, die zur Verwirklichung einer Produktekomponente beitragen. Diese Produktekomponenten werden schrittweise in eine ausführbare Hauptkomponente (Release) integriert.
- Das Change-Management verwaltet die ständig neu auftretenden Anforderungen aller interessierten Akteure und wägt deren Auswirkungen auf andere Komponenten ab.
- Das Change-Management liefert alle für das Projektmanagement nützlichen Informationen an den Projektleiter, wie etwa den Stand der Arbeiten, neue Anforderungen, eventuell aufgetauchte Probleme und die neuen Technologie-Trends auf dem Markt.

Einerseits werden die Anforderungen und die betroffenen Komponenten verwaltet, andererseits muss jede neue Anforderung mit den im Pflichtenheft aufgeführten Leistungen verglichen werden und der Kunde muss auf eventuelle Mehrkosten oder auf Zielabweichungen aufmerksam gemacht werden.

4.6.2 Projekt-Management

Das Projekt-Management liefert die nötigen Rahmenbedingungen, um Software-intensive Projekte termingerecht und kostengünstig durchführen zu können, indem durch Richtlinien für Planung, Betreuung, Ausführung und Projektüberwachung das Risiko minimiert wird.

In Projekten, in denen Objekt-orientierte Entwurfsmethoden angewandt werden, muss das Projekt Management inbesondere die Anzahl der Iterationen und die Dauer jeder Iteration festlegen.

4.6.3 Bestimmung der Rahmenbedingungen

Die Bestimmung der Rahmenbedingungen beinhaltet alle organisatorischen Massnahmen, die für eine gewünschte Projektentwicklung ideale Voraussetzungen schaffen.

Darunter kann man folgende Aktivitäten zählen :

- Programmier-Tool (Evaluation und Beschaffung)
- Anpassung des Programmier-Tools an das Projekt
- Anpassung des Projektablaufs, unter anderem RUP an ein gegebenes Projekt anpassen.
- Ausbildung
- Technische Dienste wie Backup, Administration usw.

5 Die dynamische Struktur : Die Phasen

RUP empfiehlt ein iteratives Vorgehen, um Projektrisiken minimal halten zu können und um den häufig erweiterten Benutzeranforderungen fortlaufend Rechnung tragen zu können.

Das Projekt kann in vier Phasen unterteilt werden :

- Etablierung
- Entwurf
- Konstruktion
- Übergang

Jede dieser Phasen kann mehrere Iterationen enthalten. Die Anzahl der Iterationen muss vom Projektmanager bestimmt werden. Sie hängt von der Komplexität des Projektes, der Erfahrung aller Beteiligten usw., kurz von den Risikofaktoren ab. Risikoreiche Aspekte sollten relativ früh in einen Prototypen implementiert werden, um erstens die weiteren Risiken abschätzen zu können und eventuelle Änderungen oder Kompromisse in den Anforderungen definieren zu können. Die Kosten der Änderungen aufgrund neu definierter Anforderungen steigen stark an, je später sie erfolgen, im Verlaufe der Zeit exponentiell an.

Jede einzelne Iteration läuft die vier Prozesse ab: Anforderungsanalyse, Design, Codierung und Integration. Diese vier Prozesse werden je nach der Lebensphase gewichtet, in der eine Iteration durchgeführt wird. In der Voranalyse beispielsweise werden der Codierung und der Integration kaum Beachtung geschenkt.

Nach jeder Lebensphase eines Projekts muss ein Meilenstein gesetzt werden. Es ist klar, dass ein iteratives Vorgehen schwieriger zu planen ist, als ein lineares Vorgehen, wie beispielsweise im Strassenbau. Dadurch sind klar definierte Meilensteine, die an definierten Zeitpunkten die Beendigung bestimmter Arbeiten verlangen, noch wichtiger als in einem traditionnellen Vorgehen. Diese Meilensteine sind wichtig, um den Projektablauf kontrollieren zu können, damit das Projekt zeitlich und finanziell den Rahmen nicht sprengt.

5.1 Etablierung

In der Etablierung werden vor allem die Ziele des Projektes und seine Grenzen mit Hilfe des Kunden definiert. Der Auftragnehmer lernt die Organisation und die verschiedenen Fachprozesse des Kunden verstehen, um mit ihm das Projekt und seine Anforderungen zu definieren. Zudem wird das Projekt organisiert, d. h. Termine, Kosten, Ressourcen, Softwareeinkäufe und mögliche Wiederverwendung von Systemkomponenten (auch in anderen Projekten) werden geschätzt.

In der Etablierung steht die Kommunikation zwischen Kunden und Auftragnehmer im Vordergrund. Es entsteht ein gegenseitiger Wissensaustausch.

5.2 Entwurf

In der Entwurfsphase finden wir die eigentlichen Tätigkeiten des Softwareingenieurs. Die Architektur wird festgelegt und die Anforderungen werden formalisiert und ergänzt. Die für das Projekt nötigen UML-Schemata werden erstellt. Zudem werden die strategischen und risiko-reichen Anforderungen identifiziert und durch einen evolutiven Prototypen umgesetzt. Dadurch können einerseits Gefahren und Risiken minimiert werden und andererseits kann ein genauerer Terminplan erstellt werden.

5.3 Konstruktion

In der Konstruktionsphase werden alle Anforderungen in Komponenten umgesetzt. Diese Komponenten werden in ein Gesamtsystem integriert und getestet. Das Ergebnis der Konstruktionsphase ist ein operationelles System, das beim Kunden eingesetzt werden kann und ihn vollumfänglich zufrieden stellt.

Der kontinuierliche Kundenkontakt und die richtige Wahl der Release-Zyklen ist ein wichtiger Faktor für die Zufriedenheit der Kunden. Werden die Release-Zyklen zu kurz gewählt, entsteht zusätzlicher Aufwand. Sind die Zyklen jedoch zu lang, werden die Anforderungen pro Zyklus umfangreicher und die Erfüllung der Kundenerwartungen wird schwieriger.

5.4 Übergang

In der letzten Phase wird die Applikation beim Kunden installiert. Es werden die nötigen Dokumente für Administrator und Endbenutzer erstellt und die entsprechende Ausbildung durchgeführt. Die vom Endbenutzer während dem Beta-Test festgestellten Mängel werden behoben, oder geben Anlass, ein neues (Teil-)Projekt festzulegen.

I HERMES

1 Einleitung

HERMES (**H**andbuch der **E**lektronischen **R**echenzentren des Bundes, eine **M**ethode für die **E**ntwicklung von **S**ystemen) ist ein Instrument zur Verwaltung und Durchführung von Informatikprojekten. Sie wird bereits seit 1975 in der Bundesverwaltung benutzt. Im Jahre 1986 wurde erstmals eine grössere Revision durchgeführt. Das somit entstandene HERMES-86 wurde für alle Informatikprojekte obligatorisch. Eine erneute Revision wurde im Jahre 1993 unerlässlich, um dem technologischen Fortschritt Rechnung zu tragen. Dies führte dann im Jahre 1995 zu HERMES-95.

HERMES bildet die Grundlage für die Projektleitung und für die Projektteams zur erfolgreichen Durchführung eines Informatikprojekts.

2 Bibliographie

Bundesamt für Informatik (1995) : HERMES, Conduite et déroulement de projets informatiques. OCFIM; Bern.

3 Phasenmodell von HERMES-86

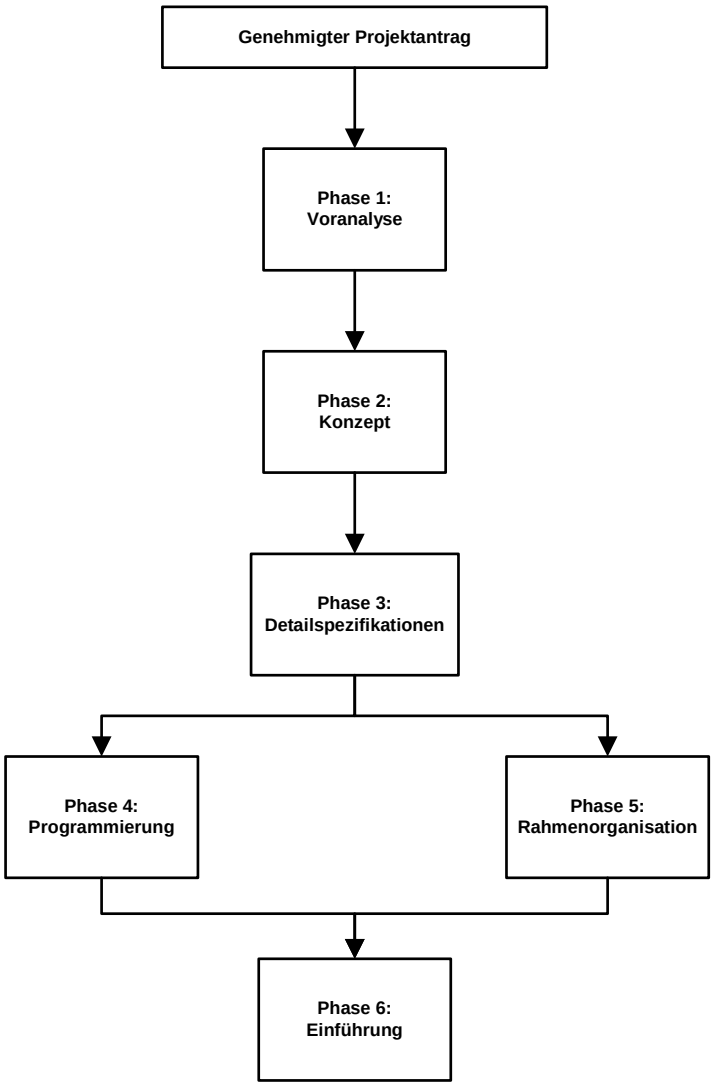


Figure 8-7: Vorgehen von HERMES-86 für Informatikprojekte

HERMES-86 gliedert ein Informatikprojekt in sechs Phasen mit folgendem prozentualen Aufwand:

Vorgehensphasen	**Aufwand**
1. Voranalyse	3-6%
2. Konzept	10-20%
3. Detailspezifikationen	20-35%
4. Programmierung	25-35%
5. Rahmenorganisation	10-15% (ohne Datenersterfassung)
6. Einführung	5-15%

Diese Angaben stammen aus dem Jahre 1986 und sind somit mit Vorsicht zu geniessen.

4 Phasenmodell von Hermes-95

4.1 Allgemeines

	Aspekte						*Submodelle*				
Phasenmodell	*Übersicht*	*Vorgehen*	*Ergebnisse*	*Entscheide*	*Kompetenzen*		*Rolle im Projekt*	*Projekt Management*	*Qualitäts-management*	*Konfigurations-management*	
Phase 1: Initialisierung											
Phase 2: Vorstudie											
Phase 3: Konzept											
Phase 4: Realisierung											
Phase 5: Einführung											

Submodelle					
Rolle im Projekt					
Projekt-Management					
Qualitätsmanagement					
Konfigurationsmanagement					

Figure 8-8: Das Phasenmodell von HERMES-95

HERMES-95 unterteilt die Systementwicklung in fünf Phasen:
1. Initialisierung
2. Vorstudie
3. Konzept
4. Realisierung
5. Einführung

Diese Phasen sind unabdingbar für die erfolgreiche Entwicklung eines Systems. Zudem werden vier Submodelle erkannt, die den Ablauf eines Projekts bezüglich der Rahmenbedingungen (Budget, Ressourcen) gewährleisten: Rolle im Projekt (Projekt-Organisation), Projektmanagement, Qualitätsmanagement und Konfigurationsmanagement. Diese Submodelle sind Aktivitäten die während des gesamten Projekts, d.h. während allen Phasen, ausgeübt werden müssen.

HERMES-95 behandelt für jede Phase und die Submodelle folgende Aspekte:
- Übersicht
- Vorgehen
- Ergebnisse
- Entscheide
- Kompetenzen

4.2 Phase 1: Initialisierung

Ziel der Phase 1 ist es, eine Grundlage zu schaffen, die es ermöglicht, eine Projektidee in die Tat umsetzen zu lassen. Sie definiert die technischen und organisatorischen Rahmenbedingungen und plant die nötigen Schritte zur Durchführung des Projektes.

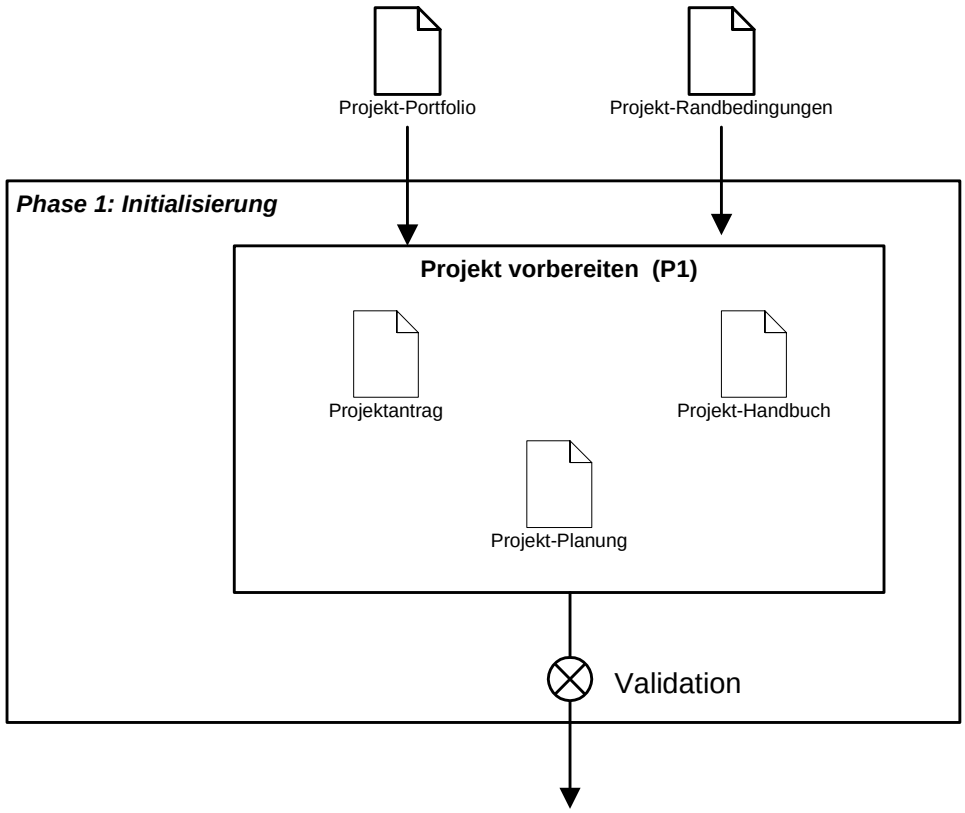


Figure 8-9: Das Vorgehen während der Phase 1 "Initialisierung".

Die Initialisierungsphase folgt nach der Ausschreibung des Projekts, nach der Erstellung des Angebots und der Arbeitsvergabe. Diese vorbereitenden Arbeiten enthalten bereits generelle Angaben zur Planung der Durchführung des Projekts und zu den notwendigen Ressourcen. Die dazu bereits erstellten Dokumente (u.a. Angebot) werden in das Projektportfolio klassiert. Das Portfolio kann als Verzeichnis aller ausstehenden Projekte festgelegt werden. Zudem müssen gegebene Randbedingungen wie vertragliche Vereinbarungen, Gesetz, bestehende IT-Architektur usw. in der Initialisierungsphase berücksichtigt werden.

Diese Initialisierungsphase muss folgende Resultate liefern:

- ***Projektantrag***: Er dient als Diskussionsgrundlage für das weitere Vorgehen. Er enthält u.a. Ausgangssituation, Ziele, Planung und Organisation.
- ***Projekthandbuch:*** Es dient als Handlungsgrundlage. Beschrieben werden Aspekte des spezifischen Vorgehens im Projekt, sowie technische und organisatorische Gegebenheiten.
- ***Projektplanung***: Sie definiert den Ablauf und die Organisation des gesamten Projekts. Sie wird kontinuierlich nachgeführt.

Die Phase wird dadurch abgeschlossen, dass die Projektleitung und der Kunde das Projekthandbuch und die Projektplanung genehmigen.

4.3 Phase 2: Vorstudie

Ziel der Phase 2 ist die Erstellung der zur Bestimmung des weiteren Vorgehens notwendigen Grundlagen. Zudem muss die Vorstudie zeigen, welche der in der Initialisierungsphase bezeichneten Teilprojekte unwirtschaftlich sind, damit sie rechtzeitig abgebrochen werden können.

Die Phase 1 übergibt der Phase 2 "Vorstudie" das Projekt-Handbuch und die Projekt-Planung. Diese werden im Laufe der Vorstudie aktualisiert.

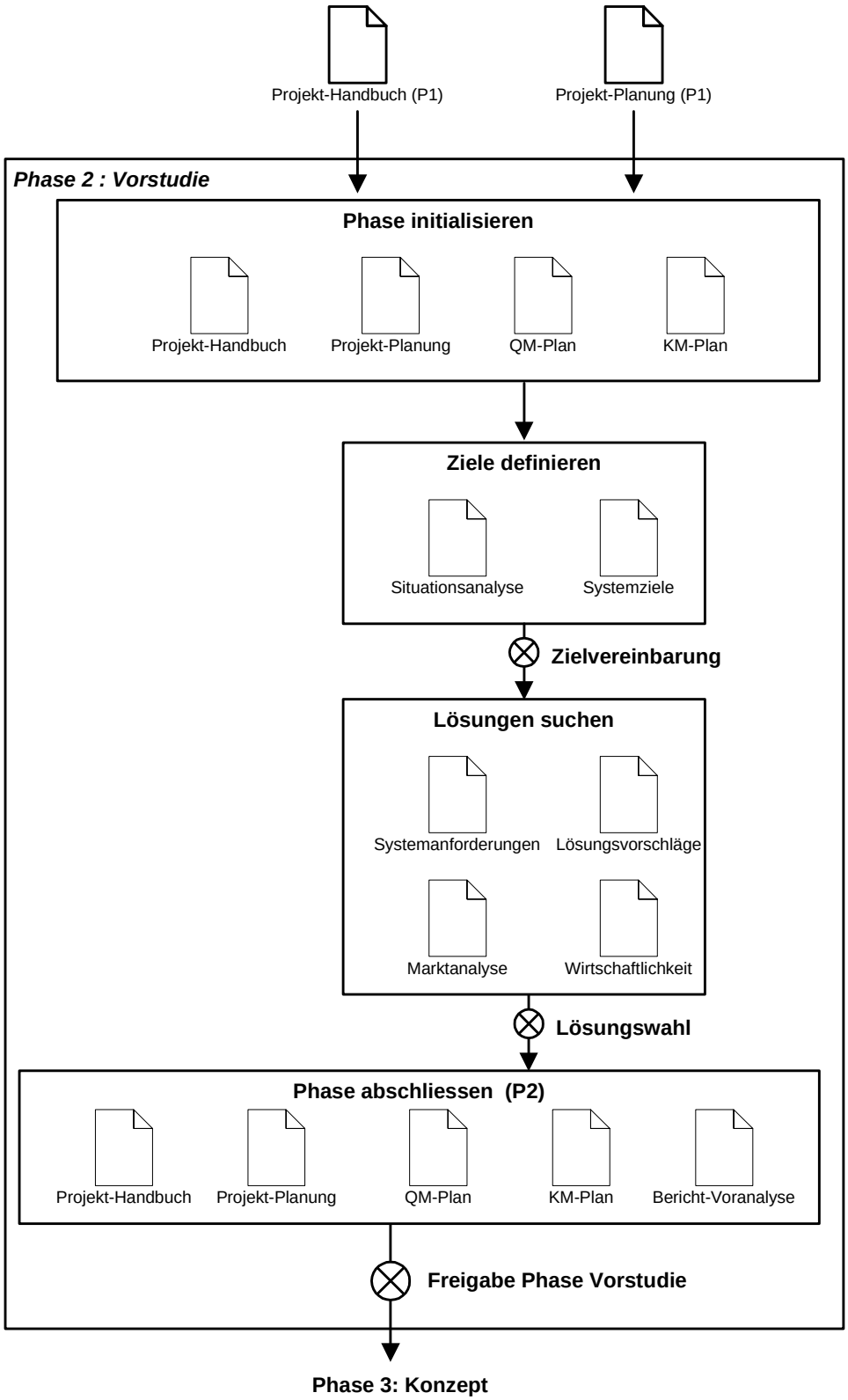


Figure 8-10: Das Vorgehen während der Phase 2 "Vorstudie".

In grösseren Projekten werden zu Beginn der Phase 2 noch folgende Dokumente erstellt:

- Qualitätsmanagementplan (QM-Plan): besteht aus Verfahren, um die Qualität während eines Projekts sicherstellen zu können. Die Qualität muss durch Gegenüberstellen der Anforderungen und der entwickelten Produkte (Systeme oder auch Dokumente) bestimmt werden.
- Konfigurationsmanagementplan (KM-Plan): besteht aus Verfahren, die die Handhabung der Versionen und Releases der Produkte und deren Schnittstellen bestimmen.

Auf Grund der in dieser Phase durchgeführten Situationsanalysen werden Systemanforderungen formuliert und nach Lösungen gesucht. Die Markt- und die Wirtschaftlichkeitsanalyse werden zudem beigezogen, um die Lösungsvorschläge evaluieren zu können und dann schliesslich auch einen Entscheid fällen zu können.

Der Bericht "Voranalyse" fasst alle Ergebnisse und Entscheide aus dieser Phase zusammen. Auf dieser Grundlage wird die Freigabe der folgende Phase 3 "Konzept" entschieden.

4.4 Phase 3: Konzept

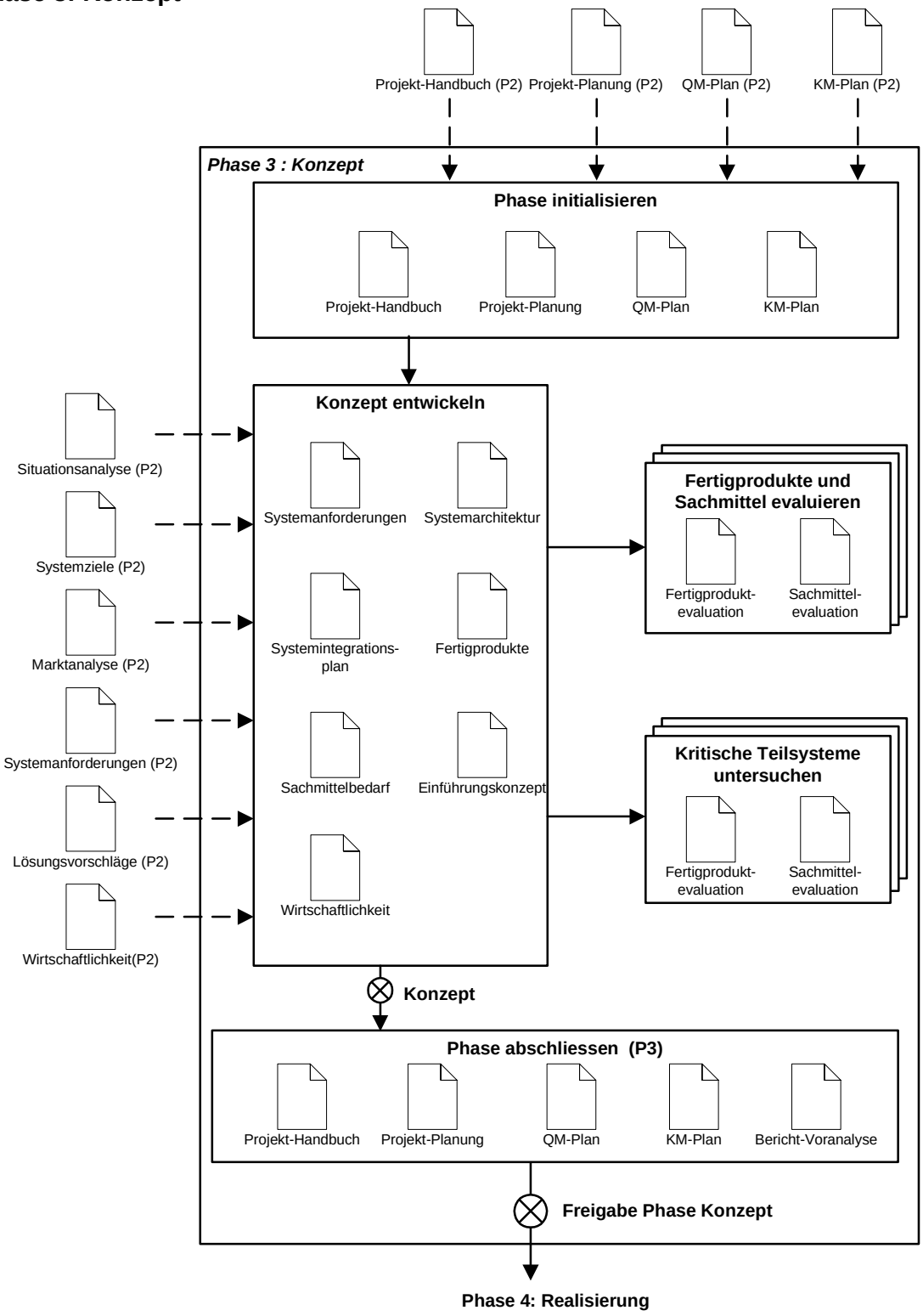


Figure 8-11: Das Vorgehen während der Phase 3 "Konzept".

Ziel der Phase 3 "Konzept" ist die Detaillierung der in der Vorstudie gewählten Lösungen, und ermöglicht eine gestützte Systembeschreibung. Das Konzept wird später an die Entwickler in der Realisierungsphase abgegeben, damit sie die Systemanforderungen umsetzen können.

Während der Konzeptphase müssen folgende Dokumente angefertigt werden:

- ***Systemarchitektur***: organisatorische und technische Architektur des Systems, hinunter bis zur kleinsten Konfigurationseinheit.
- ***Sachmittelbedarf***: benötigte Hardware und deren Kosten, ein wesentlicher Aspekt für die Wirtschaftlichkeit.
- ***Einführungskonzept***: beschreibt die organisatorische und technische Machbarkeit der Migration, vom existierenden System zum Neuen und dessen Einführung.
- ***Detailstudie***: beschreibt die Einzelheiten kritischer Aspekte, damit das Risiko einer Fehlentwicklung minimal gehalten werden kann.
- ***Fertigproduktevaluation***: vergleicht die schon bestehenden Produkte und bereitet deren Beschaffung vor.
- ***Sachmittelevaluation*** :vergleicht die verschiedenen Sachmittel (Hardware) und bereitet deren Beschaffung vor.
- ***Systemintegrationsplan***: beschreibt Richtlinien zur Konstruktion des Systems (Zusammenführen der Systemkomponenten) unter Berücksichtigung der Qualitätssicherung.
- ***Fertigprodukte***: Die aufgrund der Fertigproduktevaluation beschafften Fertigprodukte werden zu Systemkomponenten, die spezifische Randbedingungen darstellen können. Diese Randbedingungen müssen während der Konzeptphase berücksichtigt werden.
- ***Prototype***: werden dann eingeführt, wenn einige kritische Aspekte einer Systemeinheit näher untersucht werden müssen. Prototypen ermöglichen es, das Konzept und dessen Spezifikationen zu bestätigen oder zu verwerfen.

Der Bericht "Konzept" fasst alle Ergebnisse dieser Phase zusammen, und bildet die Grundlage für den Entscheid zur Freigabe der nächsten Phase.

4.5 Phase 4: Realisierung

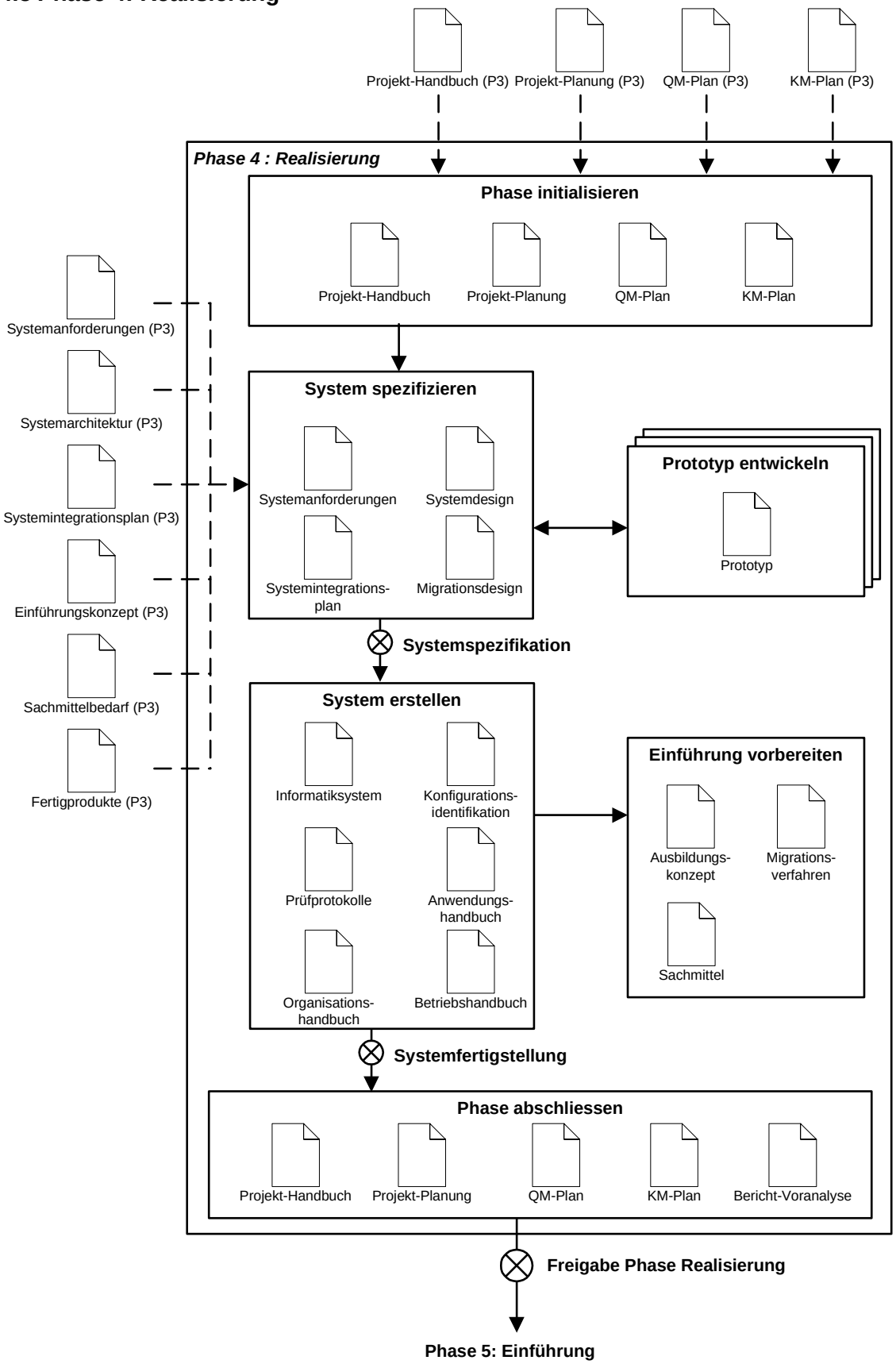


Figure 8-12: Das Vorgehen während der Phase 4 "Realisierung".

Ziel der Phase "Realisierung" ist die Erstellung des Systems, so dass es eingeführt werden kann. Zudem sollten in dieser Phase bereits Migrationsprozeduren und einige Grundlagen zur schrittweisen Einführung des Systems erstellt werden.

Während der Realisierungsphase müssen folgende Dokumente angefertigt werden:

- ***Ausbildungskonzept***: Anforderungen an die Ausbildung der Benutzer und der Betreiber.
- ***Migrationskonzept***: definiert organisatorische und technische Anforderungen an das Migrationsverfahren
- ***Systemdesign***: beschreibt die Systemarchitektur in ihren Einzelheiten. Sie stellt das Ergebnis aus der Konzeptphase bezüglich Systemarchitekur dar.
- ***Betriebshandbuch***: beinhaltet alle für den Betreiber nötigen Informationen, um den Betrieb des Systems sicherzustellen. Dazu gehören auch Anweisungen im Falle einer Störung .
- ***Organisationshandbuch***: beschreibt die Integration des Systems in eine (Rahmen-) Organisation. Das System wird bestehende Aktivitäten innerhalb einer Organisation ändern.
- ***Benutzerhandbuch***: beinhaltet alle für den Benutzer nötigen Informationen, um das System korrekt nutzen zu können. Dazu gehören auch Anweisungen im Falle einer Störung .
- ***Sachmittel***: beschreibt die zur Verfügung stehende Hardware, um die Installation des Systems vorbereiten zu können.
- ***Migrationsverfahren***: dokumentiert das Vorgehen, wie das alte System in das Neue umgewandelt werden kann.
- ***Prototyp***: werden dann eingeführt, wenn einige kritische Aspekte einer Systemeinheit näher untersucht werden müssen. Prototypen ermöglichen es, das Konzept und dessen Spezifikationen zu bestätigen oder zu verwerfen. In der Realisierungsphase ist es möglich, einen Prototypen so zu erweitern, dass er als operationelle Einheit innerhalb des Systems eingesetzt werden kann.
- ***Informatiksystem***: stellt die Gesamtheit der operationellen Einheiten und deren Schnittstellen dar, die aufgrund der Spezifikationen entwickelt worden sind.

Die Phase kann dann abgeschlossen werden, wenn einerseits das Informatiksystem durch den Kunden genehmigt wird und andererseits Projekthandbuch, Projektplan, QM-Plan und KM-Plan akzeptiert werden.

4.6 Phase 5: Einführung

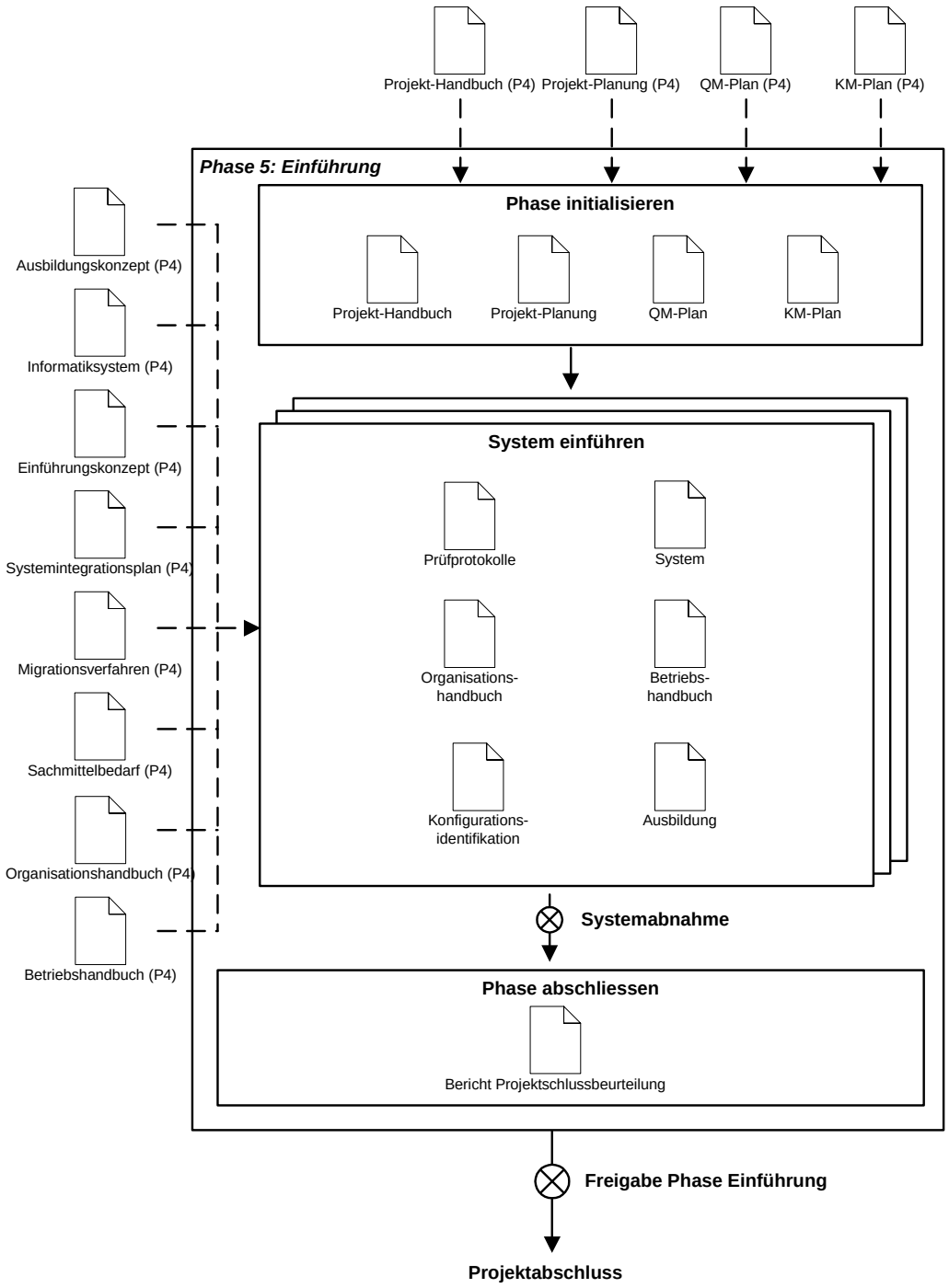


Figure 8-13: Das Vorgehen während der Phase 4 "Einführung".

Ziel der Einführungsphase ist die Installation und die Inbetriebnahme des Systems. Der Übergang vom alten System in das Neue muss sichergestellt werden.

Folgende Dokumente müssen erstellt werden :

Bericht Projektabschlussbeurteilung: besteht aus der Bilanz der vorgesehenen und der effektiven Kosten, der Einhaltung der Fristen, und der Beschreibung des effektiven Vorgehens. Daraus können die gemachten Erfahrungen formuliert und analysiert werden.

Das **Betriebshandbuch**, das **Organisationshandbuch**, die **Systembeschreibung** und die **Konfigurationsidentifikation** müssen nachgeführt werden und die Ausbildung der Benutzer muss sicher gestellt werden.

Das Projekt kann abgeschlossen werden, falls die Einführung des Systems in die Organisation und die Ausbildung der Benutzer erfolgt sind und der Projektabschlussbericht genehmigt wird.

5 Fazit

HERMES ist eine Vorgehensweise, die dem "Wasserfall"-Prinzip entspricht. Obwohl innerhalb einer Entwicklungsphase Iterationen möglich sind, werden die Hauptphasen Initialisierung, Vorstudie, Konzept, Realisierung und Einführung zeitlich linear ausgeführt. Die detaillierte Beschreibung der Phasen und der zu erwartenden Ergebnisse ermöglichen eine realistische Planung und termingerechte Ausführung von Informatikprojekten.

Lang andauernde Projekte, die nach HERMES durchgeführt werden, leiden unter der relativ statischen Struktur der Vorgehensweise, die den Einbezug von neuen Anforderungen und von neuen Technologien nicht erlaubt, die während dieser Periode auftreten.

Dennoch gilt HERMES als sichere Methode, Informatikprojekte zu planen und solche Projekte unter Berücksichtigung finanzieller und terminlicher Einschränkungen durchzuführen.

J Beschreibung der Konsistenzbedingungen

1 Einleitung

In den heutigen Beschreibungssprachen wurde das Hauptgewicht in die Formalisierung der Daten und der Auswertungen gelegt. Die Formalisierung der Konsistenzbedingungen wurde eher vernachlässigt, da diese in der ersten Konzeptionsphase nicht relevant erschienen. Vielfach werden sie in natürlicher Sprache wiedergegeben und von Programmierer in eine Programmiersprache umgesetzt. Dies führt vielfach zu Fehlentwicklungen, da die Formulierungen nicht eindeutig sind.

Um die Kommunikation dieser Bedingungen, vom Kunden über den Entwerfer bis hin zum Programmierer, sicherstellen zu können, braucht es eine formale Sprache, die nur eine einzige Interpretation zulässt.

Erste Versuche, die nötigen Sprachelemente zur Verfügung zu stellen, wurden unter anderem in INTERLIS und in OCL unternommen.

2 Bibliographie

Bundesamt für Raumplanung, Eidgenössische Vermessungsdirektion (1998) : INTERLIS, Ein Datenmodellierungs- und Austauschmechanismus für Geo-Informationssysteme, Draft, Version 2, Revision 1.

Object Management Group (1999):OMG Unified Modeling Language Specification, Version 1.3 alpha R5; 6: Object Constraint Language.

3 Inventar der Konsistenzbedingungen

3.1 Durch das Datenmodell definierte Konsistenzbedingungen

3.1.1 Optionale Attribute

Optionale Attribute sind Attribute, die nicht unbedingt einen Wert enthalten müssen.

3.1.2 Kardinalitäten

Kardinalitäten charakterisieren eine Beziehung zwischen zwei Informationsobjekttypen.

Beispiel : Eine Person besitzt kein oder mehrere Autos (0:n).

Ein Auto gehört einer oder mehreren Personen (1:n).

Mit diesen zwei Aussagen wird die Beziehung zwischen den Informationstypen Auto und Person klar festgelegt.

3.1.3 Attribut-Typ / -Format

Der Wertebereich eines Attributs ist die Gesamtheit aller möglichen Werte.

Es unterscheidet grundsätzlich fünf Typen von Wertebereichen :

- Text
- Zahl
- Koordinatenpaar
- Datum/Zeit
- Boolean

3.1.4 Wertebereiche von Text-Attributen

Der Wertebereich eines Textattributs kann durch folgende Angaben beschrieben werden:

- Maximale Anzahl von Buchstaben
- Nur Grossbuchstaben
- Eine Aufzählung möglicher Werte (z.B. Farbe (rot, blau, gelb))

3.1.5 Wertebereiche von numerischen Attributen

Der Wertebereich eines numerischen Attributs kann durch folgende Angaben beschrieben werden:

- N,M (N: Vorkommastellen, M: Nachkommastellen)
- Intervall (kleinst möglicher Wert, grösst möglicher Wert)
- Aufzählung (z.B. 1, 2, 3, 4)
- Zahlentyp (z.B. ganze Zahl, natürliche Zahl)

3.1.6 Zeitliche Attribute

Der Wertebereich eines zeitlichen Attributs kann durch folgende Angaben beschrieben werden:

- Typ (vierstellige Jahrbezeichnung, mit oder ohne Zeitangabe, mit oder ohne Datumsangabe)
- Intervall

3.1.7 Boolean Attribute

Der Wertebereich eines Boolean-Attributs besteht aus zwei Werten: "Wahr", "Falsch".

3.2 Abhängige Attribute

Ein Wertebereich eines Attributs eines Objektes kann von einem Wert eines anderen Attributs anhängen: Beispielsweise kann der Beginn einer Gültigkeit eines Objektes nicht später erfolgen als das Ende dieser Gültigkeit.

Ein Wertebereich eines Attributs eines Objektes kann von einem Wert desselben Attributs eines anderen Objektes abhängen: Beispielsweise kann eine solche Bedingung die Bildung von zyklischen Beziehungen verhindern.

Ein Wertebereich eines Attibuts eines Objektes kann von einem Wert eines anderen Attributs eines anderen Objektes abhängen. Beispielsweise kann im RBBS (gemäss SN 640 910) eine Distanz eines Objektes zum vorherliegenden Bezugspunkt nicht grösser sein als die Länge des entsprechenden Sektors.

3.3 Operatoren

Operatoren sind Symbole, die zwei Attribute oder ein Attribut mit einem Wert vergleichen lassen. Unter den gängigen Operatoren sind unter anderem '=' oder '>' vorzufinden.

4 Beschreibungselemente zur Darstellung der Konsistenzbedingungen

4.1 INTERLIS

4.1.1 Allgemeine Bemerkungen

INTERLIS ist einerseits ein Austauschmechanismus für Geo-Informationssysteme und andererseits eine Datenbeschreibungssprache, die die Daten in einer System-unabhängigen Form beschreiben lässt. INTERLIS wurde von der Schweizerischen Normen-Vereinigung als Norm SN 612'030 herausgegeben. Unterdessen wurde INTERLIS immer weiter entwickelt und eine Version 2.0 in Form eines Drafts publiziert. In diesem Anhang beziehen wir uns auf die Version 2.0, da die normierte Version 1.0 relativ wenig zur Formalisierung der Konsistenzbedingungen beiträgt.

Konsistenzbedingungen sollten in der Datenbeschreibung festgehalten werden. Im allgemeinen werden sie für die Informationsobjekt-Typen beschrieben, die sie auch betreffen. Es ist aber möglich, dass komplizierte Konsistenzbedingungen zuerst als Prozedur beschrieben und dann nur beim entsprechenden Informationsobjekt-Typen aufgerufen werden.

INTERLIS wurde von Anfang an so konzipiert, dass geographische bzw. geometrische Elemente beschrieben werden können (Punkt, Linie, Fläche). Sie ermöglicht die formale Beschreibung der topologischen Konsistenzbedingung der Überlappungsfreiheit von Flächen. Alle anderen topologischen Eigenschaften wie etwa Konnektivität, Orientierung, Anlagerung usw. werden nicht formell beschrieben.

4.1.2 Syntax

Beispiel: (stützt sich auf INTERLIS Version 2 Revision 1)

```
PROZEDUR Kontrolle_Heirat
   (IF Person.ID = Hochzeit.Person1 OR Person.ID = Hochzeit.Person2 THEN
     IstVerheiratet != TRUE
    ELSIF IstVerheiratet != FALSE) = // kontrolliert den Inhalt von
     Person.IstVerheiratet //

TABLE Person =
   ID                  [1..999999]
   IstVerheiratet      Boolean
   Name                TEXT*20
   Vorname             TEXT*20
   Alter               [0…200]
   Geschlecht          ['M', 'W']
CONSTRAINT
   UNIQUE ID;
   Alter > 18;
   Kontrolle_Heirat;
End Person;

TABLE Hochzeit =
   ID                  [1..999999]
   Ort                 TEXT*20
   Datum               DATE
   Person1             [1]→Person     // Geschlecht = 'W' //
   Person2             [1]→Person     // Geschlecht = 'M' //
CONSTRAINT
   UNIQUE ID;
End Hochzeit;
```

4.1.3 Operatoren

==, !=, <>, <=, >=, <, >, not, inside,

4.2 Object Constraint Language (OCL)

4.2.1 Allgemeine Bemerkungen

Object Constraint Language (OCL) ist eine formale Sprache, mit der Konsistenzbedingungen unmissverständlich formuliert werden können. OCL wurde erstmals von IBM in Anlehnung an die "Syntropy"-Methode eingeführt und wurde von UML übernommen, um das UML-Metamodell formal auszudrücken. Es wurde auch klar, dass mit einem Klassendiagramm in UML nicht alle Konsistenzbedingungen graphisch dargestellt werden können. Darum wird OCL auch benutzt, um das Klassendiagramm zu ergänzen. Die OCL-Ergänzung wird mit einer Notiz an eine Klasse angeheftet.

OCL ermöglicht die Beschreibung der benötigten Konsistenzbedingungen für den Objekt-orientierte Entwurf. Folglich kann sie nicht nur Konsistenzbedingungen bezüglich Daten beschreiben, sondern auch Bedingungen, die Methoden betreffen. Im Gegensatz zu anderen formalen Sprachen, wurde OCL für Systementwerfer entwickelt und nicht nur für Leute mit mathematischen Hintergrund.

OCL unterscheidet somit:

- invariante Bedingungen: sie bleiben bestehen und sind unveränderbar
- pre-Bedingungen: sie sind gültig vor der Ausführung einer Methode
- post-Bedingungen: sie sind gültig nach der Ausführung einer Methode

In diesem Dokument werden vor allem die invarianten Bedingungen beschrieben, da der Hauptbericht vorwiegend Datenmodelle behandelt.

4.2.2 Syntax

context (Typename oder Methodenname) Constrainttyp
Typename Operator Aussage

Constrainttyp = {inv, pre, post} für invariante Bedingungen, pre-Bedingungen oder post-Bedingungen.

Beispiel 1 :

```
context Entreprise inv
self.nombre_employé >50
```

Das Schlüsselwort "Self" gibt den Bezug des Attributs mit dem im "context" definierten Objekttypen.

Das Schlüsselwort "inv" bezeichnet eine invariante Konsistenzbedingung. Hiermit wird ausgesagt, dass nur Unternehmen abgebildet werden, die mehr als 50 Personen beschäftigen.

Beispiel 2: (stützt sich auf das in INTERLIS beschriebene Datenmodell)

```
context Person inv
if (self.id = hochzeit.person1 or
self.id = hochzeit.person2)
then self.IstVerheiratet = True
else self.IstVerheiratet = False
```

Beispiel 3 :

```
context Hochzeit inv
self.Person1.Geschlecht = 'W'
self.Person2.Geschlecht = 'M'
```

In einem UML-Klassenschema wird die erste Zeile (ausser bei pre- und post-Bedingungen, die sich auf Methoden beziehen) meist weggelassen, da der Kontext mit der Notiz an eine Klasse graphisch dargestellt wird.

4.2.3 Operatoren

*, +, - /, abs, <,>,<=,>=
floor
and, or, xor, not, implies, if-then-else
toUpper, concat

5 Fazit

INTERLIS bietet Sprachelemente an, die Konsistenzbedingungen beschreiben können. Leider gibt es nur wenige Beispiele, die die Anwendung dieser Sprachelemente aufzeigen. Somit ist es unklar, wie Software-Entwickler diese Sprachelemente dann auch implementieren. INTERLIS Version 2 ist in Entwicklung und wird einerseits die von der OO-Entwurfsmethode herrührenden Begriffe, wie beispielsweise Methoden, noch weiter präzisieren und andererseits die Syntax für Konsistenzbedingungen erweitern.

OCL bietet sich als Instrument zur Beschreibung der Konsistenzbedingungen für Klassendiagramme an. Es muss aber erwähnt werden, dass längst nicht alle UML-Benutzer oder Entwickler von UML-CASE-Tools OCL anwenden. Obwohl man bei OCL kaum von einem Standard sprechen kann, sollte sie berücksichtigt werden, da sie sich als Beschreibungssprache für Konsistenzbedingungen gut eignet.

K Régulation du trafic

1 Introduction

La régulation et la gestion de trafic (gestion de la mobilité) est un domaine en plein de développement provoqué par:

- La télématique routière (soit des applications d'information, soit des applications de contrôle) couple l'informatique et les technologies de la communication pour échanger des information en temps réel.
- L'apparence des systèmes de navigation ayant une certain influence sur le comportement des usagers routiers

En Europe, les professionnels du trafic ont compris qu'une forte standardisation est nécessaire afin d'assurer l'interopérabilité des systèmes et des services audelà des frontières des pays.

Au niveau de la télématique routière, les standards suivants ont été introduits:

- RDS-TMC (Radio Data System – Traffic Message Channel): standard de transmission de message de trafic par RDS (Radio Data System)
- ALERT-C: protocole de transmission des messages de trafic par RDS-TMC

Au niveau des systèmes de navigation, le standard suivant a été introduit:

- GDF (Geographic Data Files): cette norme définit un dictionnaire de données standardisées et un format d'échange. La CEN TC 278 (Centre Européen de Normalisation) a publié une première norme en 1995, la version 3. Entretemps, l'ISO TC 204 (International Standardisation Organisation) a repris les éléments du CEN-GDF et elle a établi une première version de l'ISO-GDF

En Suisse, un leitbild a été élaboré afin de donner des directives pour la politique de trafic en Suisse concernant l'intégration des nouvelles technologies de la télématique routière.

2 RDS-TMC

Le service RDS-TMC (Radio Data System - Traffic Message Channel) envoie des informations routières vers des récepteurs radio FM en utilisant une transmission numérique des événements, de leur statut et de leur référence spatiale.

Ce service permet de délivrer rapidement dans la langue choisie par l'utilisateur, les informations essentielles avec une très bonne précision. De plus il n'est pas nécessaire d'interrompre la diffusion des émissions radio.

Les données et les événements sont collectés par une centrale d'informations routières. Ces informations lui sont fournies soit directement depuis des installations sur le réseau routier, soit par les utilisateurs, soit par les autorités ou d'autres sources. Les informations sont validées puis transmises au fournisseur du service responsable de la diffusion des messages RDS-TMC.

Les informations sont codées à l'aide du protocole standard pour TMC. Chaque information est structurée en utilisant une liste standard des événements et une liste standard des localisations pour identifier le lieu où l'événement s'est produit.

Le fournisseur du service TMC transmet le message au réseau FM approprié afin qu'il puisse être diffusé comme "sub-carrier" au sein du programme radio.

Une fois le message diffusé, les récepteurs sont capables de les accepter à l'aide d'un tuner standard. Celui-ci transmet le message à l'unité de décodage.

Grâce au codage du message, l'utilisateur final peut choisir la langue dans laquelle le message doit être présenté. Il peut également définir des filtres géographiques qui doivent être appliqués au message reçu en fonction de l'itinéraire choisit.

Le codage permet également de mieux gérer les messages au niveau de la diffusion et au niveau du récepteur en identifiant l'étendue et la durée d'un événement.

2.1 Fonctionnalités du service RDS-TMC

Aujourd'hui, les informations routières émises interrompent à chaque fois le programme radio. A l'avenir ceci ne sera plus être nécessaire avec le système RDS-TMC (Radio Data System - Traffic Message Channel).

Les auditeurs FM connaissent le RDS en tant que système qui affiche sur un autoradio compatible RDS le nom du programme et qui commute automatiquement vers un meilleur émetteur du même programme en cas de baisse de la qualité de réception.

Le RDS émet parallèlement au programme radio, par l'émetteur FM, des informations textuelles codées numériquement, p.ex. le nom du programme.

Dans le cadre de RDS, un canal d'informations routières (TMC: Traffic Message Channel), par lequel il est possible de transmettre simultanément jusqu'à 20 informations routières muettes, codées numériquement est mis à disposition.

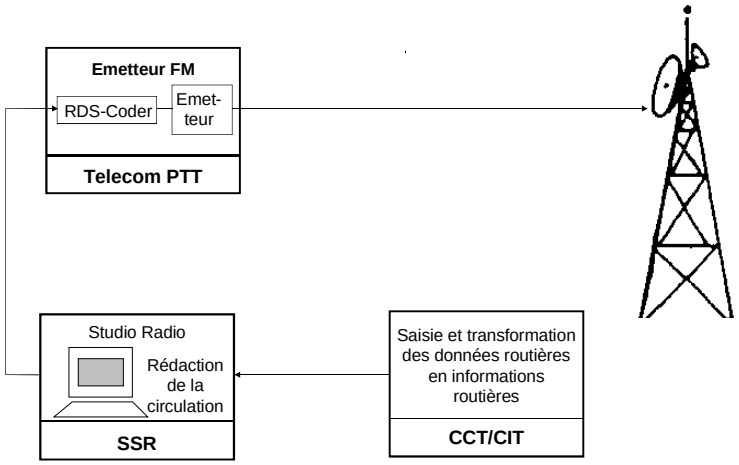


Figure 8-14: Les intervenants dans la transmission de messages de trafic par RDS.

Une information TMC standard comporte essentiellement quatre parties:

- Identification du pays, identification de l'émetteur (Program Identification-Code)
- Informations diverses comme p.ex. identification message TMC (Group-Type-Block)
- Informations sur l'événement: Description de l'événement et du sens de circulation concerné, indications concernant les déviations, étendue de l'événement (Message Code)
- Repérage dans l'espace: Description du lieu, du tronçon de route ou de l'espace en rapport avec l'événement qui fait l'objet du message (Location Code)

A titre d'exemple, à l'aide de données sur la localisation et l'événement, le message TMC envisageable ci-dessous peut être explicité:

Message Code:	224:	Voie de droite fermée
Location Codes:	99997:	Restoroute Grauholz
	99998:	Embranchement Schönbühl
	99999:	Jonction Kirchberg

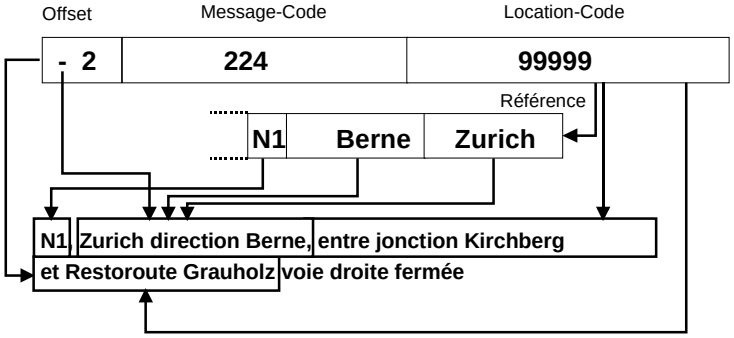


Figure 8-15: Exemple d'un message TMC

Pour pouvoir capter des informations TMC à bord d'une voiture, il faut disposer d'un poste autoradio RDS muni d'un décodeur TMC. Celui-ci consiste essentiellement d'un chip-mémoire comprenant des éléments de langage numérisés et un synthétiseur de voix.

A l'aide du code TMC reçu par RDS, l'information routière sera composée et émise en langage synthétique. Etant donné que le décodage des informations routières s'effectue seulement au niveau de l'autoradio, le conducteur peut choisir la langue d'émission et écouter ainsi les informations dans sa propre langue, même s'il se trouve dans une autre région linguistique. L'automobiliste aura la possibilité de sélectionner la réception des seules informations le concernant (p.ex. sélection pour région, route, etc.).

2.2 Participation de la Suisse dans ce processus

2.2.1 SNIR (NAVIS)

Dans le cadre du projet SNIR (Système National d'Information Routière), coordonné par la Commission de Travail d'Informatique de Trafic (CTIT), les informations codées sur le trafic (RDS-TMC) constituent un des éléments permettant d'améliorer la situation de l'information routière en Suisse. La diffusion d'informations routières via RDS-TMC est une des fonctions prévue dans SNIR.

A la suite de l'analyse préliminaire la CTIT a mis en oeuvre en 1996 un projet pilote RDS-TMC dans la région de Berne. La gestion de ce projet a été confié à l'OFROU avec comme participants la Police cantonale de Berne, SWISSCOM, la SSR, l'ACS, le TCS ainsi que B+S Ingenieure AG et Rosenthaler + Partner AG. Le projet pilote dénommé "Feldversuch RDS-TMC" a pour but d'étudier la faisabilité et les bénéfices de l'introduction de RDS-TMC depuis le levé de l'information par la Police cantonale de Berne en passant par le traitement de l'information à la Centrale d'Information du Trafic jusqu'à sa diffusion par la SSR. En 1996 – 1997 un concept regroupant les besoins en fonctionnalités, en données et du point de vue organisationnel a été élaboré. Actuellement les différentes variantes techniques pour la réception des messages sont analysées.

2.2.2 CORVETTE I et II

La Suisse participe également au projet européen CORVETTE. Ce projet a pour but de coordonner l'implémentation de systèmes intélligents de trafic (angl. ITS) dans les régions alpines. Les objectifs principaux de CORVETTE se résument en:

Acquisition de données de trafic: définition des types de données, définitions de formats communs de données, harmonisation des systèmes d'acquisition de données

Gestion de données de transport: étude des relations public / privé pour la collecte et la distribution de données de trafic, principes et règles pour l'accès aux données, initialisation de l'échange de données au-delà des frontières

Services RDS-TMC: échange de données au-delà des frontières, définition de bases légales pour les services RDS-TMC, définition des bases de repérages spatiales (location database) et des listes d'événements nationales et régionales, harmoniser les méthodes de payements, aider au développement de récepteurs RDS-TMC en différentes langues.

Collecte automatique de péages (EFC): Surveiller les besoins et les plans de mises en oeuvre pour élaborer une stratégie commune en Europe au vue de la surveillance du trafic.

Les partenaires de CORVETTE sont l'Autriche, l'Allemagne, l'Italie et la Suisse. Les réseaux couverts sont le TEN-T dans la région alpine. Il s'agit de l'Autriche, de la bavière, du nord est de l'Italie (incluant le Vénéto, le Friouleet la région de Trentine et du Haut-Adage), de la Suisse ainsi que d'importants tronçons co-urbains dans ces régions (tels que Munich, Vérone et Vienne).

La première étape (automne 1996 à septembre 1997) CORVETTE I a permis d'élaborer une étude de faisabilité. Les principaux résultats ont été:

- le choix des Centrales d'Informations de Trafic (CIT) pour l'échange de données international dans les régions concernées
- la vérification des tests pour l'interopérabilité de RDS-TMC
- la définition des niveaux de qualité requis pour l'information
- la planification de l'infrastructure nécessaire à la mise en oeuvre
- un approche mulitlatéral pour la gestion de trafic dans les régions concernées

La deuxième étape CORVETTE II traite l'élaboration des concepts nécessaires à la mise en oeuvre des différents objectifs fixés dans l'étape I. Le projet CORVETTE II a été découpé en projets Euro-Régionaux traitant les objectifs communs à tous les participants et en projets Régionaux permettant de concrétiser des besoins spécifiques à chaque pays.

Les sous-projets Euro-Régionaux sont:

- ER1: Implémentation et vérification des services RDS-TMC dans la région alpine.
- ER2: Etude et implémentation de l'échange de données international dans la région alpine.
- ER3: Gestion de trafic interrégional (Interregional re-routing)
- ER4: Etude d'impact dans les régions couvertes par CORVETTE de la "EU Council resolution 95/C 264/01" sur l'EFC

Les sous-projet régionaux suisses sont:

- CH1: Projet pilote RDS-TMC
- CH2: Implémentation d'une base de données à la Centrale d'Information de Trafic de Genève
- CH3: Intégration de systèmes d'acquistions automatiques de données de trafic

– CH4: Etude sur les services d'information de trafic du futur.

3 ALERT-C

3.1 Généralités

Contrairement aux bases de données qui gèrent des données statiques, certains services ou applications en télématique des transports doivent transmettre ou échanger des données dynamiques comme par exemple les services d'information routière, la gestion de flottes de véhicules, etc.

Une localisation est un lieu identifiable dans le monde réel auquel on associe généralement un objet du monde réel. Une localisation peut représenter un lieu précis du réseau routier ou une région par exemple. Une méthode de localisation, "location referencing" en anglais, est l'identification d'une localisation par un système qui peut la transmettre à un autre système sans ambiguïtés. Une telle méthode est utilisée pour transférer une information liée à une localisation d'un système à un autre.

Les deux méthodes présentées dans les sections suivantes sont très différentes, la première n'étant pas géo-réferencée nécessite des tables de localisation afin de pouvoir décoder l'information de localisation associée à l'information de trafic correspondante.

L'idée d'améliorer l'information de trafic a pris naissance suite aux lacunes de la diffusion des messages de trafic par les stations de radio, en particulier :

Les messages ne sont pas adaptés au conducteur. Un conducteur aimerait seulement recevoir les messages qui lui sont utiles. Un filtrage de ces messages en fonction de la position du véhicule est souhaité.

Les stations FM couvrent des zones d'émission limitée. Les messages de trafic doivent souvent être répétés pour informer les conducteurs qui entrent dans cette zone. Cette répétition devient fatigante pour les conducteurs restant dans cette zone.

Lors de longs trajets le conducteur doit chercher fréquemment des nouvelles stations FM pour recevoir l'émission correctement.

Lors de trajets à l'étranger, le conducteur doit maîtriser plusieurs langues pour comprendre les messages de trafic.

Le système RDS-TMC (Radio Data System - Traffic Message Channel) développé dans les années huitante propose une solution qui est actuellement en phase d'implémentation dans toute l'Europe. Le principe de ce service est de transmettre en continu et indépendamment de la langue des informations de trafic codées. Le récepteur décode, filtre et présente ces informations dans la langue du conducteur.

Les éléments nécessaires pour le fonctionnement du RDS-TMC ont été standardisés par le CEN TC278. Le standard ALERT-C (Advice and Problem Location for European Road Transport, version C, CEN ENV12313-1) est le protocole de codage pour RDS-TMC auquel sont associés les standards spécifiants:

- la liste d'événements (event and information codes for TMC - The "Event list", CEN ENV12313-2)
- la méthode de localisation (Location referencing rules for RDS-TMC, CEN ENV12313-3).

ALERT-C a été étendu afin d'associer des informations supplémentaires comme par exemple la durée de parcours. Contrairement à ALERT-C – qui est orienté événement – ce nouveau protocole est orienté statuts et a été appelé ALERT-Plus.

3.2 Les données

La liste d'événements d'ALERT-C comprend les catégories suivantes:

- Niveau de service
- Incidents/accidents
- Fermetures
- Restriction des voies
- Travaux
- Obstructions
- Etat de la route
- Temps
- Vents
- Environnement
- Temperature
- Activités
- Délais/ Annulations
- Véhicules dangereux
- Charges exceptionnelles
- Etat des équipements de trafic
- Régulation de trafic
- Parking
- Information

4 GDF

GDF (Geographic Data Files) est le standard utilisé pour la télématique routière. GDF a été développé en Europe pendant plus de dix années et a été accepté en 1995 comme pré-standard (ENV) par le CEN TC278 (Comité Européen de Normalisation, Technical Committee 278). Ce CEN-GDF est un standard permettant la définition et l'échange de données géographiques. Il inclut en particulier:

- un modèle de donnée indépendant des applications (modèle conceptuel orienté objet)
- un dictionnaire de donnée dépendant des applications
- la définition de concepts de qualité
- les spécifications du format d'échange

La standardisation de GDF continue actuellement au niveau mondial au sein de l'ISO TC204 (International Standard Organization - Technical Committee 204). Cet ISO-GDF qui devrait être finaliser en avril 1999 est essentiellement basé sur le document CEN-GDF. Le GDF considéré dans ce document est le CEN-GDF. Les éléments géographiques ont été décrit dans l'annexe D "Le modèle géométrique de GDF".

Des informations complémentaires sur GDF peuvent aussi être trouvées sur la homepage de GDF accessible sur le site d'ERTICO (http://www.ertico.com/gdf/index.htm) qui propose aussi un GDF helpdesk à l'adresse suivante: gdf.helpdesk@mail.ertico.com.

Le dictionnaire de données de GDF contient et couvre les thèmes suivants :

- Roads and Ferries (réseau des routes et des ferrys)
- Administratives Areas (les zones administratives)
- Settlements and Named Areas (les colonies et les lieux dits)
- Land Cover and Use (les zones d’affectation)
- Brunnels (BRidge and tuNNEL : les ponts et les tunnels)
- Railways (les chemins de fer)
- Waterways (le réseau hydrographique)
- Road Furniture (la signalisation)
- Services
- Public Transport (le transport public)
- General Features (des objets généraux comme le centre de l’objet)

GDF n'est pas directement utilisé dans des applications télématiques mais par les créateurs de carte géographique numérique qui développent ces cartes pour: la navigation automobile, la gestion de flottes de véhicules, les centres de gestion du trafic routier, etc.

L Auszug aus der Norm Geo 405

M Beispiel einer INTERLIS-Beschreibung von Strassendaten

Bundesamt für Strassenbau

STRADA-INTERFACE

Fichier de description

DRAFT

Autor:

Rainer Oggier

Version 2.06
02.04.2002

Table of Contents (D: Inhaltsverzeichnis, F: tables des matières)

```
TRANSFER STRADA_Data_Catalog;   !! Defines Transfer-Set
!! Name Conventions in STRADA-DB:

!! Short-name Relation-name         STRADA-Table-name  STRADA-Table-synonym
!! ========================================================================
!! ACO        ATTRIBUT_COMBINATION STRTB_ATTR_COMBIS  ATTR_COMBIS
!! ACP        ACCESS_PATH          STRTB_ACCESS_PATHS ACCESS_PATHS
!! ADR        ADDRESS              STRTB_ADDRESSES    ADDRESSES
!! APT        ACCESS_PATH_TITLE    STRTB_ACP_TITLES   ACP_TITLES
!! APU        ACCESS_PATH_USAGE    STRTB_ACP_USAGES   ACP_USAGES
!! AXE        AXIS                 STRTB_AXES         AXES
!! CAT        CATALOG              STRTB_CATALOGS     CATALOGS
!! CCT        CATALOG_COLUMN_TITLE STRTB_COL_TITLES   COL_TITLES
!! CHA        CHARACTERISTIC       STRTB_SBJ_CHARS    SBJ_CHARS
!! CMR        COMPOSITION_RULE     STRTB_COMPRULES    COMP_RULES
!! COD        CODE                 STRTB_CODES        CODES
!! COL        CATALOG_COLUMN       STRTB_CAT_COLUMNS  CAT_COLUMNS
!! CRS        CROSS_SECTION        STRTB_CROSS_SECTS  CROSS_SECTS
!! CSU        CROSS_SECT_USAGE     STRTB_CR_S_USAGES  CR_S_USAGES
!! CTI        CATALOG_TITLE        STRTB_CAT_TITLES   CAT_TITLES
!! CTK        CATALOG_TEXT_KEY     STRTB_CAT_TEXTKEYS CAT_TEXTKEYS
!! CTP        CODETYPE             STRTB_CODETYPES    CODETYPES
!! CTT        CATALOG_TEXT         STRTB_CAT_TEXTS    CAT_TEXTS
!! DAO        DATAOWNER            STRTB_DATAOWNERS   DATAOWNERS
!! DOC        DOCUMENT             STRTB_DOCUMENTS    DOCUMENTS
!! DOU        DATAOWNER_USAGE      STRTB_DAO_USAGES   DAO_USAGES
!! DUS        DOC_USAGE            STRTB_DOC_USAGES   DOC_USAGES
!! ETK        ELEMENTARY_TEXT_KEY  STRTB_ELEM_TXTKEYS ELEM_TXTKEYS
!! ETT        ELEMENTARY_TEXT      STRTB_ELEM_TEXTS   ELEM_TEXTS
!! JNC        JUNCTION             STRTB_JUNCTIONS    JUNCTIONS
!! LAL        LATERAL_LANE         STRTB_LAT_LANES    LAT_LANES
!! LKG        LINK_GROUP           STRTB_LINK_GROUPS  LINK_GROUPS
!! LNK        LINK                 STRTB_LINKS        LINKS
!! LSC        LOG_SECTOR_CHANGE    SYSTB_LOG_SEC_CHG  ---
!! NLO        NODE_LOCATION        STRTB_NODE_LOCS    NODE_LOCS
!! NOD        NODE                 STRTB_NODES        NODES
!! ONN        OWNER_NAME           STRTB_OWNER_NAMES  OWNER_NAMES
!! OWN        OWNER                STRTB_OWNERS       OWNERS
!! OWS        OWNER_SYNONYM        STRTB_OWNER_SYNS   OWNER_SYNS
!! PER        PERSON               STRTB_PERSONS      PERSONS
!! PLA        PLACE                STRTB_PLACES       PLACES
!! PLN        PERSON_LEG_NAME      STRTB_PER_LEG_NAMS PER_LEG_NAMS
!! PNA        PLACE_NAME           STRTB_PLACE_NAMES  PLACE_NAMES
!! PRO        PROJECT              STRTB_PROJECTS     PROJECTS
!! PVL        PAVEMENT_LAYER       STRTB_PAVE_LAYERS  PAVE_LAYERS
!! RCL        ROADSURFACE_CLASS    STRTB_RS_CLASSES   RS_CLASSES
!! RCO        ROADSURFACE_CONTROL  STRTB_RS_CONTROLS  RS_CONTROLS
!! RPT        REFERENCE_POINT      STRTB_REFERENCE_PS REFERENCE_PS
!! RRP        ROAD_REPAIR          STRTB_ROAD_REPAIRS ROAD_REPAIRS
!! SBJ        SUBJECT              STRTB_SUBJECTS     SUBJECTS
!! SFU        SUBJECT_FUNCTION     STRTB_SBJ_FUNCS    SBJ_FUNCS
!! SRP        SEC_REFERENCE_POINT  STRTB_SR_POINTS    SR_POINTS
!! SRU        SEARCH_RULE          STRTB_SEARCH_RULES SEARCH_RULES
!! SUS        SUBJ_FUNCT_USAGE     STRTB_SFU_USAGES   SFU_USAGES
!! TCO        TEXT_COMBINATION     STRTB_TEXT_COMBIS  TEXT_COMBIS
!! USR        USER                 SYSTB_USERS        USERS
!! XDB        TRANSFER_DATABASES   SYSTB_TR_DBS       ---
!! XST        TRANSFER_SETS        SYSTB_TR_SETS      ---
!! Module Traffic:
!! TIS        TIME_SERIES                    TIME_SERIES
!! VAF        VARIATION_FUNCTIONS            VARIATION_FUNCTIONS
!! TVR        TRAF_VALUES_RELS               TRAF_VALUES_RELS
!! TVA        TRAF_VALUES_ABS                TRAF_VALUES_ABS
!! TVC        TRAFFIC-VALUE_CLASSES          TRAFFIC-VALUE_CLASSES
!! TCL        TRAFFIC_VALUE_CLASSIFICATIONS TRAFFIC_VALUE_CLASSIFICATIONS

!! Units to use
!! Il ist absolutly necessary to use units of SI
!! [Lenghts]:= Meter
!! [Time]:= Second
!! [Weight]:= Kilogram
```

```
MODEL STRADA_2_06                  !! Defines the model for STRADA-DB


    DOMAIN            !! Acceptable domains within STRADA-DB
    LKoord3D     = COORD3 400000.000 150000.000 0.000 900000.000 350000.000 5000.000;
    SectorLength = DIM1 0.000 99999.999;
    Meter        = DIM1 -999999999.999 999999999.999;
    BASEID       = TEXT*20;
    VERSION      = DIM1 0 9999;
    OWNER        = TEXT*4;
    CK1          = TEXT*2;
    CK2          = TEXT*4;
    CK3          = TEXT*7;
    CK4          = TEXT*8;
    CK5          = TEXT*32;
    COMMENT      = TEXT*2000;
    DATEDOM      = TEXT*20;             !! Format: DD.MM.YYYY hh:mm:ss
                                        !! D: Day, M: Month, Y: Year
                                        !! h: Hour, m: minute, s: second
    ADTEXT       = TEXT*72;
    NAME         = TEXT*32;
    NAME2        = TEXT*72;
    TEXTID       = TEXT*10;
    DBID         = TEXT*4;
    DATAOWNER    = TEXT*4;
    ORAUSER      = TEXT*32;
    PNUM2        = DIM1 0 99;
    PNUM3        = DIM1 0 999;
    PNUM4        = DIM1 0 9999;
    PNUM5        = DIM1 0 99999;
    PNUM6        = DIM1 0 999999;
    PNUM8        = DIM1 0 99999999;
    PNUM9        = DIM1 0 999999999;
    PNUM1_1      = DIM1 0.0 9.9;
    PNUM4_2      = DIM1 0.00 9999.99;
    PNUM5_1      = DIM1 0.0 99999.9;
    PNUM5_2      = DIM1 0.00 999999.99;
    NUM1_3       = DIM1 -9.999 9.999;
    NUM1_8       = DIM1 –9.99999999 9.99999999;
    NUM1_10      = DIM1 -9.9999999999 9.9999999999;
    NUM2_2       = DIM1 -99.99 99.99;
    NUM2_3       = DIM1 -99.999 99.999;
    NUM3_2       = DIM1 -999.99 999.99;
    NUM4_3       = DIM1 -9999.999 9999.999;
    NUM4_4       = DIM1 -9999.9999 9999.9999;
    NUM6_3       = DIM1 -999999.999 999999.999;
    NUM7_4       = DIM1 -9999999.9999 9999999.9999;
    NUM9_2       = DIM1 -999999999.99 999999999.99;
    NUM11_4      = DIM1 -9999999.9999 9999999.9999;
    NUM20_10     = DIM1 -9999999999.9999999999 9999999999.9999999999;
    NUM38        = DIM1 -99999999999999999999999999999999999999
                         99999999999999999999999999999999999999;
    U_ABS        = DIM1 0.00 999999.99
    MODEL        = (Maxi,Kernel);
    OWCCODE      = (B,G,K,N,P);
    ITSCODE      = (0,1,2);
    LANDCODE     = (D,E,F,I);          !! (D:German,E:English,F:French,I:Italian)
    POSCODE      = (,+,-);
    SUBJECTCODE  = ();
    COSTCODE     = (1,2,3,4,5,6,7,8,9); !! (1: commun costs; 2: land acquisition;
                                        !! 3: excavation work; 4: bridges;
                                        !! 5: tunnel (the ones of the opencast working
                                        !! included); 6: galeries; 7: other structures;
                                        !! 8: facturies, restaurants and auxiliary
                                        !! buildings; 9: other costs);
    TDICODE      = (I,O);
    TSTCODE      = (1,2,3,4,5);
    VERSCODE     = (E,P);              !! (E:Effective,P:Predicted)
    ACCCODE      = (YU,NU,NV)          !! (YU:yes user, NU: no user, NV: not valid)
    COMCODE      = (YS,NS,NV)          !! (YS:yes system, NS:no system, NV:not valid)
    COHCODE      = (YS,NS,YU,NU,NV)    !! (YS:yes system, NS:no system, YU:yes user,
                                       !!  NU:no user, NV:not valid)
    PLACODE      = (YS,NS,YU,NU,NV)    !! (YS:yes system, NS:no system, YU:yes user,
                                       !!  NU:no user, NV:not valid)
    CHACODE      = (OR,CS,CU,ON)       !! (OR:value not yet changed, CS: changed by
                                       !! system, CU: changed by user, based on system,
                                       !! ON: changed by user on-line)
    ASPCODE      = (0,1,2,3,4,5,6,7,8) !! (0:sum, 1:average, 2:minimum, 3:maximum,
                                       !!  4:percentil,5: 95-percentil, 6:85-percentil,
                                       !!  7:15-percentil, 8:series)
```

```
KINCODE      = (GL,SK,ZR)           !! (GL:variation curve episode, SK:scalar episode,
                                    !!  ZR: time series episode)
STCODE       = (0,1,2);             !! (0:suggested; 1:imposed; 2:manual)
DECCODE     = (0,1);                !!          (0:partial; 1:total)
DECSTCODE    = (0,1,2);             !!          (0:in preparation; 1:available; 2:used)
DECSTCOH     = (0,1);               !!          (0:not consistent; 1:consistent)
CHCCLCD      = (0,1,2,3,4)          !! (0:0m-500m; 1:501m-1000m; 2:1001m-1500m;
                                    !!  3:1501m-2000m; 4:2001m-2500m)
SITUCODE     = (0,1)                !! (0:outside location; 1:inside location)
TYPCHCODE    = (0,1,2,3,4,5)        !!          (0:carriageway; 1:bridge, tunnel, galery;
                                    !!  2:tunnel 3:galery; 4:bridge;
                                    !! 5:tunnel and galery);
POSCHCODE    = (0,1,2,3)            !!          (0:at level; 1:embankment; 2:excavation;
                                   !!           3:lateral shoulder)
TYSTRCODE    = (0,1,2)              !!          (0:R; 1:S; 2:MS)
CLTRFCODE    = (0,1,2,3,4,5)        !!          (0:T1; 1:T2; 2:T3; 3:T4; 4:T5; 5:T6)
OUINON       = (0,1)                !!          (0:oui; 1:non)
DERIVCODE    = (0,1,2,3)            !!          (0:without; 1:bad; 2:average; 3:good)
METHCODE     = (0,1,2)              !!          (0:ViaPms; 1:Belman; 2:Exploratory)
STUTCODE    = (0,1,2,3,4)           !!          (0:in preparation; 1: ready for analyse;
                                    !!  2:received (measure);
                                    !!  3:received (measure, object);
                                    !! 4:received (measures, all objects))
CONDCODE     = (0,1)                !! (0:predetermined values; 1:selected values
                                    !!   in the PMS-tool)
CRITCODE     = (0,1,2)              !! (0:fixed budget; 1:fixed level serv.; 2:free)
PARAMCT      = (1,2,3)              !! (1:value1; 2:value2; 3:value3)
TYMESCODE    = (0,1,2,3)            !! (0:rehabilitation; 1:maintenance; 2:periodic;
                                  !!  3:auxiliary)
BUDGCODE     = (0,1)                !! (0:upkeep; 1:maintenance)
UNCODE       = (0,1)                !! (0:square metre; 1:metre; 2:ton; 3:piece)
   = (0,1,2,3)                      !! (0:absolut; 1:relative; 2:%; 3:expression)
STMODCODE    = (0,1,2)              !! (0:proposed; 1:imposed; 2:manual)
USAGE_CODE   = (G,N,X,A,C,V,P);     !! (G:GIS N:Numeric X:Others A:Ax Strip C:Carto
                                   !!  V:StradaView P: Projection on the road)
TOOLCODE = (D,P,X,O);               !! (D:Digitizer P:Road Projection X:Others
                                   !!  O:Digitized on screen)
ELEM_CODE = (0,1,2,3);        !! (0:Straight 1:Cercle 2:Clothoïd 3:Cubic Function)
POSITCODE = (0,1,2);                !! (0:Start 1:Intermediate 2:End)
NATCODE  = (BA,BG,HP,BR,HO,VK,P,X); !! (BA:Start/End of ax segment,
                                    !!          BG:Start/End of geometric segment
                                    !!          HP:Measured Node BR:Bridge HO:Limit
                                    !!          VK:Trafic Node BP:Reference Point X:Other)
DATATYPES = (PRO, CTP, CAT, D, N, T); !! (PRO: Project, CTP; Code Type, CAT: Text Catalog, D: Date,
                                      !!  N: Numeric, T: Text)
```

```
    TABLE Organisation =
    !! ( D:‘Metadaten der Transferdatei-Inhalte‘, F:‘Métadonnées du contenu du fichier
    !! d‘échange‘)
      DefinitionVersion:              TEXT*6      //Kernel//;
      TransferName:                   TEXT*40     //Kernel//;
      ModelDef                        MODEL       //Kernel//;
      SourceSystemName:               TEXT*40     //Kernel//;
      TargetSystemName:               TEXT*40     //Kernel//;
      TimeStampExport:                DATEDOM     //Kernel//;
      TimeStampData:                  DATEDOM     //Kernel//;
      ExportUsername:                 TEXT*40     //Kernel//;
END Organisation;

!!==============================================================================================
!! DEF: Data about the whole data set which is about to exchange
!!==============================================================================================
!! Attribut Desription
!! DefinitionVersion    Version of INTERLIS-Data-Description for STRADA, i.g. 0.10
!! TransferName         Name of Transferset, i.g.
!! SourceSystemName     Name of emitting system, i.g. ASB.RP..
!! TargetSystemName     Name of receiving system, i.g. ASB.ASB.
!! TimeStampExport      Creation date of transfer files, i.g. 18.09.1997 10:39:42
!! TimeStampData        Reference date of data in transfer files, i.g. 01.01.1996 00:00:00
!! ExportUsername       Name of responsable user of the transfer, i.g. SDBM
!! ============================================================================================
```

```
    TABLE TK_Valid =
!! ( D:‘Extraktionsdatum der Textkataloge‘, F:‘Date d'extraction des catalogues de texte')
      AccTyp:                OPTIONAL  DATEDOM;
      AxTyp:                 OPTIONAL  DATEDOM;     !! AXT
      ContTyp:               OPTIONAL  DATEDOM;     !! CLT
      DocTyp:                OPTIONAL  DATEDOM;     !! DOT
      FuncTyp:               OPTIONAL  DATEDOM;
      GeomTyp:               OPTIONAL  DATEDOM;
      HeadTyp:               OPTIONAL  DATEDOM;
      HindrTyp:              OPTIONAL  DATEDOM;
      JuncTyp:               OPTIONAL  DATEDOM;     !! JCT
      LaterTyp:              OPTIONAL  DATEDOM;     !! UST
      LayTyp:                OPTIONAL  DATEDOM;     !! PVT
      MeasTyp:               OPTIONAL  DATEDOM;     !! PRT
      NodeTyp:               OPTIONAL  DATEDOM;     !! NOT
      PlateTyp:              OPTIONAL  DATEDOM;     !! PAT
      ProjTyp:               OPTIONAL  DATEDOM;     !! PPT
      RdwTyp:                OPTIONAL  DATEDOM;     !!
      RelTyp:                OPTIONAL  DATEDOM;
      RPTyp:                 OPTIONAL  DATEDOM;     !! RET
      RRepTyp:               OPTIONAL  DATEDOM;     !! RRT
      StrulTyp:              OPTIONAL  DATEDOM;     !! SLT
      TrafvalClassTyp:       OPTIONAL  DATEDOM;     !! VCT
    END TK_Valid;

!! ================================================================================
!! DEF: Up-to-Dateness of text catalog content
!! This table defines the state of every text catalog by means of a date. The date represents
!! the point of time of the latest registration (elementary text, complex text) in a text
!! catalog
!! ================================================================================
!! ATTRIBUT DESCRIPTION:
!! AccTyp               Accident Type, i.g. 13.07.1997 14:09:53
!! AxTyp                Ax type, i.g. 13.07.1997 14:09:53
!! ContTyp              Control type, i.g. 13.07.1997 14:09:53
!! DocTyp               Document type, i.g. 13.07.1997 14:09:53
!! FuncTyp              Function type, i.g. 13.07.1997 14:09:53
!! GeomTyp              Geometric source type, i.g. 13.07.1997 14:09:53
!! HeadTyp              Salutation type, i.g. 13.07.1997 14:09:53
!! HindrTyp             Hindrance type, i.g. 13.07.1997 14:09:53
!! JuncTyp              Junction type, i.g. 13.07.1997 14:09:53
!! LaterTyp             Usage type, i.g. 13.07.1997 14:09:53
!! LayTyp               Layer type, i.g. 13.07.1997 14:09:53
!! MeasTyp              Project type, i.g. 13.07.1997 14:09:53
!! NodeTyp              Node type, i.g. 13.07.1997 14:09:53
!! PlateTyp             Shield type, i.g. 13.07.1997 14:09:53
!! ProjTyp              Projection point type, i.g. 13.07.1997 14:09:53
!! RdwTyp               Roadwork type, i.g. 13.07.1997 14:09:53
!! RelTyp               Relation type, i.g. 13.07.1997 14:09:53
!! RPTyp                Physical Reference Point type, i.g. 13.07.1997 14:09:53
!! RRepTyp              Reparation Type, i.g. 13.07.1997 14:09:53
!! StrulTyp             Link building rule type, i.g. 13.07.1997 14:09:53
!! ================================================================================
!! Rules: Only those attributs are mandatory to whose text catalog objects of the transfer
!! set have been referenced. The other attributs may not be used.
!! ================================================================================
```

```
TABLE Owner =
!! ( D:‘Eigentümer‘, F:‘Propriétaire‘)
   OWN_Owner:          OWNER                    //Kernel//;   !!               PK
   OWN_Nation:         TEXT*3;
   OWN_Canton:         OPTIONAL     TEXT*2;
   OWN_District:       OPTIONAL     TEXT*4;
   OWN_Municipality:   OPTIONAL     TEXT*4;
   OWN_Owc_Code:                    OWCCODE;
   CONSTRAINT
   UNIQUE
   OWN_Owner;
   END Owner;

!!  ATTRIBUT DESCRIPTION:
!!  OWN_Owner                                 Owner Key, i.g. ASB, AG, BL, FR, VD
!!  OWN_Nation                                Nation
!!  OWN_Canton                                Canton
!!  OWN_District                              District
!!  OWN_Municipality                          Municipality
!!  OWN_Owc_Code
!! ================================================================================
```

```
    TABLE Transfer_Sets =
    !! ( D:‘Transfersets‘ F:‘Jeux du transfert‘);
     XST_XfrSetID:                      BASEID;                                  !! PK
     XST_TDI_Code:                      TDICODE     //Kernel//;
     XST_TST_Code:                      TSTCODE     //Kernel//;
     XST_Description:                   TEXT*80     //Kernel//;
     XST_CreateDate:                    DATEDOM     //Kernel//;
     XST_MarkDate:         OPTIONAL     DATEDOM     //Kernel//;
     XST_CopyDate:         OPTIONAL     DATEDOM     //Kernel//;
     XST_ReceiveDate:      OPTIONAL     DATEDOM     //Kernel//;
     XST_IntegrateDate:    OPTIONAL     DATEDOM;    //Kernel//;
       CONSTRAINT
       UNIQUE
       XST_XfrSetID;
    END Transfer_Sets;

!! ===============================================================================
!! DEF: Defines a transfer set in the Import/Export Area of STRADA-DB
!! ===============================================================================
!! ATTRIBUTSBESCHREIBUNG:
!! XST_XfrSetID
!! XST_TDI_Code
!! XST_TST_Code
!! XST_Description
!! XST_CreateDate
!! XST_MarkDate
!! XST_CopyDate
!! XST_ReceiveDate
!! XST_IntegrateDate

 END Transfer_Metadata
!! ===============================================================================
```

```
    TABLE SimpleCatalog =
!! ( D:‘komplexe Textkatalogeinträge (denormalisierte, vereinfachte Struktur‘
!!   F:‘Enregistrements de textes complexes dans les catalogues de texte (structure simplifiée !!
et dénormalisée)‘, GEN-SN640940)
      SCA_CTK_BaseID:                  BASEID;                  !! PK             GEN-SN-I1
      SCA_CTK_Version:                 VERSION;                 !! PK             GEN-SN-I2
      SCA_CTT_LNG_Code:                LANGCODE;                !! PK/UKC         GEN-SN-I3
      SCA_CAT_BaseID:                  BASEID;
      SCA_CAT_Version:                 VERSION;
      SCA_CAT_CKO_Owner:               OWNER       //Kernel//;  !! UKC(.)->OWN
      SCA_CAT_CK:                      CK4         //Kernel//;  !! UKC
      SCA_CAT_StandardSeq: NUM5        //Kernel//;
      SCA_CTI_LNG_Code:                LANGCODE    //Kernel//;
      SCA_CTI_Abbreviation:            TEXT*10     //Kernel//;
      SCA_CTI_Title:                   TEXT*32     //Kernel//;
      SCA_CTK_StandardSeq: NUM5        //Kernel//;
      SCA_CTK_HierarchyCode: OPTIONAL TEXT*10      //Kernel//;
      SCA_CTK_CKO_Owner:               OWNER       //Kernel//;  !! UKC(.)->OWN
      SCA_CTT_TextID:                  TEXTID      //Kernel//;  !! UKC
      SCA_CTT_Text:                    NAME        //Kernel//;
      SCA_CTT_Abbreviation:            TEXT*10     //Kernel//;
      SCA_CTT_Descript: OPTIONAL       TEXT*255    //Kernel//;
      SCA_CTK_VRS_Code:                VERSCODE    //Kernel//;
      SCA_CTK_RefDate:                 DATEDOM     //Kernel//;  !!                SN-GEN-D3
      SCA_CTK_BeginValidity:           DATEDOM     //Kernel//;  !!                SN-GEN-D1
      SCA_CTK_EndValidity: OPTIONAL    DATEDOM     //Kernel//;  !!                SN-GEN-D2
      SCA_CTK_Comment:    OPTIONAL     COMMENT;
      SCA_CTK_ODO_Owner:               OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
      SCA_CTK_OrigSubDBID: DBID;                    !!             SN-GEN-A3
      SCA_CTK_DAO_Owner:               OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
      SCA_CTK_DataOwner:               DATAOWNER;               !!                SN-GEN-A5
      SCA_CTK_CreateDate:  DATEDOM;                 !!             SN-GEN-A8
      SCA_CTK_ChangeDate:  OPTIONAL    DATEDOM;                 !! SN-GEN-A9
      SCA_CTK_CHO_Owner:               OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
      SCA_CTK_ChangeUser:  ORAUSER;                 !!             SN-GEN-A11
      SCA_CTK_ITS_Code:                ITSCODE;                 !!                SN-GEN-A6
      SCA_CTK_IntegrityDate:           DATEDOM;                 !!                SN-GEN-A7
      SCA_CTK_XST_XfrSetID:            OPTIONAL    BASEID;      !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
        SCA_CTK_BaseID, SCA_CTK_Version, SCA_CTT_LNG_Code; !! PK
        SCA_CTK_CKO_Owner, SCA_CTK_CK, SCA_CTT_LNG_Code, SCA_CTK_RefDate, SCA_CTK_VRS_Code;
      !!   UKC
    END SimpleCatalog;

  End Catalog;
```

```
TABLE Document =
!! (D:‘Dokument‘ F:‘Document‘, SN640940/GEN-SN640940)
  DOC_BaseID:                          BASEID;                    !! PK              GEN-SN-I1
  DOC_Version:                         VERSION;                   !! PK              GEN-SN-I2
  DOC_CKO_Owner:                       OWNER       //Kernel//;   !! UKC(.)->OWN    SN-I1
  DOC_CK:                              CK5         //Kernel//;   !! UKC             SN-I2
  DOC_Description:     OPTIONAL        TEXT*72     //Kernel//;   !!                 SN-A3
  DOC_EditionDate:     OPTIONAL        DATEDOM     //Kernel//;   !!                 SN-A4
  DOC_SCA_DOT_CTK_BaseID: OPTIONAL BASEID       //Kernel//;   !!                 SN-A1.0
  DOC_SCA_DOT_CTK_CKO-Owner: OPTIONAL OWNER     //Kernel//;   !! FKC(1)->SCA    SN-A1.1
  DOC_SCA_DOT_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//; !! FKC(1)->SCA       SN-A1.2
  DOC_SCA_DOT_CTT_TextID: OPTIONAL TEXTID       //Kernel//;   !! FKC(1)->SCA    SN-A1.3
  DOC_TypAdText:       OPTIONAL        ADTEXT      //Kernel//;   !!                 SN-A3
  DOC_VRS_Code:                        VERSCODE    //Kernel//;   !!                 SN-GEN-A1
  DOC_RefDate:                         DATEDOM     //Kernel//;   !!                 SN-GEN-D3
  DOC_BeginValidity:                   DATEDOM     //Kernel//;   !!                 SN-GEN-D1
  DOC_EndValidity:     OPTIONAL        DATEDOM     //Kernel//;   !!                 SN-GEN-D2
  DOC_Comment:         OPTIONAL        COMMENT     //Kernel//;
  DOC_ODO_Owner:                       OWNER;                     !! FKL(.)->OWN    SN-GEN-A2
  DOC_OrigSubDBID:                     DBID;                      !!                 SN-GEN-A3
  DOC_DAO_Owner:                       OWNER;                     !! FKL(.)->OWN    SN-GEN-A4
  DOC_DataOwner:                       DATAOWNER;                 !!                 SN-GEN-A5
  DOC_CreateDate:                      DATEDOM;                   !!                 SN-GEN-A8
  DOC_ChangeDate:      OPTIONAL        DATEDOM;                   !!                 SN-GEN-A9
  DOC_CHO_Owner:                       OWNER;                     !! FKL(.)->OWN    SN-GEN-A10
  DOC_ChangeUser:                      ORAUSER;                   !!                 SN-GEN-A11
  DOC_ITS_Code:                        ITSCODE;                   !!                 SN-GEN-A6
  DOC_IntegrityDate:                   DATEDOM;                   !!                 SN-GEN-A7
  DOC_XST_XfrSetID:    OPTIONAL        BASEID;                      !!               FKL(.)->XST
  DOC_Filename:        OPTIONAL        TEXT*72     //Kernel//;
  DOC_Dop_Extension:   OPTIONAL        TEXT*3      //Kernel//;
CONSTRAINT
  UNIQUE
    DOC_BaseID,DOC_Version; !! PK
    DOC_CKO_Owner, DOC_CK; !! UKC
END Document;
```

```
TABLE SimpleSubject =
!! ( D:‘vereinfachter Beteiligter (denormalisierte, vereinfachte Struktur)‘
!!   F:‘Intervenant simplifié (structure simplifié et dénormalisée)‘,
!!   SN640940/GEN-SN640940)
  SSB_SBJ_BASEID:                  BASEID;                  !! PK             GEN-SN-I1
  SSB_SBJ_VERSION:                 VERSION;                 !! PK             GEN-SN-I2
  SSB_SBJ_CKO_Owner:               OWNER       //Kernel//;  !! UKC(.)->OWN    SN-I1
  SSB_SBJ_CK_SHORT:                CK1         //Kernel//;  !! UKC            SN-I2
  SSB_SUBJECT_CODE:                SUBJCODE    //Kernel//;  !!                SN-A1
  SSB_SBJ_LNG_CODE:                LANGCODE    //Kernel//;  !!                SN-A2
  SSB_PER_FIRSTNAME:   OPTIONAL    TEXT*30     //Kernel//;  !!                SN-A9
  SSB_PER_LASTNAME:    OPTIONAL    TEXT*30     //Kernel//;  !!                SN-A11
  SSB_PER_INITIAL:     OPTIONAL    TEXT*5      //Kernel//;  !!                SN-A10
  SSB_PER_PROFESSION: OPTIONAL     TEXT*30     //Kernel//;  !!                SN-A12
  SSB_PER_LNG_CODE_P: OPTIONAL     LANGCODE    //Kernel//;  !!                SN-A8
  SSB_PLN_FIRMNAME1:   OPTIONAL    TEXT*30     //Kernel//;  !!                SN-A3.1
  SSB_PLN_FIRMNAME2:   OPTIONAL    TEXT*30     //Kernel//;  !!                SN-A3.2
  SSB_PLN_BRANCH:      OPTIONAL    TEXT*30     //Kernel//;  !!                SN-A4
  SSB_PER_LNG_CODE_O: OPTIONAL     LANGCODE    //Kernel//;  !!                SN-A??
  SSB_SBJ_SBT_BaseID: OPTIONAL     BASEID      //Kernel//;  !! FKL(1)->SCA    SN-A???
  SSB_SBJ_SBT_CTK_CKO-Owner: OPTIONAL OWNER    //Kernel//;  !! FKC(1)->SCA    SN-A???
  SSB_SBJ_SBT_CTT_TextID: OPTIONAL TEXTID      //Kernel//;  !! FKC(1)->SCA    SN-A???
  SSB_SBJ_SBT_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//;   !! FKC(1)->SCA    SN-A???
  SSB_SBJ_LINKTYPADTEXT:OPTIONAL   ADTEXT      //Kernel//;  !!                SN-A???
  SSB_PER_PET_BaseID: OPTIONAL     BASEID      //Kernel//;  !! FKL(2)->SCA    SN-A6.0
  SSB_PER_PET_CTK_CKO-Owner: OPTIONAL OWNER    //Kernel//;  !! FKC(2)->SCA    SN-A6.1
  SSB_PER_PET_CTT_TextID: OPTIONAL TEXTID      //Kernel//;  !! FKC(2)->SCA    SN-A6.2
  SSB_PER_PET_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//; !! FKC(2)->SCA      SN-A6.3
  SSB_PER_HEADTYPADTEXT: OPTIONAL ADTEXT       //Kernel//;  !!                SN-A7
  SSB_PLN_LNG_CODE:    OPTIONAL    LANGCODE    //Kernel//;
  SSB_ADR_STREET:                  TEXT*30     //Kernel//;  !!                SN-A13
  SSB_ADR_HOUSENR:     OPTIONAL    NUM5        //Kernel//;  !!                SN-A14
  SSB_ADR_HOUSENRADTEXT: OPTIONAL TEXT*7       //Kernel//;  !!                SN-A15
  SSB_ADR_ADDADDRESS: OPTIONAL     TEXT*30     //Kernel//;  !!                SN-A16
  SSB_ADR_LNG_CODE:                LANGCODE    //Kernel//;  !!                SN-A17
  SSB_ADR_PLACE:                   TEXT*24     //Kernel//;  !!                SN-A18
  SSB_ADR_ZIP:                     TEXT*8      //Kernel//;  !!                SN-A19
  SSB_ADR_Nation:                  TEXT*3      //Kernel//;  !!                SN-A20
  SSB_SBJ_SUBJECTADTEXT: OPTIONAL TEXT*30      //Kernel//;  !!                SN-A21
  SSB_SBJ_VRS_Code:                VERSCODE    //Kernel//;  !! UKC            SN-GEN-A1
  SSB_SBJ_RefDate:                 DATEDOM     //Kernel//;  !! UKC            SN-GEN-D3
  SSB_SBJ_BeginValidity:           DATEDOM     //Kernel//;  !!                SN-GEN-D1
  SSB_SBJ_EndValidity: OPTIONAL    DATEDOM     //Kernel//;  !!                SN-GEN-D2
  SSB_SBJ_Comment:     OPTIONAL    COMMENT     //Kernel//;
  SSB_SBJ_ODO_Owner:               OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  SSB_SBJ_OrigSubDBID: DBID;                    !!            SN-GEN-A3
  SSB_SBJ_DAO_Owner:               OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  SSB_SBJ_DataOwner:               DATAOWNER;               !!                SN-GEN-A5
  SSB_SBJ_CreateDate:  DATEDOM;                 !!            SN-GEN-A8
  SSB_SBJ_ChangeDate: OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
  SSB_SBJ_CHO_Owner:               OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  SSB_SBJ_ChangeUser:  ORAUSER;                 !!            SN-GEN-A11
  SSB_SBJ_ITS_Code:                ITSCODE;                 !!                SN-GEN-A6
  SSB_SBJ_IntegrityDate:           DATEDOM;                 !!                SN-GEN-A7
  SSB_SBJ_XST_XfrSetID: OPTIONAL   BASEID;                     !!             FKL(.)->XST
CONSTRAINT
  UNIQUE
    SSB_SBJ_BASEID, SSB_SBJ_VERSION; !! PK
    SSB_SBJ_CKO_Owner, SSB_SBJ_CK_SHORT; !! UKC
END SimpleSubject;
```

```
TABLE Axes =
!!(D:‘Axe‘ F:‘Axe‘, SN640941/GEN-SN640940)
  AXE_BaseID:                       BASEID;                 !! PK             SN-GEN-I1
  AXE_Version:                      VERSION;                !! PK             SN-GEN-I2
  AXE_CKO_Owner:                    OWNER       //Kernel//; !! UKC->OWN       SN-I1
  AXE_CK:                           CK3         //Kernel//; !! UKC            SN-I2
  AXE_POS_Code:                     POSCODE     //Kernel//; !! UKC            SN-I3
  AXE_Name:           OPTIONAL      NAME        //Kernel//; !!                SN-A2
  AXE_SCA_AXT_CTK_BaseID:           BASEID      //Kernel//; !! FKL(1)->SCA    SN-A1.0
  AXE_SCA_AXT_CTK_CKO_Owner:        OWNER       //Kernel//; !! FKC(1)->SCA    SN-A1.1
  AXE_SCA_AXT_CTT_LNG_Code:         LANGCODE    //Kernel//; !! FKC(1)->SCA    SN-A1.2
  AXE_SCA_AXT_CTT_TextID:           TEXTID      //Kernel//; !! FKC(1)->SCA    SN-A1.3
  AXE_TypAdText:      OPTIONAL      ADTEXT      //Kernel//; !!                SN-A3
  AXE_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
  AXE_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
  AXE_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
  AXE_EndValidity:    OPTIONAL      DATEDOM     //Kernel//; !!                SN-GEN-D2
  AXE_Comment:        OPTIONAL      COMMENT     //Kernel//; !!
  AXE_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
  AXE_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
  AXE_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
  AXE_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
  AXE_CreateDate:                   DATETIME;               !!                SN-GEN-A8
  AXE_ChangeDate:     OPTIONAL      DATETIME;               !!                SN-GEN-A9
  AXE_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
  AXE_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
  AXE_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
  AXE_IntegrityDate:                DATETIME;               !!                SN-GEN-A7
  AXE_XST_XfrSetID:   OPTIONAL      BASEID;                 !! FKL(.)->XST
CONSTRAINT
  UNIQUE
     AXE_BaseID, AXE_Version; !! PK
     AXE_CKO_Owner, AXE_CK, AXE_POS_Code; !! UKC
END Axis;
```

```
    TABLE Reference_PS =
    !!( D:‘Bezugspunkt‘ F:‘Point de référence‘, SN640941/GEN-SN640940)
      RPT_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
      RPT_Version:                      VERSION;                 !! PK             SN-GEN-I2
      RPT_AXE_BaseID:                   BASEID      //Kernel//;  !! FKL(1)->AXE    SN-I1.0
      RPT_AXE_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1)->AXE    SN-I1.1
      RPT_AXE_CK:                       CK3         //Kernel//;  !! UKC(1)->AXE    SN-I1.2
      RPT_AXE_POS_Code:                 POSCODE     //Kernel//;  !! UKC(1)->AXE    SN-I1.3
      RPT_AXE_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1)->AXE    SN-I1.4
      RPT_CK:                           CK2         //Kernel//;  !! UKC            SN-I2
      RPT_SCA_RET_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(2)->SCA    SN-A1.0
      RPT_SCA_RET_CTK_CKO_Owner:        OWNER       //Kernel//;  !! FKC(2)->SCA    SN-A1.1
      RPT_SCA_RET_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! FKC(2)->SCA    SN-A1.2
      RPT_SCA_RET_CTT_TextID:           TEXTID      //Kernel//;  !! FKC(2)->SCA    SN-A1.3
      RPT_TypAdText:       OPTIONAL     ADTEXT      //Kernel//;  !!                SN-A5
      RPT_Name:            OPTIONAL     NAME        //Kernel//;  !!                SN-A7
      RPT_SortKey:                      NUM8        //Kernel//;  !!                SN-A2
      RPT_SectorL:                      NUM4_3      //Kernel//;  !!                SN-A3
      RPT_SLPrecision:     OPTIONAL     NUM2_3      //Kernel//;  !!                SN-A6
      RPT_LatDist:                      NUM2_3      //Kernel//;  !!                SN-A4
      RPT_km_Number:       OPTIONAL     NUM4_3      //Kernel//;  !!                SN-A8
      RPT_CoorY:           OPTIONAL     NUM6_3      //Kernel//;  !!                SN-A9
      RPT_CoorX:           OPTIONAL     NUM6_3      //Kernel//;  !!                SN-A10
      RPT_PreciXY:         OPTIONAL     NUM2_3      //Kernel//;  !!                SN-A11
      RPT_CoorZ:           OPTIONAL     NUM4_3      //Kernel//;  !!                SN-A12
      RPT_PreciZ:          OPTIONAL     NUM2_3      //Kernel//;  !!                SN-A13
      RPT_PlateAxisDist:   OPTIONAL     NUM2_2      //Kernel//;  !!                SN-A14
      RPT_PlateRefDist:    OPTIONAL     NUM2_2      //Kernel//;  !!                SN-A15
      RPT_MunicipalNum:    OPTIONAL     TEXT*4      //Kernel//;  !!                SN-A16
      RPT_AdditionalInfo: OPTIONAL      TEXT*4      //Kernel//;  !!                SN-A17
      RPT_SCA_PAT_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(3)->SCA    SN-A18.0
      RPT_SCA_PAT_CTK_CKO_Owner:        OWNER       //Kernel//;  !! FKC(3)->SCA    SN-A18.1
      RPT_SCA_PAT_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! FKC(3)->SCA    SN-A18.2
      RPT_SCA_PAT_CTT_TextID:           TEXTID      //Kernel//;  !! FKC(3)->SCA    SN-A18.3
      RPT_TypAdText2:      OPTIONAL     ADTEXT      //Kernel//;  !!                SN-A19
      RPT_SCA_PPT_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(4)->SCA    SN-A20.0
      RPT_SCA_PPT_CTK_CKO_Owner:        OWNER       //Kernel//;  !! FKC(4)->SCA    SN-A20.1
      RPT_SCA_PPT_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! FKC(4)->SCA    SN-A20.2
      RPT_SCA_PPT_CTT_TextID:           TEXTID      //Kernel//;  !! FKC(4)->SCA    SN-A20.3
      RPT_TypAdText3:      OPTIONAL     ADTEXT      //Kernel//;  !!                SN-A21
      RPT_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
      RPT_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
      RPT_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
      RPT_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
      RPT_Comment:         OPTIONAL     COMMENT     //Kernel//;
      RPT_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
      RPT_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
      RPT_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
      RPT_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
      RPT_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
      RPT_ChangeDate:      OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
      RPT_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
      RPT_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
      RPT_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
      RPT_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
      RPT_XST_XfrSetID:    OPTIONAL     BASEID;                     !!             FKL(.)->XST
    CONSTRAINT
      UNIQUE
RPT_BaseID, RPT_Version;  !!PK
RPT_AXE_CKO_Owner, RPT_AXE_CK, RPT_AXE_POS_Code, RPT_AXE_VRS_Code, RPT_CK;  !!UKC
    END Reference_Point;
```

```
TABLE SR_Points =
!! ( D:‘Hilfsbezugspunkte‘ F:‘Points de repère d‘aide‘, SN640941/GEN-SN640940)
  SRP_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
  SRP_Version:                      VERSION;                 !! PK             SN-GEN-I2
  SRP_CKO_Owner:                    OWNER       //Kernel//;  !! UKC            SN-I1
  SRP_CK:                           CK4         //Kernel//;  !! UKC            SN-I2
  SRP_AXE_CKO_Owner:                OWNER       //Kernel//;  !! FKC(1)->AXE    SN-A1.1
  SRP_AXE_CK:                       CK3         //Kernel//;  !! FKC(1)->AXE    SN-A1.2
  SRP_AXE_POS_Code:                 POSCODE     //Kernel//;  !! FKC(1)->AXE    SN-A1.3
  SRP_AXE_VRS_Code:                 VERSCODE    //Kernel//;  !! FKC(1)->AXE    SN-A1.4
  SRP_RPT_BaseID:                   BASEID      //Kernel//;  !! FKL(2)->RPT    SN-A1.10
  SRP_RPT_CK:                       CK2         //Kernel//;  !! FKC(2)->RPT    SN-A1.11
  SRP_RPT_VRS_Code:                 VERSCODE    //Kernel//;  !! FKC(2)->RPT    SN-A1.12
  SRP_RefpointDist:                 U_ABS       //Kernel//;  !!                SN-A1.13
  SRP_LatDist:                      NUM2_3      //Kernel//;  !!                SN-A1.14
  SRP_Name:           OPTIONAL      NAME        //Kernel//;  !!                SN-A2
  SRP_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  SRP_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  SRP_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  SRP_EndValidity:    OPTIONAL      DATEDOM     //Kernel//;  !!                SN-GEN-D2
  SRP_Comment:        OPTIONAL      COMMENT     //Kernel//;
  SRP_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  SRP_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  SRP_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  SRP_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  SRP_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  SRP_ChangeDate:     OPTIONAL      DATEDOM;                 !!                SN-GEN-A9
  SRP_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  SRP_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  SRP_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  SRP_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  SRP_XST_XfrSetID:   OPTIONAL      BASEID;                     !!             FKL(.)->XST
CONSTRAINT
UNIQUE
  SRP_BaseID, SRP_Version; !! PK
  SRP_CKO_Owner, SRP_CK; !! UKC
END Sec_Reference_Point;
```

```
    TABLE Link_Groups =
    !! (D:‘Abschnittsgruppe‘ F:‘Groupe de tronçon‘, SN640941/GEN-SN640940)
      LKG_BaseID:                       BASEID;                 !! PK             SN-GEN-I1
      LKG_Version:                      VERSION;                !! PK             SN-GEN-I2
      LKG_CKO_Owner:                    OWNER       //Kernel//; !! UKC(.)->OWN    SN-I1
      LKG_CK:                           CK2         //Kernel//; !! UKC            SN-I2
      LKG_SCA_SLT_CTK_BaseID:           BASEID      //Kernel//; !! FKL(2)->SCA    SN-A3.0
      LKG_SCA_SLT_CTK_CKO_Owner:        OWNER       //Kernel//; !! UKC(2)->SCA    SN-A3.1
      LKG_SCA_SLT_CTT_LNG_Code:         LANGCODE    //Kernel//; !! UKC(2)->SCA    SN-A3.2
      LKG_SCA_SLT_CTT_TextID:           TEXTID      //Kernel//; !! UKC(2)->SCA    SN-A3.3
      LKG_TypAdText:       OPTIONAL     ADTEXT      //Kernel//; !!                SN-A1
      LKG_LKG_BaseID:      OPTIONAL     BASEID      //Kernel//; !! FKL(3)->LKG
      LKG_LKG_CKO_Owner:   OPTIONAL     OWNER       //Kernel//; !! FKC(3)->LKG
      LKG_LKG_CK:          OPTIONAL     CK2         //Kernel//; !! FKC(3)->LKG
      LKG_LKG_VRS_Code:    OPTIONAL     VERSCODE    //Kernel//; !! FKC(3)->LKG
      LKG_Value:           OPTIONAL     NUM7_4      //Kernel//; !!                SN-A2
      LKG_Name:            OPTIONAL     NAME        //Kernel//; !!                SN-A3
      LKG_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
      LKG_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
      LKG_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
      LKG_EndValidity:     OPTIONAL     DATEDOM     //Kernel//; !!                SN-GEN-D2
      LKG_Comment:         OPTIONAL     COMMENT     //Kernel//;
      LKG_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
      LKG_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
      LKG_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
      LKG_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
      LKG_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
      LKG_ChangeDate:      OPTIONAL     DATEDOM;                !!                SN-GEN-A9
      LKG_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
      LKG_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
      LKG_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
      LKG_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
      LKG_XST_XfrSetID:    OPTIONAL     BASEID;                 !! FKL(.)->XST
CONSTRAINT
      UNIQUE
        LKG_BaseID, LKG_Version;  !! PK
        LKG_CKO_Owner, LKG_CK, LKG_SCA_SLT_CAT_CKO_Owner LKG_SCA_SLT_CAT_CK
        LKG_SCA_SLT_CTK_CKO_Owner, LKG_SCA_SLT_CTT_LNG_Code, LKG_SCA_SLT_CTT_TextID; !! UKC
    END Link_Group;
```

```
TABLE Projects =
!! (D:‘Projekt‘ F:‘Projet‘, GEN-SN640940)
  PRO_BaseID:                        BASEID;                  !! PK             SN-GEN-I1
  PRO_Version:                       VERSION;                 !! PK             SN-GEN-I2
  PRO_CKO_Owner:                     OWNER       //Kernel//;  !! UKC(.)->OWN
  PRO_CK:                            CK4         //Kernel//;  !! UKC
  PRO_Name:                          NAME        //Kernel//;
  PRO_Year:                          NUM4        //Kernel//;
  PRO_ForNum:          OPTIONAL      NUM4        //Kernel//;
  PRO_SCA_PRT_CTK_BaseID:            BASEID      //Kernel//;  !! FKL(1)->SCA
  PRO_SCA_PRT_CTK_CKO_Owner:         OWNER       //Kernel//;  !! FKC(1)->SCA
  PRO_SCA_PRT_CTT_LNG_Code:          LANGCODE    //Kernel//;  !! FKC(1)->SCA
  PRO_SCA_PRT_CTT_TextID:            TEXTID      //Kernel//;  !! FKC(1)->SCA
  PRO_TypAdText:       OPTIONAL      ADTEXT      //Kernel//;
  PRO_LKG_BaseID:      OPTIONAL      BASEID      //Kernel//;  !! FKL(2)->LKG
  PRO_LKG_CKO_Owner:   OPTIONAL      OWNER       //Kernel//;  !! FKC(2)->LKG
  PRO_LKG_CK:                        CK4         //Kernel//;  !! FKC(2)->LKG
  PRO_LKG_SCA_SLT_CTK_BaseID: OPTIONAL BASEID   //Kernel//;  !! FKL(3)->LKG
  PRO_LKG_SCA_SLT_CTK_CKO-Owner: OPTIONAL OWNER //Kernel//;  !! FKC(3)->LKG
  PRO_LKG_SCA_SLT_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//;!! FKC(3)->LKG
  PRO_LKG_SCA_SLT_CTT_TextID: OPTIONAL TEXTID   //Kernel//;  !! FKC(3)->LKG
  PRO_Costs:                         NUM9_2      //Kernel//;
  PRO_OwnerAccount:    OPTIONAL      TEXT*16     //Kernel//;
  PRO_VRS_Code:                      VERSCODE    //Kernel//;  !!                SN-GEN-A1
  PRO_RefDate:                       DATEDOM     //Kernel//;  !!                SN-GEN-D3
  PRO_BeginValidity:                 DATEDOM     //Kernel//;  !!                SN-GEN-D1
  PRO_EndValidity:     OPTIONAL      DATEDOM     //Kernel//;  !!                SN-GEN-D2
  PRO_Comment:         OPTIONAL      COMMENT     //Kernel//;
  PRO_ODO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  PRO_OrigSubDBID:                   DBID;                    !!                SN-GEN-A3
  PRO_DAO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  PRO_DataOwner:                     DATAOWNER;               !!                SN-GEN-A5
  PRO_CreateDate:                    DATEDOM;                 !!                SN-GEN-A8
  PRO_ChangeDate:      OPTIONAL      DATEDOM;                 !!                SN-GEN-A9
  PRO_CHO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  PRO_ChangeUser:                    ORAUSER;                 !!                SN-GEN-A11
  PRO_ITS_Code:                      ITSCODE;                 !!                SN-GEN-A6
  PRO_IntegrityDate:                 DATEDOM;                 !!                SN-GEN-A7
  PRO_XST_XfrSetID:    OPTIONAL      BASEID;                     !!             FKL(.)->XST
CONSTRAINT
  UNIQUE
    PRO_BaseID, PRO_Version; !! PK
    PRO_CKO_OWNER, PRO_CK; !! UKC
END Project;
```

```
TABLE Junctions =
!! (D:‘Anschluss‘ F:‘Jonction‘, SN640941/GEN-SN640940)
  JNC_BaseID:                        BASEID;                  !! PK             SN-GEN-I1
  JNC_Version:                       VERSION;                 !! PK             SN-GEN-I2
  JNC_CKO_Owner:                     OWNER       //Kernel//;  !! UKC(.)->OWN    SN-I1
  JNC_CK:                            CK4         //Kernel//;  !! UKC            SN-I2
  JNC_Name:           OPTIONAL       NAME        //Kernel//;  !!                SN-A2
  JNC_SCA_JCT_CTK_BaseID:            BASEID      //Kernel//;  !! FKL(1)->SCA    SN-A1.0
  JNC_SCA_JCT_CTK_CKO_Owner:         OWNER       //Kernel//;  !! FKC(1)->SCA    SN-A1.1
  JNC_SCA_JCT_CTT_LNG_Code:          LANGCODE    //Kernel//;  !! FKC(1)->SCA    SN-A1.2
  JNC_SCA_JCT_CTT_TextID:            TEXTID      //Kernel//;  !! FKC(1)->SCA    SN-A1.3
  JNC_TypAdText:      OPTIONAL       ADTEXT      //Kernel//;  !!                SN-A3
  JNC_VRS_Code:                      VERSCODE    //Kernel//;  !!                SN-GEN-A1
  JNC_RefDate:                       DATEDOM     //Kernel//;  !!                SN-GEN-D3
  JNC_BeginValidity:                 DATEDOM     //Kernel//;  !!                SN-GEN-D1
  JNC_EndValidity:    OPTIONAL       DATEDOM     //Kernel//;  !!                SN-GEN-D2
  JNC_Comment:        OPTIONAL       COMMENT     //Kernel//;
  JNC_ODO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  JNC_OrigSubDBID:                   DBID;                    !!                SN-GEN-A3
  JNC_DAO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  JNC_DataOwner:                     DATAOWNER;               !!                SN-GEN-A5
  JNC_CreateDate:                    DATEDOM;                 !!                SN-GEN-A8
  JNC_ChangeDate:     OPTIONAL       DATEDOM;                 !!                SN-GEN-A9
  JNC_CHO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  JNC_ChangeUser:                    ORAUSER;                 !!                SN-GEN-A11
  JNC_ITS_Code:                      ITSCODE;                 !!                SN-GEN-A6
  JNC_IntegrityDate:                 DATEDOM;                 !!                SN-GEN-A7
  JNC_XST_XfrSetID:   OPTIONAL       BASEID;                     !!             FKL(.)->XST
CONSTRAINT
  UNIQUE
    JNC_BaseID, JNC_Version; !! PK
    JNC_CKO_Owner, JNC_CK; !! UNC
END Junction;
```

```
TABLE Nodes =
!! ( D:‘Knoten‘ F:‘Nœud‘, SN640941/GEN-SN640940)
  NOD_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
  NOD_Version:                      VERSION;                 !! PK             SN-GEN-I2
  NOD_CKO_Owner:                    OWNER       //Kernel//;  !! UKC(.)->OWN    SN-I1
  NOD_CK:                           CK4         //Kernel//;  !! UKC            SN-I2
  NOD_Name:           OPTIONAL      NAME        //Kernel//;  !!                 SN-A3
  NOD_SCA_NOT_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(1)->SCA    SN-A1.0
  NOD_SCA_NOT_CTK_CKO_Owner:        OWNER       //Kernel//;  !! FKC(1)->SCA    SN-A1.1
  NOD_SCA_NOT_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! FKC(1)->SCA    SN-A1.2
  NOD_SCA_NOT_CTT_TextID:           TEXTID      //Kernel//;  !! FKC(1)->SCA    SN-A1.3
  NOD_TypAdText:      OPTIONAL      ADTEXT      //Kernel//;  !!                 SN-A4
  NOD_NOD_BaseID:     OPTIONAL      BASEID      //Kernel//;  !! FKL(2)->NOD    SN-A5.0
  NOD_NOD_CKO_Owner:  OPTIONAL      OWNER       //Kernel//;  !! FKC(2)->NOD    SN-A5.1
  NOD_NOD_CK:         OPTIONAL      CK4         //Kernel//;  !! FKC(2)->NOD    SN-A5.2
  NOD_NOD_VRS_Code:   OPTIONAL      VERSCODE    //Kernel//;  !! FKC(2)->NOD    SN-A5.3
  NOD_JNC_BaseID:     OPTIONAL      BASEID      //Kernel//;  !! FKL(3)->JCT    SN-A6.0
  NOD_JNC_CKO_Owner:  OPTIONAL      OWNER       //Kernel//;  !! FKC(3)->JCT    SN-A6.1
  NOD_JNC_CK:         OPTIONAL      CK4         //Kernel//;  !! FKC(3)->JCT    SN-A6.2
  NOD_JNC_VRS_Code:   OPTIONAL      VERSCODE    //Kernel//;  !! FKC(3)->JCT    SN-A6.3
  NOD_Branches:       OPTIONAL      NUM2        //Kernel//;  !!                 SN-A7
  NOD_Levels:         OPTIONAL      NUM2        //Kernel//;  !!                 SN-A8
  NOD_CoorY:          OPTIONAL      NUM6_3      //Kernel//;  !!                 SN-A9
  NOD_CoorX           OPTIONAL      NUM6_3      //Kernel//;  !!                 SN-A10
  NOD_CoorZ:          OPTIONAL      NUM4_3      //Kernel//;
  NOD_VRS_Code:                     VERSCODE    //Kernel//;  !!                 SN-GEN-A1
  NOD_RefDate:                      DATEDOM     //Kernel//;  !!                 SN-GEN-D3
  NOD_BeginValidity:                DATEDOM     //Kernel//;  !!                 SN-GEN-D1
  NOD_EndValidity:    OPTIONAL      DATEDOM     //Kernel//;  !!                 SN-GEN-D2
  NOD_Comment:        OPTIONAL      COMMENT     //Kernel//;
  NOD_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  NOD_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  NOD_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  NOD_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  NOD_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  NOD_ChangeDate:     OPTIONAL      DATEDOM;                 !!                SN-GEN-A9
  NOD_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  NOD_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  NOD_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  NOD_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  NOD_XST_XfrSetID:   OPTIONAL      BASEID;                     !!             FKL(.)->XST
CONSTRAINT
  UNIQUE
    NOD_BaseID, NOD_Version; !! PK
    NOD_CKO_Owner, NOD_CK;!! UKC
END Node;
```

```
TABLE Node_Locs =
!! ( D:‘Knotenorte‘ F:‘Lieu de nœud‘, SN640941-Node-A2/GEN-SN640940)
  NLO_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
  NLO_Version:                      VERSION;                 !! PK             SN-GEN-I2
  NLO_NOD_BaseID:                   BASEID      //Kernel//;  !! FKL(1)->NOD
  NLO_NOD_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1)->NOD
  NLO_NOD_CK:                       CK4         //Kernel//;  !! UKC(1)->NOD
  NLO_NOD_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1)->NOD
  NLO_RPT_BaseID:                   BASEID      //Kernel//;  !! FKL(2)->RPT
  NLO_RPT_AXE_CKO_Owner:            OWNER       //Kernel//;  !! UKC(2)->RPT
  NLO_RPT_AXE_CK:                   CK3         //Kernel//;  !! UKC(2)->RPT
  NLO_RPT_AXE_POS_Code:             POSCODE     //Kernel//;  !! UKC(2)->RPT
  NLO_RPT_AXE_VRS_Code:             VERSCODE    //Kernel//;  !! UKC(2)->RPT
  NLO_RPT_CK:                       CK2         //Kernel//;  !! UKC(2)->RPT
  NLO_RPT_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(2)->RPT
  NLO_RefPointDist:                 U_ABS       //Kernel//;
  NLO_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  NLO_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  NLO_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  NLO_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
  NLO_Comment:         OPTIONAL     COMMENT     //Kernel//;
  NLO_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  NLO_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  NLO_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  NLO_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  NLO_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  NLO_ChangeDate:      OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
  NLO_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  NLO_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  NLO_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  NLO_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  NLO_XST_XfrSetID:    OPTIONAL     BASEID;                  !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    NLO_BaseID, NLO_Version; !! PK
    NLO_NOD_CKO_Owner, NLO_NOD_CK, NLO_NOD_VRS_Code, NLO_RPT_AXE_CKO_Owner,
    NLO_RPT_AXE_CK, NLO_RPT_AXE_POS_Code, NLO_RPT_AXE_VRS_Code, NLO_RPT_CK,
    NLO_RPT_VRS_Code; !! UKC
END Node_Location;
```

```
TABLE Links =
!! ( D:‘Abschnitt‘ F:‘Tronçon‘, SN640941/GEN-SN640940)
  LNK_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
  LNK_Version:                      VERSION;                 !! PK             SN-GEN-I2
  LNK_AXE_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1)->AXE    SN-I1.1
  LNK_AXE_CK:                       CK3         //Kernel//;  !! UKC(1)->AXE    SN-I1.2
  LNK_AXE_POS_Code:                 POSCODE     //Kernel//;  !! UKC(1)->AXE    SN-I1.3
  LNK_AXE_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1)->AXE    SN-I1.1
  LNK_NLO_BaseID_1:                 BASEID;     //Kernel//;  !! FKL(2)->NLO    SN-I2.0
  LNK_NLO_NOD_BaseID_1:             BASEID;     //Kernel//;  !! FKL(.)->NLO    SN-I2.1
  LNK_NLO_NOD_CKO_Owner_1:          OWNER       //Kernel//;  !! UKC(2)->NLO    SN-I2.2
  LNK_NLO_NOD_CK_1:                 CK4         //Kernel//;  !! UKC(2)->NLO    SN-I2.3
  LNK_NLO_NOD_VRS_CODE:             VERSCODE    //Kernel//;  !! UKC(2)->NLO    SN-I2.4
  LNK_NLO_RPT_BaseID_1:             BASEID;     //Kernel//;  !! FKL(.)->NLO    SN-I2.5
  LNK_NLO_RPT_CK_1:                 CK2         //Kernel//;  !! UKC(2)->NLO    SN-I2.6
  LNK_NLO_RPT_VRS_Code_1:           VERSCODE    //Kernel//;  !! UKC(2)->NLO    SN-I2.7
  LNK_NLO_VRS_Code_1:               VERSCODE    //Kernel//;  !! UKC(2)->NLO    SN-I2.8
  LNK_NLO_BaseID_2:                 BASEID      //Kernel//;  !! FKL(3)->NLO    SN-I3.0
  LNK_NLO_NOD_BaseID_2:             BASEID      //Kernel//;  !! FKL(.)->NLO    SN-I3.1
  LNK_NLO_NOD_CKO_Owner_2:          OWNER       //Kernel//;  !! UKC(3)->NLO    SN-I3.2
  LNK_NLO_NOD_CK_2:                 CK4         //Kernel//;  !! UKC(3)->NLO    SN-I3.3
  LNK_NLO_NOD_VRS_CODE_2:           VERSCODE    //Kernel//;  !! UKC(3)->NLO    SN-I3.4
  LNK_NLO_RPT_BaseID_2:             BASEID;     //Kernel//;  !! FKL(.)->NLO    SN-I3.5
  LNK_NLO_RPT_CK_2:                 CK3         //Kernel//;  !! UKC(3)->NLO    SN-I3.6
  LNK_NLO_RPT_VRS_Code_2:           VERSCODE    //Kernel//;  !! UKC(3)->NLO    SN-I3.7
  LNK_NLO_VRS_Code_2:               VERSCODE    //Kernel//;  !! UKC(3)->NLO    SN-I3.8
  LNK_LKG_BaseID:                   BASEID      //Kernel//;  !! FKL(4)->LKG    SN-I4.0
  LNK_LKG_CKO_Owner:                OWNER       //Kernel//;  !! UKC(4)->LKG    SN-I4.1
  LNK_LKG_CK:                       CK4         //Kernel//;  !! UKC(4)->LKG    SN-I4.2
  LNK_LKG_SCA_SLT_CTK_BaseID:       BASEID      //Kernel//;  !! FKL(.)->LKG    SN-I4.30
  LNK_LKG_SCA_SLT_CTK_CKO_Owner:    OWNER       //Kernel//;  !! UKC(4)->LKG    SN-A4.31
  LNK_LKG_SCA_SLT_CTT_LNG_Code:     LANGCODE    //Kernel//;  !! UKC(4)->LKG    SN-A4.32
  LNK_LKG_SCA_SLT_CTT_TextID:       TEXTID      //Kernel//;  !! UKC(4)->LKG    SN-A4.33
  LNK_LKG_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(4)->LKG    SN-A4.4
  LNK_Sortkey:                      NUM4        //Kernel//;  !!                SN-A1
  LNK_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  LNK_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  LNK_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  LNK_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
  LNK_Comment:         OPTIONAL     COMMENT     //Kernel//;
  LNK_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  LNK_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  LNK_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  LNK_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  LNK_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  LNK_ChangeDate:      OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
  LNK_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  LNK_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  LNK_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  LNK_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  LNK_XST_XfrSetID:    OPTIONAL     BASEID;                  !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    LNK_BaseID, LNK_Version; !! PK
    LNK_AXE_CKO_Owner, LNK_AXE_CK, LNK_AXE_POS_Code, LNK_AXE_VRS_Code,
    LNK_NLO_NOD_CKO_Owner_1, LNK_NLO_NOD_CK_1, LNK_NLO_NOD_VRS_Code_1, LNK_NLO_RPT_CK_1,
    LNK_NLO_RPT_VRS_Code_1, LNK_NLO_VRS_Code_1,
    LNK_NLO_NOD_CKO_Owner_2, LNK_NLO_NOD_CK_2, LNK_NLO_NOD_VRS_Code_2, LNK_NLO_RPT_CK_2,
    LNK_NLO_RPT_VRS_Code_2, LNK_NLO_VRS_Code_2,
    LNK_LKG_CKO_Owner, LNK_LKG_CK, LNK_LKG_SCA_SLT_CAT_CKO_Owner, LNK_LKG_SCA_SLT_CAT_CK,
    LNK_LKG_SCA_SLT_CTK_CKO_Owner, LNK_LKG_SCA_SLT_CTT_LNG_Code,
    LNK_LKG_SCA_SLT_CTT_TEXTID; !! UKC
END Link;
```

```
TABLE Geometries =
!! (D:'Geometrien', F:'Géométries')
  GEO_BaseID                        BASEID;                      !!            PK
  GEO_Version                       VERSION;                     !!            PK
  GEO_CKO_Owner                     OWNER       //Kernel//;   !! UKC→OWN
  GEO_CK                            CK4         //Kernel//;   !! UKC
  GEO_Name             OPTIONAL     NAME        //Kernel//;
  GEO_PRO_BaseID                    BASEID      //Kernel//;   !! FKL(1)→PRO
  GEO_PRO_CKO_Owner                 OWNER       //Kernel//;   !! FKC(1)→PRO
  GEO_PRO_CK                        CK          //Kernel//;   !! FKC(1)→PRO
  GEO_PRO_VRS_Code                  VERSCODE    //Kernel//;   !! FKC(1)→PRO
  GEO_Usage_Code                    USAGE_CODE  //Kernel//;
  GEO_Tool_Code                     TOOLCODE    //Kernel//;
  GEO_Scale            OPTIONAL     NUM8        //Kernel//;
  GEO_SCA_CTK_BaseID   OPTIONAL     BASEID      //Kernel//;   !! FKL(2)→SCA
  GEO_SCA_CTK_CKO_Owner             OWNER       //Kernel//;   !! FKC(2)→CTK
  GEO_SCA_CTT_LNG_Code              LANGCODE    //Kernel//;   !! FKC(2)→CTK
  GEO_SCA_CTT_TextID                TEXTID      //Kernel//;   !! FKC(2)→CTK
  GEO_TypAdText        OPTIONAL     ADTEXT      //Kernel//;
  GEO_Comment          OPTIONAL     COMMENT     //Kernel//;
  GEO_RefDate                       DATETIME    //Kernel//;
  GEO_BeginValidity                 DATETIME    //Kernel//;
  GEO_EndValidity      OPTIONAL     DATETIME    //Kernel//;
  GEO_VRS_Code                      VERSCODE    //Kernel//;
  GEO_ODO_Owner                     OWNER;                   !! FKL(.)→OWN
  GEO_OrigSubDBID                   DBID;
  GEO_DAO_Owner                     OWNER;                   !! FKL(.)→OWN
  GEO_DataOwner                     DATAOWNER;
  GEO_CreateDate                    DATETIME;
  GEO_ChangeDate       OPTIONAL     DATETIME;
  GEO_CHO_Owner                     OWNER;                   !! FKL(.)→OWN
  GEO_ChangeUser                    ORAUSER;
  GEO_Its_Code                      ITSCODE;
  GEO_IntegrityDate                 DATETIME;
  GEO_XST_XfrSetID     OPTIONAL     BASEID;                  !! FKL(.)→XST
CONSTRAINT
  UNIQUE
    GEO_BaseID, GEO_Version;!! PK
    GEO_CKO_Owner, GEO_CK; !! UKC
END Geometries;
```

```
TABLE Hor_Segments =
!! (D:'Horizontal-Segmente', F:'Segments horizontaux')
  HOS_BaseID                        BASEID;                       !!            PK
  HOS_Version                       VERSION;                      !!            PK
  HOS_GEO_BaseID                    BASEID      //Kernel//;   !! FKL(1)→GEO
  HOS_GEO_Version                   VERSION     //Kernel//;   !! FKL(1)→GEO
  HOS_GEO_CKO_Owner                 OWNER       //Kernel//;   !! UKC(1)→GEO
  HOS_GEO_CK                        CK4         //Kernel//;   !! UKC(1)→GEO
  HOS_GEO_VRS_Code                  VERSCODE    //Kernel//;   !! UKC(1)→GEO
  HOS_StandardSeq                   NUM38       //Kernel//;   !! UKC
  HOS_Name            OPTIONAL      NAME        //Kernel//;
  HOS_Comment         OPTIONAL      COMMENT     //Kernel//;
  HOS_RefDate                       DATETIME    //Kernel//;
  HOS_BeginValidity                 DATETIME    //Kernel//;
  HOS_EndValidity     OPTIONAL      DATETIME    //Kernel//;
  HOS_VRS_Code                      VERSCODE    //Kernel//;
  HOS_ODO_Owner                     OWNER;                    !! FKL(.)→OWN
  HOS_OrigSubDBID                   DBID;
  HOS_DAO_Owner                     OWNER;                    !! FKL(.)→OWN
  HOS_DataOwner                     DATAOWNER;
  HOS_CreateDate                    DATETIME;
  HOS_ChangeDate      OPTIONAL      DATETIME;
  HOS_CHO_Owner                     OWNER;                    !! FKL(.)→OWN
  HOS_ChangeUser                    ORAUSER;
  HOS_ITS_Code                      ITSCODE;
  HOS_IntegrityDate                 DATETIME;
  HOS_XST_XfrSetID    OPTIONAL      BASEID;                   !! FKL(.)→XST
CONSTRAINT
  UNIQUE
    HOS_BaseID, HOS_Version; !! PK
    HOS_GEO_CKO_Owner, HOS_GEO_CK, HOS_GEO_VRS_CODE, HOS_StandardSeq; !! UKC
END Horizontal_Segments;
```

```
TABLE Hor_Elements =
!! ( D:'Horizontal-Elemente' F:'Eléments horizontaux')
  HOE_BaseID                        BASEID;                  !! PK
  HOE_Version                       VERSION;                 !! PK
  HOE_HOS_BaseID                    BASEID      //Kernel//;  !! FKL(1)→HOS
  HOE_HOS_Version                   VERSION     //Kernel//;  !! FKL(1)→HOS
  HOE_HOS_GEO_BaseID                BASEID      //Kernel//;  !! FKL(.)→HOS
  HOE_HOS_GEO_Version               VERSION     //Kernel//;  !! FKL(.)→HOS
  HOE_HOS_GEO_CKO_Owner             OWNER       //Kernel//;  !! UKC(1)→HOS
  HOE_HOS_GEO_CK                    CK4         //Kernel//;  !! UKC(1)→HOS
  HOE_HOS_GEO_VRS_Code              VERSCODE    //Kernel//;  !! UKC(1)→HOS
  HOE_HOS_StandardSeq               NUM38       //Kernel//;  !! UKC(1)→HOS
  HOE_StandardSeq                   NUM38       //Kernel//;  !! UKC
  HOE_Elem_Code                     ELEM_CODE   //Kernel//;
  HOE_Value_1                       NUM7_4      //Kernel//;
  HOE_Value_2          OPTIONAL     NUM7_4      //Kernel//;
  HOE_Value_3          OPTIONAL     NUM7_4      //Kernel//;
  HOE_CoorX                         NUM6_3      //Kernel//;
  HOE_CoorY                         NUM6_3      //Kernel//;
  HOE_ElLength                      NUM6_3      //Kernel//;
  HOE_Comment          OPTIONAL     COMMENT     //Kernel//;
  HOE_RefDate                       DATETIME    //Kernel//;
  HOE_BeginValidity                 DATETIME    //Kernel//;
  HOE_EndValidity      OPTIONAL     DATETIME    //Kernel//;
  HOE_VRS_Code                      VERSCODE    //Kernel//;
  HOE_ODO_Owner                     OWNER;                   !! FKL(.)→OWN
  HOE_OrigSubDBID                   DBID;
  HOE_DAO_Owner                     OWNER;                   !! FKL(.)→OWN
  HOE_DataOwner                     DATAOWNER;
  HOE_CreateDate                    DATETIME;
  HOE_ChangeDate       OPTIONAL     DATETIME;
  HOE_CHO_Owner                     OWNER;                   !! FKL(.)→OWN
  HOE_ChangeUser                    ORAUSER;
  HOE_ITS_Code                      ITSCODE;
  HOE_IntegrityDate                 DATETIME;
  HOE_XST_XferSetID    OPTIONAL     BASEID;                  !! FKL(.)→XST
CONSTRAINT
  UNIQUE
    HOE_BaseID, HOE_Version; !! PK
    HOE_HOS_GEO_CKO_Owner, HOE_HOS_GEO_CK, HOE_HOS_GEO_VRS_CODE, HOE_HOS_StandardSeq;
    HOE_StandardSeq; !!UKC
END Horizontal_Elements;
```

```
TABLE Hor_Cal_Points =
!! ( D:'Horizontal-Kalibrierungspunkte' F:'Points de calage horizontaux')
  HOC_BaseID                        BASEID;                  !! PK
  HOC_Version                       VERSION;                 !! PK
  HOC_RPT_BaseID                    BASEID;                  !! FKL(1)→RPT
  HOC_AXE_CKO_Owner                 OWNER       //Kernel//;  !! UKC(1)→RPT
  HOC_AXE_CK                        CK3         //Kernel//;  !! UKC(1)→RPT
  HOC_AXE_POS_Code                  POSCODE     //Kernel//;  !! UKC(1)→RPT
  HOC_AXE_VRS_Code                  VERSCODE    //Kernel//;  !! UKC(1)→RPT
  HOC_RPT_CK                        CK2         //Kernel//;  !! UKC(1)→RPT
  HOC_RPT_VRS_Code                  VERSCODE    //Kernel//;  !! UKC(1)→RPT
  HOC_HOE_BaseID                    BASEID      //Kernel//;  !! FKL(2)→HOE
  HOC_HOE_Version                   VERSION     //Kernel//;  !! FKL(2)→HOE
  HOC_GEO_CKO_Owner                 OWNER       //Kernel//;  !! UKC(2)→HOE
  HOC_GEO_CK                        CK4         //Kernel//;  !! UKC(2)→HOE
  HOC_GEO_VRS_CODE                  VERSCODE    //Kernel//;  !! UKC(2)→HOE
  HOC_HOS_StandardSeq               NUM38       //Kernel//;  !! UKC(2)→HOE
  HOC_HOE_StandardSeq               NUM38       //Kernel//;  !! UKC(2)→HOE
  HOC_RefPointDist                  NUM1_8      //Kernel//;  !! UKC
  HOC_HoeDist                       NUM1_8      //Kernel//;  !! UKC
  HOC_Posit_Code                    POSITCODE   //Kernel//;
  HOC_Nat_Code                      NATCODE     //Kernel//;
  HOC_Cal_Factor                    NUM1_8      //Kernel//;
  HOC_Comment          OPTIONAL     COMMENT     //Kernel//;
  HOC_RefDate                       DATETIME    //Kernel//;
  HOC_BeginValidity                 DATETIME    //Kernel//;
  HOC_EndValidity      OPTIONAL     DATETIME    //Kernel//;
  HOC_Vrs_Code                      VERSCODE    //Kernel//;
  HOC_ODO_Owner                     OWNER;                   !! FKL(.)→OWN
  HOC_OrigSubDBID                   DBID;
  HOC_DAO_Owner                     OWNER;                   !! FKL(.)→OWN
  HOC_DataOwner                     DATAOWNER;
  HOC_CreateDate                    DATETIME;
  HOC_ChangeDate       OPTIONAL     DATETIME;
  HOC_CHO_Owner                     OWNER;                   !! FKL(.)→OWN
  HOC_ChangeUser                    ORAUSER;
  HOC_ITS_Code                      ITSCODE;
  HOC_IntegrityDate                 DATETIME;
  HOC_XST_XfrSetID OPTIONAL         BASEID;                  !! FKL(.)→XST
CONSTRAINT
  UNIQUE
    HOC_BaseID, HOC_Version !! PK
    HOC_AXE_CKO_OWNER, HOC_AXE_CK, HOC_AXE_POS_CODE, HOC_AXE_VRS_CODE, RPT_CK,
    HOC_RPT_VRS_CODE, HOC_GEO_CKO_OWNER, HOC_GEO_CK, HOC_GEO_VRS_CODE,
    HOC_REFPOINTDIST, HOC_HOEDIST !! UKC
END Horizontal_Calibration_Points;
```

```
    TABLE Cross_Sects =
    !! ( D:‘Geometrisches Profil‘ F:‘Profil géométrique‘, SN640942/GEN-SN640940)
      CRS_BaseID:                    BASEID;                 !! PK             SN-GEN-I1
      CRS_Version:                   VERSION;                !! PK             SN-GEN-I2
      CRS_RPT_BaseID:                BASEID      //Kernel//; !! FKL(1)->RPT    SN-I1.10
      CRS_AXE_CKO_Owner:             OWNER       //Kernel//; !! UKC(1)->RPT    SN-I1.11
      CRS_AXE_CK:                    CK3         //Kernel//; !! UKC(1)->RPT    SN-I1.12
      CRS_AXE_POS_Code:              POSCODE     //Kernel//; !! UKC(1)->RPT    SN-I1.13
      CRS_AXE_VRS_Code:              VERSCODE    //Kernel//; !! UKC(1)->RPT    SN-I1.14
      CRS_RPT_CK:                    CK2         //Kernel//; !! UKC(1)->RPT    SN-I1.15
      CRS_RPT_VRS_Code:              VERSCODE    //Kernel//; !! UKC(1)->RPT    SN-I1.16
      CRS_RefpointDist:              U_ABS       //Kernel//; !! UKC            SN-I1.21
      CRS_LatDist:                   NUM2_3;     //Kernel//; !!                SN-I1.22
      CRS_Width:                     NUM2_2      //Kernel//; !!                SN-A1
      CRS_AvgWidthLeft:   OPTIONAL   NUM2_2      //Kernel//; !!                SN-A2
      CRS_AvgWidthRigth:  OPTIONAL   NUM2_2      //Kernel//; !!                SN-A3
      CRS_PRO_BaseID:                BASEID;     //Kernel//; !! FKL(3)->PRO    SN-A4.0
      CRS_PRO_CKO_Owner:             OWNER       //Kernel//; !! FKC(3)->PRO    SN-A4.1
      CRS_PRO_CK:                    CK4         //Kernel//; !! FKC(3)->PRO    SN-A4.2
      CRS_PRO_VRS_Code:              VERSCODE    //Kernel//; !! FKC(3)->PRO    SN-A4.3
      CRS_VRS_Code:                  VERSCODE    //Kernel//; !!                SN-GEN-A1
      CRS_RefDate:                   DATEDOM     //Kernel//; !!                SN-GEN-D3
      CRS_BeginValidity:             DATEDOM     //Kernel//; !!                SN-GEN-D1
      CRS_EndValidity:    OPTIONAL   DATEDOM     //Kernel//; !!                SN-GEN-D2
      CRS_Comment:        OPTIONAL   COMMENT     //Kernel//;
      CRS_ODO_Owner:                 OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
      CRS_OrigSubDBID:               DBID;                   !!                SN-GEN-A3
      CRS_DAO_Owner:                 OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
      CRS_DataOwner:                 DATAOWNER;              !!                SN-GEN-A5
      CRS_CreateDate:                DATEDOM;                !!                SN-GEN-A8
      CRS_ChangeDate:     OPTIONAL   DATEDOM;                !!                SN-GEN-A9
      CRS_CHO_Owner:                 OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
      CRS_ChangeUser:                ORAUSER;                !!                SN-GEN-A11
      CRS_ITS_Code:                  ITSCODE;                !!                SN-GEN-A6
      CRS_IntegrityDate:             DATEDOM;                !!                SN-GEN-A7
      CRS_XST_XfrSetID:   OPTIONAL   BASEID;                    !!             FKL(.)->XST
    CONSTRAINT
      UNIQUE
CRS_BaseID, CRS_Version; !! PK
CRS_AXE_CKO_Owner, CRS_AXE_CK, CRS_AXE_POS_Code, CRS_AXE_VRS_Code, CRS_RPT_CK, CRS_RPT_VRS_Code,
CRS_RefpointDist; !! UKC
    END Cross_Section;
```

```
    TABLE Cr_S_Usages =
    !! ( D:‘Fahrbahn-Nutzung‘ F:‘Usage de la chaussée‘, SN640942/GEN-SN640940)
      CSU_BaseID:                       BASEID                  !! PK             SN-GEN-I1
      CSU_Version:                      VERSION                 !! PK             SN-GEN-I2
      CSU_AXE_CKO_Owner:                OWNER       //Kernel//; !! UKC(1,2)->RPT  SN-I1.1
      CSU_AXE_CK:                       CK3         //Kernel//; !! UKC(1,2)->RPT  SN-I1.2
      CSU_AXE_POS_Code:                 POSCODE     //Kernel//; !! UKC(1,2)->RPT  SN-I1.3
      CSU_AXE_VRS_Code:                 VERSCODE    //Kernel//; !! UKC(1,2)->RPT  SN-I1.4
      CSU_RPT_BaseID_1:                 BASEID      //Kernel//; !! FKL(1)->RPT    SN-I1.10
      CSU_RPT_CK_1:                     CK2         //Kernel//; !! UKC(1)->RPT    SN-I1.11
      CSU_RPT_VRS_Code_1:               VERSCODE    //Kernel//; !! UKC(1)->RPT    SN-I1.12
      CSU_RPDistBegin:                  U_ABS       //Kernel//; !! UKC            SN-I1.21
      CSU_RPT_BaseID_2:                 BASEID      //Kernel//; !! FKL(2)->RPT    SN-I2.0
      CSU_RPT_CK_2:                     CK2         //Kernel//; !! UKC(2)->RPT    SN-I2.1
      CSU_RPT_VRS_Code_2:               VERSCODE    //Kernel//; !! UKC(2)->RPT    SN-I2.2
      CSU_RPDistEnd:                    U_ABS       //Kernel//; !! UKC            SN-I2.11
      CSU_PosBegin:                     NUM2_2      //Kernel//; !! UKC            SN-I3
      CSU_PosEnd:                       NUM2_2      //Kernel//; !! UKC            SN-I4
      CSU_POS_Code:                     POSCODE     //Kernel//; !!                SN-A1
      CSU_Width:                        NUM2_2      //Kernel//; !!                SN-A2
      CSU_WidthBegin:                   NUM2_2      //Kernel//;
      CSU_WidthEnd:                     NUM2_2      //Kernel//;
      CSU_SCA_UST_CTK_BaseID:           BASEID      //Kernel//; !! FKL(3)->SCA    SN-A3.0
      CSU_SCA_UST_CTK_CKO_Owner:        OWNER       //Kernel//; !! FKC(3)->SCA    SN-A3.1
      CSU_SCA_UST_CTT_LNG_Code:         LANGCODE    //Kernel//; !! FKC(3)->SCA    SN-A3.2
      CSU_SCA_UST_CTT_TextID:           TEXTID      //Kernel//; !! FKC(3)->SCA    SN-A3.3
      CSU_TypAdText:       OPTIONAL     ADTEXT      //Kernel//; !!                SN-A4
      CSU_PRO_BaseID:                   BASEID      //Kernel//; !! FKL(5)->PRO    SN-A5.0
      CSU_PRO_CKO_Owner:                OWNER       //Kernel//; !! FKC(5)->PRO    SN-A5.1
      CSU_PRO_CK:                       CK4         //Kernel//; !! FKC(5)->PRO    SN-A5.2
      CSU_PRO_VRS_Code:                 VERSCODE    //Kernel//; !! FKC(5)->PRO    SN-A5.3
      CSU_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
      CSU_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
      CSU_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
      CSU_EndValidity:     OPTIONAL     DATEDOM     //Kernel//; !!                SN-GEN-D2
      CSU_Comment:         OPTIONAL     COMMENT     //Kernel//;
      CSU_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
      CSU_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
      CSU_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
      CSU_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
      CSU_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
      CSU_ChangeDate:      OPTIONAL     DATEDOM;                !!                SN-GEN-A9
      CSU_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
      CSU_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
      CSU_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
      CSU_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
      CSU_XST_XfrSetID:    OPTIONAL     BASEID;                 !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
CSU_BaseID, CSU_Version; !! PK
CSU_AXE_CKO_Owner, CSU_AXE_CK, CSU_AXE_POS_Code, CSU_AXE_VRS_Code,
CSU_RPT_CK_1, CSU_RPT_VRS_Code_1, CSU_RPDistBegin,
CSU_RPT_CK_2, CSU_RPT_VRS_Code_2, CSU_RPDistEnd, CSU_PosBegin, CSU_PosEnd; !! UKC
    END Cross_Sect_Usage;
```

```
TABLE Lat_Lanes =
!! (D:‘Nebenstreifen‘ F:‘Parties latérales‘, SN640942/GEN-SN640940)
  LAL_BaseID:                       BASEID                  !! PK              SN-GEN-I1
  LAL_Version:                      VERSION                 !! PK              SN-GEN-I2
  LAL_AXE_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1,2)->RPT  SN-I1.1
  LAL_AXE_CK:                       CK3         //Kernel//;  !! UKC(1,2)->RPT  SN-I1.2
  LAL_AXE_POS_Code:                 POSCODE     //Kernel//;  !! UKC(1,2)->RPT  SN-I1.3
  LAL_AXE_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1,2)->RPT  SN-I1.4
  LAL_RPT_BaseID_1:                 BASEID      //Kernel//;  !! FKL(1)->RPT    SN-I1.10
  LAL_RPT_CK_1:                     CK2         //Kernel//;  !! UKC(1)->RPT    SN-I1.11
  LAL_RPT_VRS_Code_1:               VERSCODE    //Kernel//;  !! UKC(1)->RPT    SN-I1.12
  LAL_RPDistBegin:                  U_ABS       //Kernel//;  !! UKC            SN-I1.21
  LAL_RPT_BaseID_2:                 BASEID      //Kernel//;  !! FKL(2)->RPT    SN-I2.0
  LAL_RPT_CK_2:                     CK2         //Kernel//;  !! UKC(2)->RPT    SN-I2.1
  LAL_RPT_VRS_Code_2:               VERSCODE    //Kernel//;  !! UKC(2)->RPT    SN-I2.2
  LAL_RPDistEnd:                    U_ABS       //Kernel//;  !! UKC            SN-I2.11
  LAL_PosBegin:                     NUM2_2      //Kernel//;  !! UKC            SN-I3
  LAL_PosEnd:                       NUM2_2      //Kernel//;  !! UKC            SN-I4
  LAL_POS_Code:                     POSCODE     //Kernel//;  !!                SN-A1
  LAL_Width:                        NUM2_2      //Kernel//;  !!                SN-A2
  LAL_WidthBegin:                   NUM2_2      //Kernel//;
  LAL_WidthEnd:                     NUM2_2      //Kernel//;
  LAL_SCA_UST_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(4)->SCA    SN-A3.0
  LAL_SCA_UST_CTK_CKO_Owner:        OWNER       //Kernel//;  !! FKC(4)->SCA    SN-A3.1
  LAL_SCA_UST_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! FKC(4)->SCA    SN-A3.2
  LAL_SCA_UST_CTT_TextID:           TEXTID      //Kernel//;  !! FKC(4)->SCA    SN-A3.3
  LAL_UST_TypAdText:   OPTIONAL     ADTEXT      //Kernel//;  !!                  SN-A5
  LAL_SCA_PLT_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(5)->SCA    SN-A4.0
  LAL_SCA_PLT_CTK_CKO_Owner:        OWNER       //Kernel//;  !! FKC(5)->SCA    SN-A4.1
  LAL_SCA_PLT_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! FKC(5)->SCA    SN-A4.2
  LAL_SCA_PLT_CTT_TextID:           TEXTID      //Kernel//;  !! FKC(5)->SCA    SN-A4.3
  LAL_PLT_TypAdText    OPTIONAL     ADTEXT      //Kernel//;  !!                  SN-A6
  LAL_PRO_BaseID:                   BASEID      //Kernel//;  !! FKL(5)->PRO    SN-A7.0
  LAL_PRO_CKO_Owner:                OWNER       //Kernel//;  !! FKC(5)->PRO    SN-A7.1
  LAL_PRO_CK:                       CK4         //Kernel//;  !! FKC(5)->PRO    SN-A7.2
  LAL_PRO_VRS_Code:                 VERSCODE    //Kernel//;  !! FKC(5)->PRO    SN-A7.3
  LAL_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  LAL_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  LAL_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  LAL_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
  LAL_Comment:         OPTIONAL     COMMENT     //Kernel//;
  LAL_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
  LAL_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
  LAL_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
  LAL_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
  LAL_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
  LAL_ChangeDate:      OPTIONAL     DATEDOM;                !!                SN-GEN-A9
  LAL_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
  LAL_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
  LAL_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
  LAL_IntegrityDate:                DATEDOM;                   !!             SN-GEN-A7
  LAL_XST_XfrSetID:    OPTIONAL     BASEID;                    !!             FKL(.)->XST
CONSTRAINT
  UNIQUE
    LAL_BaseID, LAL_Version; !! PK
    LAL_AXE_CKO_Owner, LAL_AXE_CK, LAL_AXE_POS_Code, LAL_AXE_VRS_Code,
    LAL_RPT_CK_1, LAL_RPT_VRS_Code_1, LAL_RPDistBegin,
    LAL_RPT_CK_2, LAL_RPT_VRS_Code_2, LAL_RPDistEnd, LAL_PosBegin, LAL_PosEnd; !! UKC
END Lateral_Lane;
```

```
TABLE Pave_Layers =
!! ( D:‘ Belagsschichten‘ F:‘Structure de la chaussée‘, SN640943/GEN-SN640940)
  PVL_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
  PVL_Version:                      VERSION;                 !! PK             SN-GEN-I2
  PVL_PRO_BaseID:                   BASEID      //Kernel//;  !! FKL(1)->PRO    SN-I7.0
  PVL_PRO_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1)->PRO    SN-I7.1
  PVL_PRO_CK:                       CK4         //Kernel//;  !! UKC(1)->PRO    SN-I7.2
  PVL_PRO_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1)->PRO    SN-I7.3
  PVL_SCA_PLT_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(2)->SCA    SN-I5.0
  PVL_SCA_PLT_CTK_CKO_Owner:        OWNER       //Kernel//;  !! UKC(2)->SCA    SN-I5.1
  PVL_SCA_PLT_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! UKC(2)->SCA    SN-I5.2
  PVL_SCA_PLT_CTT_TextID:           TEXTID      //Kernel//;  !! UKC(2)->SCA    SN-I5.3
  PVL_TypAdText:       OPTIONAL     ADTEXT      //Kernel//;  !!                SN-A6
  PVL_ThickMounted:                 NUM1_3      //Kernel//;  !!                SN-A3
  PVL_ThickRemoved:                 NUM1_3      //Kernel//;  !!                SN-A4
  PVL_ThickUsed:                    NUM1_3      //Kernel//;  !!                SN-A5
  PVL_Cst_Code:        OPTIONAL     COSTCODE    //Kernel//;  !!                SN-A10
  PVL_CostAccount:     OPTIONAL     NUM2        //Kernel//;  !!                SN-A11
  PVL_CostSubAccount: OPTIONAL      NUM2        //Kernel//;  !!                SN-A12
  PVL_Costs:           OPTIONAL     NUM9_2      //Kernel//;  !!                SN-A8
  PVL_OwnerAccount:    OPTIONAL     TEXT*16     //Kernel//;  !!                SN-A9
  PVL_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  PVL_RefDate_A:                    DATEDOM     //Kernel//;  !!                SN-GEN-D3
  PVL_BeginValidity_A: DATEDOM      //Kernel//; !!            UKC               SN-I3
  PVL_EndValidity_A:                DATEDOM     //Kernel//;  !!                SN-I4
  PVL_Comment:         OPTIONAL     COMMENT     //Kernel//;
  PVL_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  PVL_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  PVL_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  PVL_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  PVL_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  PVL_ChangeDate:      OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
  PVL_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  PVL_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  PVL_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  PVL_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  PVL_XST_XfrSetID:    OPTIONAL     BASEID;                  !! FKL(.)->XST
  PVL_PLO_BaseID:                   BASEID;     //Kernel//;  !! FKL(.)->PLO
  PVL_PLO_AXE_CKO_Owner:            OWNER       //Kernel//;  !! UKC(3,4)->RPT
  PVL_PLO_AXE_CK:                   CK3         //Kernel//;  !! UKC(3,4)->RPT
  PVL_PLO_AXE_POS_Code:             POSCODE     //Kernel//;  !! UKC(3,4)->RPT
  PVL_PLO_AXE_VRS_Code:             VERSCODE    //Kernel//;  !! UKC(3,4)->RPT
  PVL_PLO_Sequence:                 NUM3        //Kernel//;  !! UKC
  PVL_PLO_RPT_BaseID1:              BASEID      //Kernel//;  !! FKL(3)->RPT
  PVL_PLO_RPT_CK:                   CK2         //Kernel//;  !! FKC(3)->RPT
  PVL_PLO_RPT_VRS_Code:             VERSCODE    //Kernel//;  !! FKC(3)->RPT
  PVL_PLO_RPDistBegin:              U_ABS       //Kernel//;
  PVL_PLO_LatDistBegin:             NUM2_3      //Kernel//;
  PVL_PLO_RPT_BaseID_2:             BASEID      //Kernel//;  !! FKL(4)->RPT
  PVL_PLO_RPT_CK_2:                 CK2         //Kernel//;  !! FKC(4)->RPT
  PVL_PLO_RPT_VRS_Code_2:           VERSCODE    //Kernel//;  !! FKC(4)->RPT
  PVL_PLO_RPDistEnd:                U_ABS       //Kernel//;
  PVL_PLO_LatDistEnd:               NUM2_3      //Kernel//;
  PVL_PLO_WidthBegin:               NUM2_2      //Kernel//;
  PVL_PLO_WidthEnd:                 NUM2_2      //Kernel//;
  PVL_PLO_CreateDate:               DATEDOM;                 !!                SN-GEN-A8
  PVL_PLO_ITS_Code:                 ITSCODE;                 !!                SN-GEN-A6
  PVL_PLO_IntegrityDate:            DATEDOM;                 !!                SN-GEN-A7
  PVL_PLO_XST_XfrSetID: OPTIONAL    BASEID;                  !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    PVL_BaseID, PVL_Version; !! PK
    PVL_PLO_AXE_CKO_Owner, PVL_PLO_AXE_CK, PVL_PLO_AXE_POS_Code, PVL_PLO_AXE_VRS_Code,
    PVL_PRO_CKO_Owner, PVL_PRO_CK, PVL_PRO_VRS_Code,
    PVL_SCA_PLT_CTK_CKO_Owner, PVL_SCA_PLT_CTT_LNG_Code, PVL_SCA_PLT_CTT_TextID,
    PVL_PLO_Sequence, PVL_BeginValidity_A; !! UKC
END Pavement_Layer;
```

```
TABLE Road_Repairs =
!! ( D:‘Belagsreparaturen‘ F:‘Réparations de la chaussée‘, GEN-SN640940)
  RRP_BaseID:                     BASEID                  !! PK             SN-GEN-I1
  RRP_Version:                    VERSION                 !! PK             SN-GEN-I2
  RRP_AXE_CKO_Owner:              OWNER       //Kernel//;  !! UKC(1,2)->RPT
  RRP_AXE_CK:                     CK3         //Kernel//;  !! UKC(1,2)->RPT
  RRP_AXE_POS_Code:               POSCODE     //Kernel//;  !! UKC(1,2)->RPT
  RRP_AXE_VRS_Code:               VERSCODE    //Kernel//;  !! UKC(1,2)->RPT
  RRP_RPT_BaseID_1:               BASEID      //Kernel//;  !! FKL(1)->RPT
  RRP_RPT_CK_1:                   CK2         //Kernel//;  !! UKC(1)->RPT
  RRP_RPT_VRS_Code_1:             VERSCODE    //Kernel//;  !! UKC(1)->RPT
  RRP_RPDistBegin:                U_ABS       //Kernel//;  !! UKC
  RRP_LatDistBegin:               NUM2_3      //Kernel//;  !! UKC
  RRP_RPT_BaseID_2:               BASEID      //Kernel//;  !! FKL(2)->RPT
  RRP_RPT_CK_2:                   CK2         //Kernel//;  !! UKC(2)->RPT
  RRP_RPT_VRS_Code_2:             VERSCODE    //Kernel//;  !! UKC(2)->RPT
  RRP_RPDistEnd:                  U_ABS       //Kernel//;  !! UKC
  RRP_LatDistEnd:                 NUM2_3      //Kernel//;  !! UKC
  RRP_PRO_BaseID:                 BASEID      //Kernel//;  !! UKL(4)->PRO
  RRP_PRO_CKO_Owner:              OWNER       //Kernel//;  !! UKC(4)->PRO
  RRP_PRO_CK:                     CK4         //Kernel//;  !! UKC(4)->PRO
  RRP_PRO_VRS_Code:               VERSCODE    //Kernel//;  !! UKC(4)->PRO
  RRP_SCA_RRT_CTK_BaseID:         BASEID      //Kernel//;  !! FKL(5)->SCA
  RRP_SCA_RRT_CTK_CKO_Owner:      OWNER       //Kernel//;  !! FKC(5)->SCA
  RRP_SCA_RRT_CTT_LNG_Code:       LANGCODE    //Kernel//;  !! FKC(5)->SCA
  RRP_SCA_RRT_CTT_TextID:         TEXTID      //Kernel//;  !! FKC(5)->SCA
  RRP_TypAdText:       OPTIONAL   ADTEXT      //Kernel//;
  RRP_WidthBegin:                 NUM2_2      //Kernel//;
  RRP_WidthEnd:                   NUM2_2      //Kernel//;
  RRP_Depth:           NUM1_3     //Kernel//;
  RRP_Cst_Code:        OPTIONAL   COSTCODE    //Kernel//;
  RRP_CostAccount:     OPTIONAL   NUM2        //Kernel//;
  RRP_CostSubAccount:  OPTIONAL   NUM2        //Kernel//;
  RRP_Costs:           OPTIONAL   NUM9_2      //Kernel//;
  RRP_OwnerAccount:    OPTIONAL   TEXT*16     //Kernel//;
  RRP_VRS_Code:                   VERSCODE    //Kernel//;  !!                SN-GEN-A1
  RRP_RefDate_A:                  DATEDOM     //Kernel//;  !!                SN-GEN-D3
  RRP_BeginValidity_A:            DATEDOM     //Kernel//;  !! UKC            SN-GEN-D1
  RRP_EndValidity_A:   OPTIONAL   DATEDOM     //Kernel//;  !!                SN-GEN-D2
  RRP_Comment:         OPTIONAL   COMMENT;
  RRP_ODO_Owner:                  OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
  RRP_OrigSubDBID:                DBID;                   !!                SN-GEN-A3
  RRP_DAO_Owner:                  OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
  RRP_DataOwner:                  DATAOWNER;              !!                SN-GEN-A5
  RRP_CreateDate:                 DATEDOM;                !!                SN-GEN-A8
  RRP_ChangeDate:      OPTIONAL   DATEDOM;                !!                SN-GEN-A9
  RRP_CHO_Owner:                  OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
  RRP_ChangeUser:                 ORAUSER;                !!                SN-GEN-A11
  RRP_ITS_Code:                   ITSCODE;                !!                SN-GEN-A6
  RRP_IntegrityDate:              DATEDOM;                !!                SN-GEN-A7
  RRP_XST_XfrSetID:    OPTIONAL   BASEID;                 !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    RRP_BaseID, RRP_Version; !! PK
    RRP_AXE_CKO_Owner, RRP_AXE_CK, RRP_AXE_POS_Code, RRP_AXE_VRS_Code,
    RRP_RPT_CK_1, RRP_RPT_VRS_Code_1, RRP_RPDistBegin, RRP_LatDistBegin,
    RRP_RPT_CK_2, RRP_RPT_VRS_Code_2, RRP_RPDistEnd, RRP_LatDistEnd,
    RRP_PRO_CKO_Owner, RRP_PRO_CK, RRP_PRO_VRS_Code, RRP_BeginValidity_A;!! UKC
END Road_Repair;
```

```
TABLE RS_Classes =
!! ( D:‘Bewertungsregeln‘ F:‘Règles d‘évaluation‘, GEN-SN640940)
  RCL_BaseID:                          BASEID;                  !! PK             SN-GEN-I1
  RCL_Version:                         VERSION;                 !! PK             SN-GEN-I2
  RCL_CKO_Owner:                       OWNER       //Kernel//;  !! UKC(.)->OWN
  RCL_IndexFunction:                   TEXT*8      //Kernel//;  !! UKC
  RCL_CK:                              NUM2        //Kernel//;  !! UKC
  RCL_SCA_CLT_CTK_BaseID:              BASEID      //Kernel//;  !! FKL(1)->SCA
  RCL_SCA_CLT_CTK_CKO_Owner:           OWNER       //Kernel//;  !! FKC(1)->SCA
  RCL_SCA_CLT_CTT_LNG_Code:            LANGCODE    //Kernel//;  !! FKC(1)->SCA
  RCL_SCA_CLT_CTT_TextID:              TEXTID      //Kernel//;  !! FKC(1)->SCA
  RCL_TypAdText:       OPTIONAL        ADTEXT      //Kernel//;
  RCL_Value1:                          NUM4_4      //Kernel//;
  RCL_Value2:                          NUM4_4      //Kernel//;
  RCL_Name:                            NAME        //Kernel//;
  RCL_VRS_Code:                        VERSCODE    //Kernel//;  !!                SN-GEN-A1
  RCL_RefDate:                         DATEDOM     //Kernel//;  !!                SN-GEN-D3
  RCL_BeginValidity:                   DATEDOM     //Kernel//;  !!                SN-GEN-D1
  RCL_EndValidity:     OPTIONAL        DATEDOM     //Kernel//;  !!                SN-GEN-D2
  RCL_Comment:         OPTIONAL        COMMENT     //Kernel//;
  RCL_ODO_Owner:                       OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  RCL_OrigSubDBID:                     DBID;                    !!                SN-GEN-A3
  RCL_DAO_Owner:                       OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  RCL_DataOwner:                       DATAOWNER;               !!                SN-GEN-A5
  RCL_CreateDate:                      DATEDOM;                 !!                SN-GEN-A8
  RCL_ChangeDate:      OPTIONAL        DATEDOM;                 !!                SN-GEN-A9
  RCL_CHO_Owner:                       OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  RCL_ChangeUser:                      ORAUSER;                 !!                SN-GEN-A11
  RCL_ITS_Code:                        ITSCODE;                 !!                SN-GEN-A6
  RCL_IntegrityDate:                   DATEDOM;                 !!                SN-GEN-A7
  RCL_XST_XfrSetID:    OPTIONAL        BASEID;                  !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    RCL_BaseID, RCL_Version; !! PK
    RCL_CKO_Owner, RCL_IndexFunction, RCL_CK!! UKC
END Roadsurface_Class;
```

```
TABLE RS_Controls =
!! ( D:‘Zustandskontrollen‘ F:‘Contrôle de la surface de la chaussée‘, GEN-SN640940)
  RCO_BaseID:                       BASEID;                  !! PK             SN-GEN-I1
  RCO_Version:                      VERSION;                 !! PK             SN-GEN-I2
  RCO_AXE_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1,2)->RPT
  RCO_AXE_CK:                       CK3         //Kernel//;  !! UKC(1,2)->RPT
  RCO_AXE_POS_Code:                 POSCODE     //Kernel//;  !! UKC(1,2)->RPT
  RCO_AXE_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1,2)->RPT
  RCO_RPT_BaseID_1:                 BASEID      //Kernel//;  !! FKL(1)->RPT
  RCO_RPT_CK_1:                     CK2         //Kernel//;  !! UKC(1)->RPT
  RCO_RPT_VRS_Code_1:               VERSCODE    //Kernel//;  !! UKC(1)->RPT
  RCO_RPDistBegin:                  U_ABS       //Kernel//;  !! UKC
  RCO_LatDistBegin:                 NUM2_3      //Kernel//;  !! UKC
  RCO_RPT_BaseID_2:                 BASEID      //Kernel//;  !! FKL(2)->RPT
  RCO_RPT_CK_2:                     CK2         //Kernel//;  !! UKC(2)->RPT
  RCO_RPT_RefDate_2:                VERSCODE    //Kernel//;  !! UKC(2)->RPT
  RCO_RPDistEnd:                    U_ABS       //Kernel//;  !! UKC
  RCO_LatDistEnd:                   NUM2_3      //Kernel//;  !! UKC
  RCO_PRO_BaseID:                   BASEID      //Kernel//;  !! FKL(4)->PRO
  RCO_PRO_CKO_Owner:                OWNER       //Kernel//;  !! FKC(4)->PRO
  RCO_PRO_CK:                       CK4         //Kernel//;  !! FKC(4)->PRO
  RCO_PRO_VRS_Code:                 VERSCODE    //Kernel//;  !! FKC(4)->PRO
  RCO_SCA_CLT_CTK_BaseID:           BASEID      //Kernel//;  !! FKL(5)->SCA
  RCO_SCA_CLT_CTK_CKO_Owner:        OWNER       //Kernel//;  !! UKC(5)->SCA
  RCO_SCA_CLT_CTT_LNG_Code:         LANGCODE    //Kernel//;  !! UKC(5)->SCA
  RCO_SCA_CLT_CTT_TextID:           TEXTID      //Kernel//;  !! UKC(5)->SCA
  RCO_TypAdText:       OPTIONAL     ADTEXT      //Kernel//;
  RCO_MeasurIntrval:                NUM3_2      //Kernel//;
  RCO_Value1:                       NUM4_4      //Kernel//;
  RCO_Value2:                       NUM4_4      //Kernel//;
  RCO_Value3:          OPTIONAL     NUM4_4      //Kernel//;
  RCO_Speed:           OPTIONAL     NUM3        //Kernel//;
  RCO_Width:           OPTIONAL     NUM2_2      //Kernel//;
  RCO_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  RCO_RefDate_A:                    DATEDOM     //Kernel//;  !!                SN-GEN-D3
  RCO_BeginValidity_A:              DATEDOM     //Kernel//;  !! UKC            SN-GEN-D1
  RCO_EndValidity_A:   OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
  RCO_Comment:         OPTIONAL     COMMENT     //Kernel//;
  RCO_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  RCO_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  RCO_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  RCO_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  RCO_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  RCO_ChangeDate:      OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
  RCO_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  RCO_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  RCO_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  RCO_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  RCO_XST_XfrSetID:    OPTIONAL     BASEID;                  !! FKL(.)->XST
  RCO_Data_Exchange:   OPTIONAL     OUINON;
CONSTRAINT
  UNIQUE
    RCO_BaseID,RCO_Version;
    RCO_AXE_CKO_Owner, RCO_AXE_CK, RCO_AXE_POS_Code, RCO_AXE_VRS_Code,
    RCO_RPT_CK_1, RCO_RPT_VRS_Code_1, RCO_RPDistBegin, RCO_LatDistBegin,
    RCO_RPT_CK_2, RCO_RPT_VRS_Code_2, RCO_RPDistEnd, RCO_LatDistEnd,
    RCO_SCA_CLT_CAT_CKO_Owner, RCO_SCA_CLT_CAT_CK, RCO_SCA_CLT_CTK_CKO_Owner,
    RCO_SCA_CLT_CTT_LNG_Code, RCO_SCA_CLT_CTT_TextID, RCO_BeginValidity_A; !! UKC
END Roadsurface_Control;
```

```
TABLE Measure_Units =
!! (D: 'Masseinheiten', F: 'Unités de mesure', SN-Norm Trafic Basic Data, GEN-SN640940)
  MEU_BASEID:                       BASEID;                 !! PK             SN-GEN-I1
  MEU_VERSION:                      VERSION;                !! PK             SN-GEN-I2
  MEU_CKO_OWNER:                    CKO         //Kernel//; !! UKC(.)->OWN    SN-I1.1
  MEU_CK:                           CK          //Kernel//; !! UKC            SN-I1.2
  MEU_NAME:           OPTIONAL      NAME        //Kernel//; !!
  MEU_STRANDARDSEQ:                 NUM5        //Kernel//;
  MEU_EXPMETER:                     NUM2        //Kernel//;
  MEU_EXPSECOND:                    NUM2        //Kernel//;
  MEU_EXPKILO:                      NUM2        //Kernel//;
  MEU_EXPAMPERE:                    NUM2        //Kernel//;
  MEU_EXPKELVIN:                    NUM2        //Kernel//;
  MEU_EXPMOL:                       NUM2        //Kernel//;
  MEU_EXPCANDELA:                   NUM2        //Kernel//;
  MEU_EXPVEHICLE:                   NUM2        //Kernel//;
  MEU_EXPDEZIBEL:                   NUM2        //Kernel//;
  MEU_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
  MEU_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
  MEU_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
  MEU_EndValidity:    OPTIONAL      DATEDOM     //Kernel//; !!                SN-GEN-D2
  MEU_Comment:        OPTIONAL      COMMENT;
  MEU_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
  MEU_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
  MEU_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
  MEU_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
  MEU_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
  MEU_ChangeDate:     OPTIONAL      DATEDOM;                !!                SN-GEN-A9
  MEU_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
  MEU_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
  MEU_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
  MEU_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
  MEU_XST_XferSetID:  OPTIONAL      BASEID;                 !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    MEU_BASEID, MEU_VERSION; !! PK
    MEU_CKO_OWNER, MEU_CK; !! UKC
END Measure_Units;
```

```
TABLE Scalar_Lists;
  SCL_BASEID:                    BASEID;                !! PK             SN-GEN-I1
  SCL_VERSION:                   VERSION;               !! PK             SN-GEN-I2
  SCL_CKO_OWNER:                 CKO         //Kernel//;  !! UKC(.)->OWN
  SCL_CK:                        CK5         //Kernel//;  !! UKC
  SCL_NAME:           OPTIONAL   NAME        //Kernel//;
  SCL_MEU_BASEID:                BASEID                 !! FKL(1)->MEU
  SCL_MEU_VERSION:               VERSION                !! FKL(1)->MEU
  SCL_MEU_CKO_OWNER:             CKO         //Kernel//  !! FKC(1)->MEU
  SCL_MEU_CK:                    CK          //Kernel//  !! FKC(1)->MEU
  SCL_DIVIDEND:                  NUM         //Kernel//;
  SCL_DIVISOR:                   NUM         //Kernel//;
  SCL_VRS_Code:                  VERSCODE    //Kernel//;  !!               SN-GEN-A1
  SCL_RefDate:                   DATEDOM     //Kernel//;  !!               SN-GEN-D3
  SCL_BeginValidity:             DATEDOM     //Kernel//;  !!               SN-GEN-D1
  SCL_EndValidity:    OPTIONAL   DATEDOM     //Kernel//;  !!               SN-GEN-D2
  SCL_Comment:        OPTIONAL   COMMENT;
  SCL_ODO_Owner:                 OWNER;                 !! FKL(.)->OWN    SN-GEN-A2
  SCL_OrigSubDBID:               DBID;                  !!                SN-GEN-A3
  SCL_DAO_Owner:                 OWNER;                 !! FKL(.)->OWN    SN-GEN-A4
  SCL_DataOwner:                 DATAOWNER;             !!                SN-GEN-A5
  SCL_CreateDate:                DATEDOM;               !!                SN-GEN-A8
  SCL_ChangeDate:     OPTIONAL   DATEDOM;               !!                SN-GEN-A9
  SCL_CHO_Owner:                 OWNER;                 !! FKL(.)->OWN    SN-GEN-A10
  SCL_ChangeUser:                ORAUSER;               !!                SN-GEN-A11
  SCL_ITS_Code:                  ITSCODE;               !!                SN-GEN-A6
  SCL_IntegrityDate:             DATEDOM;               !!                SN-GEN-A7
  SCL_XST_XferSetID:  OPTIONAL   BASEID;                !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    SCL_BASEID, SCL_VERSION; !! PK
    SCL_CKO_OWNER, SCL_CK; !! UKC
END Scalar_Lists;
```

```
TABLE Variation_Function_Lists
  TFL_BASEID:                       BASEID;                 !! PK             SN-GEN-I1
  TFL_VERSION:                      VERSION;                !! PK             SN-GEN-I2
  VFL_CKO-OWNER:                    CKO         //Kernel//; !! UKC(.)->OWN
  VFL_CK:                           CK          //Kernel//; !! UKC
  VFL_NAME:           OPTIONAL      NAME        //Kernel//;
  VFL_ASPECT_CODE:                  ASPCODE     //Kernel//;
  VFL_SCL_BASEID1:                  BASEID;                 !! FKL(1)->SCL
  VFL_SCL_VERSION1:                 VERSION;                !! FKL(1)->SCL
  VFL_SCL_CKO_OWNER1:               CKO         //Kernel//; !! FKC(1)->SCL
  VFL_SCL_CK1:                      CK          //Kernel//; !! FKC(1)->SCL
  VFL_SCL_BASEID2:                  BASEID;                 !! FKL(2)->SCL
  VFL_SCL_VERSION2:                 VERSION;                !! FKL(2)->SCL
  VFL_SCL_CKO_OWNER2:               CKO         //Kernel//; !! FKC(2)->SCL
  VFL_SCL_CK2:                      CK          //Kernel//; !! FKC(2)->SCL
  VFL_SCL_BASEID3:                  BASEID;                 !! FKL(3)->SCL
  VFL_SCL_VERSION3:                 VERSION;                !! FKL(3)->SCL
  VFL_SCL_CKO_OWNER3:               CKO         //Kernel//; !! FKC(3)->SCL
  VFL_SCL_CK3:                      CK          //Kernel//; !! FKC(3)->SCL
  VFL_ELEMENTS:                     NUM6        //Kernel//;
  VFL_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
  VFL_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
  VFL_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
  VFL_EndValidity:    OPTIONAL      DATEDOM     //Kernel//; !!                SN-GEN-D2
  VFL_Comment:        OPTIONAL      COMMENT;
  VFL_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
  VFL_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
  VFL_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
  VFL_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
  VFL_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
  VFL_ChangeDate:     OPTIONAL      DATEDOM;                !!                SN-GEN-A9
  VFL_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
  VFL_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
  VFL_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
  VFL_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
  VFL_XST_XferSetID:  OPTIONAL      BASEID;                 !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    VFL_BASEID, VFL_VERSION; !! PK
    VFL_CKO_OWNER, VFL_CK; !! UKC
END Variation_Function_List;
```

```
TABLE Episode =
  EPI_BASEID:                    BASEID;                !! PK             SN-GEN-I1
  EPI_VERSION:                   VERSION;               !! PK             SN-GEN-I2
  EPI_CKO_OWNER:                 CKO         //Kernel//;  !! UKC(.)->OWN
  EPI_CK:                        CK5         //Kernel//;  !! UKC
  EPI_KIND:                      KIND        //Kernel//;
  EPI_NAME:           OPTIONAL   NAME        //Kernel//;
  EPI_STANDARDSEQ:               NUM5        //Kernel//;
  EPI_VRS_Code:                  VERSCODE    //Kernel//;  !!               SN-GEN-A1
  EPI_RefDate:                   DATEDOM     //Kernel//;  !!               SN-GEN-D3
  EPI_BeginValidity:             DATEDOM     //Kernel//;  !!               SN-GEN-D1
  EPI_EndValidity:    OPTIONAL   DATEDOM     //Kernel//;  !!               SN-GEN-D2
  EPI_Comment:        OPTIONAL   COMMENT;
  EPI_ODO_Owner:                 OWNER;                 !! FKL(.)->OWN    SN-GEN-A2
  EPI_OrigSubDBID:               DBID;                  !!                SN-GEN-A3
  EPI_DAO_Owner:                 OWNER;                 !! FKL(.)->OWN    SN-GEN-A4
  EPI_DataOwner:                 DATAOWNER;             !!                SN-GEN-A5
  EPI_CreateDate:                DATEDOM;               !!                SN-GEN-A8
  EPI_ChangeDate:     OPTIONAL   DATEDOM;               !!                SN-GEN-A9
  EPI_CHO_Owner:                 OWNER;                 !! FKL(.)->OWN    SN-GEN-A10
  EPI_ChangeUser:                ORAUSER;               !!                SN-GEN-A11
  EPI_ITS_Code:                  ITSCODE;               !!                SN-GEN-A6
  EPI_IntegrityDate:             DATEDOM;               !!                SN-GEN-A7
  EPI_XST_XferSetID:  OPTIONAL   BASEID;                !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    EPI_BASEID, EPI_VERSION; !! PK
    EPI_CKO_OWNER, EPI_CK; !! UKC (+ EPI_STRANDARDSEQ)
END Episode;
```

```
TABLE Scenarios =
  SCE_BASEID:                       BASEID;                  !! PK             SN-GEN-I1
  SCE_VERSION:                      VERSION;                 !! PK             SN-GEN-I2
  SCE_CKO_OWNER:                    CKO         //Kernel//;  !! UKC(.)->OWN
  SCE_CK:                           CK          //Kernel//;  !! UKC
  SCE_STANDARDSEQ:                  NUM5        //Kernel//;
  SCE_NAME:           OPTIONAL      NAME        //Kernel//;
  SCE_SCA_BASEID:                   BASEID      //KERNEL//;  !! FKL(1)->SCA
  SCE_SCA_CTK_CKO-Owner:            OWNER       //Kernel//;  !! FKC(1)->SCA
  SCE_SCA_CTT_TextID:               TEXTID      //Kernel//;  !! FKC(1)->SCA
  SCE_SCA_CTT_Lng_code:             LANGCODE    //Kernel//;  !! FKC(1)->SCA
  SCE_SCA_CTK_RefDate:              DATEDOM     //Kernel//;  !! FKC(1)->SCA
  SCE_TYPADTEXT:      OPTIONAL      ADTEXT      //Kernel//;
  SCE_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  SCE_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  SCE_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  SCE_EndValidity:    OPTIONAL      DATEDOM     //Kernel//;  !!                SN-GEN-D2
  SCE_Comment:        OPTIONAL      COMMENT;
  SCE_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  SCE_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  SCE_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  SCE_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  SCE_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  SCE_ChangeDate:     OPTIONAL      DATEDOM;                 !!                SN-GEN-A9
  SCE_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  SCE_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  SCE_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  SCE_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  SCE_XST_XferSetID:  OPTIONAL      BASEID;                  !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    SCE_BASEID, SCE_VERSION; !! PK
    SCE_CKO_OWNER, SCE_CK; !! UKC
END Scenarios;
```

```
TABLE Traffic_Segment_Groups =
      TSG_BASEID:                       BASEID      //Kernel//;   !! PK             SN-GEN-I1
      TSG_VERSION:                      VERSION     //Kernel//;   !! PK             SN-GEN-I2
      TSG_CKO_OWNER:                    CKO         //Kernel//;   !! UKC(.)->OWN
      TSG_CK_OWNER:                     CK          //Kernel//;   !! UKC
      TSG_NAME:           OPTIONAL      NAME        //Kernel//;
      TSG_VALUE:          OPTIONAL      NUM7_4      //Kernel//;
      TSG_SCA_CTK_BASEID:               BASEID;                   !! FKL(1)->SCA
      TSG_SCA_CTK_CKO-Owner:            OWNER       //Kernel//;   !! FKC(1)->SCA
      TSG_SCA_CTT_TextID:               TEXTID      //Kernel//;   !! FKC(1)->SCA
      TSG_SCA_CTT_Lng_code:             LANGCODE    //Kernel//;   !! FKC(1)->SCA
      TSG_SCA_CTK_RefDate:              DATEDOM     //Kernel//;   !! FKC(1)->SCA
      TSG_TYPADTEXT:      OPTIONAL      ADTEXT      //Kernel//;
      TSG_VRS_Code:                     VERSCODE    //Kernel//;   !!                SN-GEN-A1
      TSG_RefDate:                      DATEDOM     //Kernel//;   !!                SN-GEN-D3
      TSG_BeginValidity:                DATEDOM     //Kernel//;   !!                SN-GEN-D1
      TSG_EndValidity:    OPTIONAL      DATEDOM     //Kernel//;   !!                SN-GEN-D2
      TSG_Comment:        OPTIONAL      COMMENT;
      TSG_ODO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A2
      TSG_OrigSubDBID:                  DBID;                     !!                SN-GEN-A3
      TSG_DAO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A4
      TSG_DataOwner:                    DATAOWNER;                !!                SN-GEN-A5
      TSG_CreateDate:                   DATEDOM;                  !!                SN-GEN-A8
      TSG_ChangeDate:     OPTIONAL      DATEDOM;                  !!                SN-GEN-A9
      TSG_CHO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A10
      TSG_ChangeUser:                   ORAUSER;                  !!                SN-GEN-A11
      TSG_ITS_Code:                     ITSCODE;                  !!                SN-GEN-A6
      TSG_IntegrityDate:                DATEDOM;                  !!                SN-GEN-A7
      TSG_XST_XferSetID:  OPTIONAL      BASEID;                   !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
        TSG_BASEID, TSG_VERSION; !! PK
        TGS_CKO_OWNER, TGS_CK; !! UKC
    END Traffic_Locations;
```

```
TABLE Traffic_Segments =
  TRS_BASEID:                    BASEID      //Kernel//;   !! PK             SN-GEN-I1
  TRS_VERSION:                   VERSION     //Kernel//;   !! PK             SN-GEN-I2
  TRS_STANDARDSEQ:               NUM5        //Kernel//;
  TRS_AXE_BASEID:                BASEID;                   !! FKL(1)->AXE
  TRS_AXE_CKO_OWNER:             CKO         //Kernel//;   !! FKC(1)->AXE
  TRS_AXE_CK_OWNER:              CK          //Kernel//;   !! FKC(1)->AXE
  TRS_AXE_POS_CODE:              POSCODE     //Kernel//;   !! FKC(1)->AXE
  TRS_RPT_BASEID1:               BASEID                    !! FKL(2)->RPT
  TRS_RPT_CKO_OWNER1:            CKO         //Kernel//;   !! FKC(2)->RPT
  TRS_RPT_CK1:                   CK          //Kernel//;   !! FKC(2)->RPT
  TRS_RPDIST1:                   U_REL       //Kernel//;
  TRS_RPT_BASEID2:               BASEID                    !! FKL(3)->RPT
  TRS_RPT_CKO_OWNER2:            CKO         //Kernel//;   !! FKC(3)->RPT
  TRS_RPT_CK2:                   CK          //Kernel//;   !! FKC(3)->RPT
  TRS_RPDIST2:                   U_REL       //Kernel//;
  TRS_TSG_BASEID:                BASEID                    !! FKL(4)->TSG
  TRS_TSG_VERSION:               VERSION                   !! FKL(4)->TSG
  TRS_TSG_CKO_OWNER:             CKO         //Kernel//;   !! FKC(4)->TSG
  TRS_TSG_CK:                    CK          //Kernel//;   !! FKC(4)->TSG
  TRS_VRS_Code:                  VERSCODE    //Kernel//;   !!                SN-GEN-A1
  TRS_RefDate:                   DATEDOM     //Kernel//;   !!                SN-GEN-D3
  TRS_BeginValidity:             DATEDOM     //Kernel//;   !!                SN-GEN-D1
  TRS_EndValidity:     OPTIONAL  DATEDOM     //Kernel//;   !!                SN-GEN-D2
  TRS_Comment:         OPTIONAL  COMMENT;
  TRS_ODO_Owner:                 OWNER;                    !! FKL(.)->OWN   SN-GEN-A2
  TRS_OrigSubDBID:               DBID;                     !!               SN-GEN-A3
  TRS_DAO_Owner:                 OWNER;                    !! FKL(.)->OWN   SN-GEN-A4
  TRS_DataOwner:                 DATAOWNER;                !!               SN-GEN-A5
  TRS_CreateDate:                DATEDOM;                  !!               SN-GEN-A8
  TRS_ChangeDate:      OPTIONAL  DATEDOM;                  !!               SN-GEN-A9
  TRS_CHO_Owner:                 OWNER;                    !! FKL(.)->OWN   SN-GEN-A10
  TRS_ChangeUser:                ORAUSER;                  !!               SN-GEN-A11
  TRS_ITS_Code:                  ITSCODE;                  !!               SN-GEN-A6
  TRS_IntegrityDate:             DATEDOM;                  !!               SN-GEN-A7
  TRS_XST_XferSetID:   OPTIONAL  BASEID;                   !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    TRS_BASEID, TRS_VERSION; !! PK
END Traffic_Locations
```

```
TABLE Traffic_Location =
  TRL_BASEID:                       BASEID      //Kernel//;   !! PK             SN-GEN-I1
  TRL_VERSION:                      VERSION     //Kernel//;   !! PK             SN-GEN-I2
  TRL_Orientation_Code:             ORICODE     //Kernel//;
  TRL_TRS_BASEID:                   BASEID      //Kernel//;   !! FKL(1)->TRS
  TRL_TRS_VERSION:                  VERSION     //Kernel//;   !! FKL(1)->TRS
  TRL_RPT_BASEID1:                  BASEID                    !! FKL(2)->RPT
  TRL_RPT_CKO_OWNER1:               CKO         //Kernel//;   !! FKC(2)->RPT
  TRL_RPT_CK1:                      CK          //Kernel//;   !! FKC(2)->RPT
  TRL_RPDIST1:                      U_REL       //Kernel//;
  TRL_WIDTH1:                       NUM2_2      //Kernel//;
  TRL_LATDIST1:                     NUM3_3      //Kernel//;
  TRL_RPT_BASEID2:                  BASEID                    !! FKL(3)->RPT
  TRL_RPT_CKO_OWNER2:               CKO         //Kernel//;   !! FKC(3)->RPT
  TRL_RPT_CK2:                      CK          //Kernel//;   !! FKC(3)->RPT
  TRL_RPDIST2:                      U_REL       //Kernel//;
  TRL_WIDTH2:                       NUM2_2      //Kernel//;
  TRL_LATDIST2:                     NUM3_3      //Kernel//;
  TRL_LANENUMBER:                   NUM2        //Kernel//;
  TRL_VRS_Code:                     VERSCODE    //Kernel//;   !!               SN-GEN-A1
  TRL_RefDate:                      DATEDOM     //Kernel//;   !!               SN-GEN-D3
  TRL_BeginValidity:                DATEDOM     //Kernel//;   !!               SN-GEN-D1
  TRL_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;   !!               SN-GEN-D2
  TRL_Comment:         OPTIONAL     COMMENT;
  TRL_ODO_Owner:                    OWNER;                    !! FKL(.)->OWN   SN-GEN-A2
  TRL_OrigSubDBID:                  DBID;                     !!               SN-GEN-A3
  TRL_DAO_Owner:                    OWNER;                    !! FKL(.)->OWN   SN-GEN-A4
  TRL_DataOwner:                    DATAOWNER;                !!               SN-GEN-A5
  TRL_CreateDate:                   DATEDOM;                  !!               SN-GEN-A8
  TRL_ChangeDate:      OPTIONAL     DATEDOM;                  !!               SN-GEN-A9
  TRL_CHO_Owner:                    OWNER;                    !! FKL(.)->OWN   SN-GEN-A10
  TRL_ChangeUser:                   ORAUSER;                  !!               SN-GEN-A11
  TRL_ITS_Code:                     ITSCODE;                  !!               SN-GEN-A6
  TRL_IntegrityDate:                DATEDOM;                  !!               SN-GEN-A7
  TRL_XST_XferSetID:   OPTIONAL     BASEID;                   !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    TRL_BASEID, TRL_VERSION; !! PK
END Traffic_Locations
```

```
TABLE Time_Series =
  TIS_BaseID:                    BASEID;                  !! PK             SN-GEN-I1
  TIS_Version:                   VERSION;                 !! PK             SN-GEN-I2
  TIS_CKO_Owner:                 CKO        //Kernel//;   !! UKC(1)
  TIS_CK:                        CK5        //Kernel//;   !! UKC(1)
  TIS_Name:           OPTIONAL   NAME2      //Kernel//;
  TIS_StandardSeq:               NUM5       //Kernel//;
  TIS_Kind:                      KIND       //Kernel//;
  TIS_VFL_BaseID:                BASEID                   !! FKL(2)->VFL
  TIS_VFL_Version:               VERSION                  !! FKL(2)->VFL
  TIS_VFL_CK_Owner:              CKO        //Kernel//;   !! FKC(2)->VFL
  TIS_VFL_CK:                    CK         //Kernel//;   !! FKC(2)->VFL
  TIS_VFL_VRS_Code:              VERSCODE   //Kernel//;   !! FKC(2)->VFL
  TIS_EPI_BaseID1:    OPTIONAL   BASEID     //Kernel//;   !! FKL(3)->EPI
  TIS_EPI_Version1:   OPTIONAL   VERSION    //Kernel//;   !! FKL(3)->EPI
  TIS_EPI_CKO_Owner1: OPTIONAL   CKO        //Kernel//;   !! FKC(3)->EPI
  TIS_EPI_CK1:        OPTIONAL   CK5        //Kernel//;   !! FKC(3)->EPI
  TIS_EPI_VRS_Code1:  OPTIONAL   VERSCODE   //Kernel//;   !! FKC(3)->EPI
  TIS_EPI_BaseID2:    OPTIONAL   BASEID     //Kernel//;   !! FKL(4)->EPI
  TIS_EPI_Version2:   OPTIONAL   VERSION    //Kernel//;   !! FKL(4)->EPI
  TIS_EPI_CKO_Owner2: OPTIONAL   CKO        //Kernel//;   !! FKC(4)->EPI
  TIS_EPI_CK2:        OPTIONAL   CK5        //Kernel//;   !! FKC(4)->EPI
  TIS_EPI_VRS_Code2:  OPTIONAL   VERSCODE   //Kernel//;   !! FKC(4)->EPI
  TIS_EPI_BaseID3:    OPTIONAL   BASEID     //Kernel//;   !! FKL(5)->EPI
  TIS_EPI_Version3:   OPTIONAL   VERSION    //Kernel//;   !! FKL(5)->EPI
  TIS_EPI_CKO_Owner3: OPTIONAL   CKO        //Kernel//;   !! FKC(5)->EPI
  TIS_EPI_CK3:        OPTIONAL   CK5        //Kernel//;   !! FKC(5)->EPI
  TIS_EPI_VRS_Code3:  OPTIONAL   VERSCODE   //Kernel//;   !! FKC(5)->EPI
  TIS_SCL_BaseID:                BASEID;                  !! FKL(6)->SCL
  TIS_SCL_Version:               VERSION;                 !! FKL(6)->SCL
  TIS_SCL_CK_Owner:              CKO        //Kernel//;   !! FKC(6)->SCL
  TIS_SCL_CK:                    CK5        //Kernel//;   !! FKC(6)->SCL
  TIS_SCE_BaseID:                BASEID;                  !! FKL(7)->SCE
  TIS_SCE_Version:               VERSION;   //Kernel//;   !! FKL(7)->SCE
  TIS_SCE_CK_Owner:              CKO        //Kernel//;   !! FKC(7)->SCE
  TIS_SCE_CK:                    CK         //Kernel//;   !! FKC(7)->SCE
  TIS_PRO_BaseID:     OPTIONAL   BASEID;                  !! FKL(8)->PRO
  TIS_PRO_CK_Owner:   OPTIONAL   CKO        //Kernel//;   !! FKC(8)->PRO
  TIS_PRO_CK:         OPTIONAL   CK         //Kernel//;   !! FKC(8)->PRO
  TIS_PRO_VRS_Code    OPTIONAL   VERSCODE   //Kernel//;   !! FKC(8)->PRO
  TIS_Space_Code:                SPACODE    //Kernel//;
  TIS_TRL_BaseID:     OPTIONAL   BASEID     //Kernel//;   !! FKL(9)->TRL
  TIS_TRL_Version:    OPTIONAL   VERSION    //Kernel//;   !! FKL(9)->TRL
  TIS_TRL_AXE_CKO_Owner: OPTIONAL CKO       //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_AXE_CK:     OPTIONAL   CK3        //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_AXE_POS_Code OPTIONAL  POSCODE    //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_AXE_VRS_Code OPTIONAL  VERSCODE   //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_RPT_BaseID1: OPTIONAL  BASEID     //Kernel//;   !! FKL(.)->TRL
  TIS_TRL_RPT_CK1:    OPTIONAL   CK2        //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_RPT_VRS_Code: OPTIONAL VERSCODE   //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_RPDistBegin: OPTIONAL  U_ABS      //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_LatDistBegin: OPTIONAL NUM2_3     //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_WidthBegin:  OPTIONAL  NUM2_2     //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_RPT_BaseID2: OPTIONAL  BASEID     //Kernel//;   !! FKL(.)->TRL
  TIS_TRL_RPT_CK2:    OPTIONAL   CK2        //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_RPT_VRS_Code2 OPTIONAL VERSCODE   //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_RPDistEnd:   OPTIONAL  U_ABS      //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_LatDistEnd: OPTIONAL   NUM2_3     //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_WidthEnd:    OPTIONAL  NUM2_2     //Kernel//;   !! FKC(9)->TRL
  TIS_TRL_VRS_Code:    OPTIONAL  VERSCODE   //Kernel//;   !! FKC(9)->TRL
  TIS_TRS_BaseID:      OPTIONAL  BASEID     //Kernel//;   !! FKL(10)->TRS
  TIS_TRS_Version:     OPTIONAL  VERSION    //Kernel//;   !! FKL(10)->TRS
  TIS_TRS_AXE_CKO_Owner: OPTIONAL CKO       //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_AXE_CK:      OPTIONAL  CK3        //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_AXE_POS_Code OPTIONAL  POSCODE    //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_AXE_VRS_Code OPTIONAL  VERSCODE   //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_RPT_BaseID1: OPTIONAL  BASEID     //Kernel//;   !! FKL(.)->TRS
  TIS_TRS_RPT_CK1:     OPTIONAL  CK2        //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_RPT_VRS_Code: OPTIONAL VERSCODE   //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_RPDistBegin: OPTIONAL  U_ABS      //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_RPT_BaseID2: OPTIONAL  BASEID     //Kernel//;   !! FKL(.)->TRS
  TIS_TRS_RPT_CK2:     OPTIONAL  CK2        //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_RPT_VRS_Code2 OPTIONAL VERSCODE   //Kernel//;   !! FKC(10)->TRS
  TIS_TRS_RPDistEnd:   OPTIONAL  U_ABS      //Kernel//;   !! FKC(10)->TRS
  TIS_TSG_BaseID:      OPTIONAL  BASEID                   !! FKL(11)->TSG
  TIS_TSG_VERSION:     OPTIONAL  VERSION                  !! FKL(11)->TSG
  TIS_TSG_CKO_Owner:   OPTIONAL  CKO        //Kernel//;   !! FKC(11)->TSG
  TIS_TSG_CK:          OPTIONAL  CK5        //Kernel//;   !! FKC(11)->TSG
  TIS_TSG_VRS_Code:    OPTIONAL  VERSCODE   //Kernel//;   !! FKC(11)->TSG
  TIS_LNK_BaseID:      OPTIONAL  BASEID     //Kernel//;   !! FKL(12)->LNK
```

```
  TIS_LNK_AXE_CKO_Owner: OPTIONAL   OWNER       //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_AXE_CK:      OPTIONAL     CK3         //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_AXE_POS_Code: OPTIONAL    POSCODE     //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_AXE_VRS_Code: OPTIONAL    VERSCODE    //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_NOD_CKO_Owner_1: OPTIONAL OWNER   //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_NOD_CK_1: OPTIONAL    CK4         //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_NOD_VRS_CODE: OPTIONAL VERSCODE   //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_RPT_CK_1: OPTIONAL    CK2         //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_RPT_VRS_Code_1: OPTIONAL VERSCODE //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_VRS_Code_1: OPTIONAL  VERSCODE    //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_NOD_CKO_Owner_2: OPTIONAL OWNER   //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_NOD_CK_2: OPTIONAL    CK4         //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_NOD_VRS_CODE_2: OPTIONAL VERSCODE //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_RPT_CK_2: OPTIONAL    CK3         //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_RPT_VRS_Code_2: OPTIONAL VERSCODE //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_NLO_VRS_Code_2: OPTIONAL  VERSCODE    //Kernel//;   !! FKC(12)->LNK
  TIS_LNK_LKG_BASEID:  OPTIONAL     BASEID      //Kernel//;   !! FKL(13)->LKG
  TIS_LNK_LKG_CKO_Owner: OPTIONAL   CKO         //Kernel//;   !! FKC(13)->LKG
  TIS_LNK_LKG_CK:      OPTIONAL     CK2         //Kernel//;   !! FKC(13)->LKG
  TIS_LNK_LKG_SCA_SLT_CTK_CKO_Owner: OPTIONAL CKO //Kernel//; !! FKC(13)->LKG
  TIS_LNK_LKG_SCA_SLT_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//; !! FKC(13)->LKG
  TIS_LNK_LKG_SCA_SLT_CTT_TextID: OPTIONAL TEXTID //Kernel//; !! FKC(13)->LKG
  TIS_LKG_BaseID:      OPTIONAL     BASEID      //Kernel//;   !! FKL(13)->LKG
  TIS_LKG_CKO_Owner:   OPTIONAL     CKO         //Kernel//;   !! FKC(13)->LKG
  TIS_LKG_CK:          OPTIONAL     CK2         //Kernel//;   !! FKC(13)->LKG
  TIS_LKG_SCA_SLT_CTK_CKO_Owner: OPTIONAL CKO   //Kernel//;   !! FKC(13)->LKG
  TIS_LKG_SCA_SLT_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//; !! FKC(13)->LKG
  TIS_LKG_SCA_SLT_CTT_TextID: OPTIONAL TEXTID   //Kernel//;   !! FKC(13)->LKG
  TIS_Orientation_Code:             ORICODE     //Kernel//;
  TIS_VRS_Code:                     VERSCODE    //Kernel//;   !!                SN-GEN-A1
  TIS_RefDate:                      DATEDOM     //Kernel//;   !!                SN-GEN-D3
  TIS_BeginValidity:                DATEDOM     //Kernel//;   !!                SN-GEN-D1
  TIS_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;   !!                SN-GEN-D2
  TIS_Comment:         OPTIONAL     COMMENT;
  TIS_ODO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A2
  TIS_OrigSubDBID:                  DBID;                     !!                SN-GEN-A3
  TIS_DAO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A4
  TIS_DataOwner:                    DATAOWNER;                !!                SN-GEN-A5
  TIS_CreateDate:                   DATEDOM;                  !!                SN-GEN-A8
  TIS_ChangeDate:      OPTIONAL     DATEDOM;                  !!                SN-GEN-A9
  TIS_CHO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A10
  TIS_ChangeUser:                   ORAUSER;                  !!                SN-GEN-A11
  TIS_ITS_Code:                     ITSCODE;                  !!                SN-GEN-A6
  TIS_IntegrityDate:                DATEDOM;                  !!                SN-GEN-A7
  TIS_XST_XferSetID:   OPTIONAL     BASEID;                   !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    TIS_BASEID, TIS_VERSION;
END Time_Series;
```

```
TABLE Traffic_Value_Classification =
  TCL_BASEID                        BASEID;                !! PK             SN-GEN-I1
  TCL_VERSION:                      VERSION;               !! PK             SN-GEN-I2
  TCL_CKO_OWNER:                    CKO         //Kernel//   !! UKC(.)->OWN
  TCL_CK:                           CK          //Kernel//   !! UKC
  TCL_NAME:           OPTIONAL      NAME        //Kernel//;
  TCL_STANDARDSEQ:                  NUM5        //Kernel//;
  TCL_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  TCL_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  TCL_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  TCL_EndValidity:    OPTIONAL      DATEDOM     //Kernel//;  !!                SN-GEN-D2
  TCL_Comment:        OPTIONAL      COMMENT;
  TCL_ODO_Owner:                    OWNER;                 !! FKL(.)->OWN    SN-GEN-A2
  TCL_OrigSubDBID:                  DBID;                  !!                SN-GEN-A3
  TCL_DAO_Owner:                    OWNER;                 !! FKL(.)->OWN    SN-GEN-A4
  TCL_DataOwner:                    DATAOWNER;             !!                SN-GEN-A5
  TCL_CreateDate:                   DATEDOM;               !!                SN-GEN-A8
  TCL_ChangeDate:     OPTIONAL      DATEDOM;               !!                SN-GEN-A9
  TCL_CHO_Owner:                    OWNER;                 !! FKL(.)->OWN    SN-GEN-A10
  TCL_ChangeUser:                   ORAUSER;               !!                SN-GEN-A11
  TCL_ITS_Code:                     ITSCODE;               !!                SN-GEN-A6
  TCL_IntegrityDate:                DATEDOM;               !!                SN-GEN-A7
  TCL_XST_XferSetID:  OPTIONAL      BASEID;                !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    TCL_BASEID, TCL_VERSION; !! PK
    TCL_CKO_OWNER, TCL_CK; !! UKC  (+ STANDARDSEQ)
END Variation_Functions;
```

```
TABLE Traffic_Value_Classes =
  TVC_BASEID                        BASEID;                  !! PK             SN-GEN-I1
  TVC_VERSION:                      VERSION;                 !! PK             SN-GEN-I2
  TVC_TCL_BASEID:                   BASEID;                  !! FKL(1)->TCL
  TVC_TCL_VERSION:                  VERSION;                 !! FKL(1)->TCL
  TVC_TCL_CKO_OWNER:                CKO         //Kernel//   !! UKC(1)->TCL
  TVC_TCL_CK:                       CK          //Kernel//   !! UKC(1)->TCL
  TVC_SCA_CTK_BASEID:               BASEID;                  !! FKL(2)->SCA
  TVC_SCA_CTK_CKO-Owner:            OWNER       //Kernel//;  !! UKC(2)->SCA
  TVC_SCA_CTT_TextID:               TEXTID      //Kernel//;  !! UKC(2)->SCA
  TVC_SCA_CTT_Lng_code:             LANGCODE    //Kernel//;  !! UKC(2)->SCA
  TVC_SCA_CTK_RefDate:              DATEDOM     //Kernel//;  !! UKC(2)->SCA
  TVC_TYPADTEXT:       OPTIONAL     ADTEXT      //Kernel//;
  TVC_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  TVC_OrigSubDBID:                  DBID;                    !!                  SN-GEN-A3
  TVC_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  TVC_DataOwner:                    DATAOWNER;               !!                  SN-GEN-A5
  TVC_CreateDate:                   DATEDOM;                 !!                  SN-GEN-A8
  TVC_ChangeDate:      OPTIONAL     DATEDOM;                 !!                  SN-GEN-A9
  TVC_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  TVC_ChangeUser:                   ORAUSER;                 !!                  SN-GEN-A11
  TVC_ITS_Code:                     ITSCODE;                 !!                  SN-GEN-A6
  TVC_IntegrityDate:                DATEDOM;                 !!                  SN-GEN-A7
  TVC_XST_XferSetID:   OPTIONAL     BASEID;                  !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    TVC_BASEID, TVC_VERSION; !! PK
    TVC_TCL_CKO_OWNER, TVC_TCL_CK, TVC_SCA_CTK_CKO_Owner, TVC_CTT_TexteID,
     TVC_SCA_CTT_Lng_code, TVC_SCA_CTK_RefDate; !! UKC
END Traffic Value_Classes;
```

```
TABLE Variation_Functions =
    !! (D:‘Ganglinien‘ F: ‘Courbe de variation‘)
      VAF_BaseID:                       BASEID;                 !! PK             SN-GEN-I1
      VAF_Version:                      VERSION;                !! PK             SN-GEN-I2
      VAF_TIS_BaseID:                   BASEID      //Kernel//; !! FKL(1)->TIS
      VAF_TIS_Version:                  VERSION     //Kernel//; !! FKL(1)->TIS
      VAF_TIS_CKO_Owner:                OWNER       //Kernel//; !! UKC(1)->TIS
      VAF_TIS_CK:                       CK5         //Kernel//; !! UKC(1)->TIS
      VAF_TIS_VRS_Code:                 VERSCODE    //Kernel//; !! UKC(1)->TIS
      VAF_ACCEPTANCE_Code:              ACCCODE     //Kernel//;
      VAF_COMPLETE_Code:                COMCODE     //Kernel//;
      VAF_COHERENCE_Code:               COHCODE     //Kernel//;
      VAF_TVC_BaseID:                   BASEID      //Kernel//; !! FKL(2)->TVC
      VAF_TVC_Version:                  VERSION     //Kernel//; !! FKL(2)->TVC
      VAF_TVC_TCL_BaseID:               BASEID      //Kernel//; !! FKL(.)->TVC
      VAF_TVC_TCL_Version:              VERSION     //Kernel//; !! FKL(.)->TVC
      VAF_TVC_TCL_CKO_Owner:            OWNER       //Kernel//; !! UKC(2)->TVC
      VAF_TVC_TCL_CK:                   CK5         //Kernel//; !! UKC(2)->TVC
      VAF_TVC_TCL_VRS_Code:             VERSCODE    //Kernel//; !! UKC(2)->TVC
      VAF_TVC_SCA_VCT_CTK_BaseID:       BASEID      //Kernel//; !! FKL(3)->TVC
      VAF_TVC_SCA_VCT_CTK_CKO_Owner:    OWNER       //Kernel//; !! UKC(3)->SCA
      VAF_TVC_SCA_VCT_CTT_LNG_Code:     LANGCODE    //Kernel//; !! UKC(3)->SCA
      VAF_TVC_SCA_VCT_CTT_TextID:       TEXTID      //Kernel//; !! UKC(3)->SCA
      VAF_Comment:          OPTIONAL    COMMENT     //Kernel//;
      VAF_RefDate_A:                    DATEDOM     //Kernel//; !!                SN-GEN-D3
      VAF_BeginValidity_A:              DATEDOM     //Kernel//; !! UKC            SN-GEN-D1
      VAF_EndValidity_A:    OPTIONAL    DATEDOM     //Kernel//; !!                SN-GEN-D2
      VAF_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
      VAF_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
      VAF_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
      VAF_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
      VAF_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
      VAF_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
      VAF_ChangeDate:       OPTIONAL    DATEDOM;                !!                SN-GEN-A9
      VAF_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
      VAF_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
      VAF_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
      VAF_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
      VAF_XST_XfrSetID:     OPTIONAL    BASEID;                 !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
        VAF_BASEID, VAF_VERSION;
        VAF_TIS_CKO_Owner, VAF_TIS_CK, VAF_TIS_VRS_Code,
        VAF_TVC_TCL_CKO_Owner, VAF_TVC_TCL_CK, VAF_TVC_TCL_VRS_Code,
        VAF_TVC_SCA_VCT_CAT_CKO_Owner, VAF_TVC_SCA_VCT_CAT_CK, VAF_TVC_SCA_VCT_CTK_CKO_Owner,
        VAF_TVC_SCA_VCT_CTT_LNG_Code, VAF_TVC_SCA_VCT_CTT_TextID; !! UKC
END Variation_Functions;

!!=============================================================================================
!! DEF: Variation Functions is a groupe of caracteristics appearing periodically.
!! Variation Function can be considered as a descrete function. On the x-axis, we have
!! represented a bounded domain (ex: year) which is subdivided into units with constant
!! width, called a base (ex: week), and on the y-axis the parameter is represented (ex: number
!! of vehicle). The difference between time series and the variation function is the constant
!! base of a variation function.
!!=============================================================================================
!! Attribut description
!!    VAF_BASEID:             First part of the primary key
!!    VAF_VERSION:            Version number, part of primary key
!!    VAF_TIS_BASEID:         Logical Key of a time series
!!    VAF_TIS_VERSION:        Version of time series
!!    VAF_TIS_CKO_Owner:      Owner of conceptual key of the time series
!!    VAF_TIS_CK              Conceptual Key of time series
!!    VAF_ACCEPTANCE_CODE:    Acceptance Code for User
!!    VAF_COMPLETE_CODE:      Completeness Code in the system
!!    VAF_COHERENCE_CODE:     Coherence Code for user and system
!!    VAF_TVC_BASEID:         1st logical foreign key of Traffic Value Classification
!!    VAF_TVC_VERSION:        2nd logical foreign
!!    VAF_TVC_TCL_BaseID, VAF_TVC_TCL_Version, VAF_TVC_TCL_CKO_Owner, VAF_TVC_TCL_CK,
!!    VAF_TVC_TCL_VRS_Code, VAF_TVC_SCA_VCT_CTK_BaseID, VAF_TVC_SCA_VCT_CAT_CKO_Owner,
!!    VAF_TVC_SCA_VCT_CAT_CK, VAF_TVC_SCA_VCT_CTK_CKO_Owner, VAF_TVC_SCA_VCT_CTT_LNG_Code,
!!    VAF_TVC_SCA_VCT_CTT_TextID:  Necessary attributs to identify a Traffic Value
!!                                 Classification
!!=============================================================================================
```

```
VIEW Traffic_Values =
    !! (D:‘Makroobjekt Verkehrswerte‘ F:‘ Objet macro valeurs du trafic ‘)
      OVA_TIS_BaseID:                   BASEID;     !! <-TIS_BaseID
      OVA_TIS_Version:                  VERSION;    !! <-TIS_Version
      OVA_TIS_CKO_Owner:                CKO;        !! <-TIS_CKO_Owner
      OVA_TIS_CK:                       CK5;        !! <-TIS_CK
      OVA_TIS_Name:       OPTIONAL      NAME2;      !! <-TIS_Name
      OVA_TIS_StandardSeq:              NUM5;       !! <-TIS_StandardSeq
      OVA_TIS_Kind:                     KIND;       !! <-TIS_Kind
      OVA_TIS_VFL_BaseID:               BASEID;     !! <-TIS_VFL_BaseID
      OVA_TIS_VFL_Version:              VERSION;    !! <-TIS_VFL_Version
      OVA_TIS_VFL_CK_Owner:             CKO;        !! <-TIS_VFL_CK_Owner
      OVA_TIS_VFL_CK:                   CK;         !! <-TIS_VFL_CK
      OVA_TIS_VFL_VRS_Code:             VERSCODE;   !! <-TIS_VFL_VRS_Code
      OVA_TIS_EPI_BaseID1: OPTIONAL     BASEID;     !! <-TIS_EPI_BaseID1
      OVA_TIS_EPI_Version1: OPTIONAL    VERSION;    !! <-TIS_EPI_Version1
      OVA_TIS_EPI_CKO_Owner1: OPTIONAL  CKO;        !! <-TIS_EPI_CKO_Owner1
      OVA_TIS_EPI_CK1:    OPTIONAL      CK5;        !! <-TIS_EPI_CK1
      OVA_TIS_EPI_VRS_Code1: OPTIONAL   VERSCODE;   !! <-TIS_EPI_VRS_Code1
      OVA_TIS_EPI_BaseID2: OPTIONAL     BASEID;     !! <-TIS_EPI_BaseID2
      OVA_TIS_EPI_Version2: OPTIONAL    VERSION;    !! <-TIS_EPI_Version2
      OVA_TIS_EPI_CKO_Owner2: OPTIONAL  CKO;        !! <-TIS_EPI_CKO_Owner2
      OVA_TIS_EPI_CK2:    OPTIONAL      CK5;        !! <-TIS_EPI_CK2
      OVA_TIS_EPI_VRS_Code2:OPTIONAL    VERSCODE;   !! <-TIS_EPI_VRS_Code2
      OVA_TIS_EPI_BaseID3: OPTIONAL     BASEID;     !! <-TIS_EPI_BaseID3
      OVA_TIS_EPI_Version3: OPTIONAL    VERSION;    !! <-TIS_EPI_Version3
      OVA_TIS_EPI_CKO_Owner3: OPTIONAL  CKO;        !! <-TIS_EPI_CKO_Owner3
      OVA_TIS_EPI_CK3:    OPTIONAL      CK5;        !! <-TIS_EPI_CK3
      OVA_TIS_EPI_VRS_Code3: OPTIONAL   VERSCODE;   !! <-TIS_EPI_VRS_Code3
      OVA_TIS_SCL_BaseID:               BASEID;;    !! <-TIS_SCL_BaseID
      OVA_TIS_SCL_Version:              VERSION;;   !! <-TIS_SCL_Version
      OVA_TIS_SCL_CK_Owner:             CKO;        !! <-TIS_SCL_CK_Owner
      OVA_TIS_SCL_CK:                   CK5;        !! <-TIS_SCL_CK
      OVA_TIS_SCE_BaseID:               BASEID;;    !! <-TIS_SCE_BaseID
      OVA_TIS_SCE_Version:              VERSION;;   !! <-TIS_SCE_Version
      OVA_TIS_SCE_CK_Owner:             CKO;        !! <-TIS_SCE_CK_Owner
      OVA_TIS_SCE_CK:                   CK;         !! <-TIS_SCE_CK
      OVA_TIS_PRO_BaseID:  OPTIONAL     BASEID;;    !! <-TIS_PRO_BaseID
      OVA_TIS_PRO_CK_Owner: OPTIONAL    CKO;        !! <-TIS_PRO_CK_Owner
      OVA_TIS_PRO_CK:      OPTIONAL     CK;         !! <-TIS_PRO_CK
      OVA_TIS_PRO_VRS_Code: OPTIONAL    VERSCODE;   !! <-TIS_PRO_VRS_Code
      OVA_TIS_Space_Code:               SPACODE;    !! <-TIS_Space_Code
      OVA_TIS_TRL_BaseID:  OPTIONAL     BASEID;     !! <-TIS_TRL_BaseID
      OVA_TIS_TRL_Version: OPTIONAL     VERSION;    !! <-TIS_TRL_Version
      OVA_TIS_TRL_AXE_CKO_Owner: OPTIONAL CKO;      !! <-TIS_TRL_AXE_CKO_Owner
      OVA_TIS_TRL_AXE_CK:  OPTIONAL     CK3;        !! <-TIS_TRL_AXE_CK
      OVA_TIS_TRL_AXE_POS_Code: OPTIONAL POSCODE;   !! <-TIS_TRL_AXE_POS_Code
      OVA_TIS_TRL_AXE_VRS_Code: OPTIONAL VERSCODE;  !! <-TIS_TRL_AXE_VRS_Code
      OVA_TIS_TRL_RPT_BaseID1: OPTIONAL BASEID;     !! <-TIS_TRL_RPT_BaseID1
      OVA_TIS_TRL_RPT_CK1: OPTIONAL     CK2;        !! <-TIS_TRL_RPT_CK1
      OVA_TIS_TRL_RPT_VRS_Code: OPTIONAL VERSCODE;  !! <-TIS_TRL_RPT_VRS_Code
      OVA_TIS_TRL_RPDistBegin: OPTIONAL U_ABS;      !! <-TIS_TRL_RPDistBegin
      OVA_TIS_TRL_LatDistBegin: OPTIONAL NUM2_3;    !! <-TIS_TRL_LatDistBegin
      OVA_TIS_TRL_WidthBegin: OPTIONAL  NUM2_2;     !! <-TIS_TRL_WidthBegin
      OVA_TIS_TRL_RPT_BaseID2: OPTIONAL BASEID;     !! <-TIS_TRL_RPT_BaseID2
      OVA_TIS_TRL_RPT_CK2: OPTIONAL     CK2;        !! <-TIS_TRL_RPT_CK2
      OVA_TIS_TRL_RPT_VRS_Code2: OPTIONAL VERSCODE; !! <-TIS_TRL_RPT_VRS_Code2
      OVA_TIS_TRL_RPDistEnd: OPTIONAL   U_ABS;      !! <-TIS_TRL_RPDistEnd
      OVA_TIS_TRL_LatDistEnd: OPTIONAL  NUM2_3;     !! <-TIS_TRL_LatDistEnd
      OVA_TIS_TRL_WidthEnd: OPTIONAL    NUM2_2;     !! <-TIS_TRL_WidthEnd
      OVA_TIS_TRL_VRS_Code: OPTIONAL    VERSCODE;   !! <-TIS_TRL_VRS_Code
      OVA_TIS_TRS_BaseID:  OPTIONAL     BASEID;     !! <-TIS_TRS_BaseID
      OVA_TIS_TRS_Version: OPTIONAL     VERSION;    !! <-TIS_TRS_Version
      OVA_TIS_TRS_AXE_CKO_Owner: OPTIONAL CKO;      !! <-TIS_TRS_AXE_CKO_Owner
      OVA_TIS_TRS_AXE_CK:  OPTIONAL     CK3;        !! <-TIS_TRS_AXE_CK
      OVA_TIS_TRS_AXE_POS_Code: OPTIONAL POSCODE;   !! <-TIS_TRS_AXE_POS_Code
      OVA_TIS_TRS_AXE_VRS_Code: OPTIONAL VERSCODE;  !! <-TIS_TRS_AXE_VRS_Code
      OVA_TIS_TRS_RPT_BaseID1: OPTIONAL BASEID;     !! <-TIS_TRS_RPT_BaseID1
      OVA_TIS_TRS_RPT_CK1: OPTIONAL     CK2;        !! <-TIS_TRS_RPT_CK1
      OVA_TIS_TRS_RPT_VRS_Code: OPTIONAL VERSCODE;  !! <-TIS_TRS_RPT_VRS_Code
      OVA_TIS_TRS_RPDistBegin: OPTIONAL U_ABS;      !! <-TIS_TRS_RPDistBegin
      OVA_TIS_TRS_RPT_BaseID2: OPTIONAL BASEID;     !! <-TIS_TRS_RPT_BaseID2
      OVA_TIS_TRS_RPT_CK2: OPTIONAL     CK2;        !! <-TIS_TRS_RPT_CK2
      OVA_TIS_TRS_RPT_VRS_Code2: OPTIONAL VERSCODE; !! <-TIS_TRS_RPT_VRS_Code2
      OVA_TIS_TRS_RPDistEnd: OPTIONAL   U_ABS;      !! <-TIS_TRS_RPDistEnd
      OVA_TIS_TSG_BaseID:  OPTIONAL     BASEID;     !! <-TIS_TSG_BaseID
      OVA_TIS_TSG_VERSION: OPTIONAL     VERSION;    !! <-TIS_TSG_VERSION
      OVA_TIS_TSG_CKO_Owner: OPTIONAL   CKO;        !! <-TIS_TSG_CKO_Owner
      OVA_TIS_TSG_CK:      OPTIONAL     CK5;        !! <-TIS_TSG_CK
      OVA_TIS_TSG_VRS_Code:OPTIONAL     VERSCODE;   !! <-TIS_TSG_VRS_Code
```

```
OVA_TIS_LNK_BaseID:  OPTIONAL     BASEID;     !! <-TIS_LNK_BaseID
OVA_TIS_LNK_AXE_CKO_Owner: OPTIONAL OWNER;    !! <-TIS_LNK_AXE_CKO_Owner
OVA_TIS_LNK_AXE_CK:  OPTIONAL     CK3;        !! <-TIS_LNK_AXE_CK
OVA_TIS_LNK_AXE_POS_Code: OPTIONAL POSCODE;   !! <-TIS_LNK_AXE_POS_Code
OVA_TIS_LNK_AXE_VRS_Code: OPTIONAL VERSCODE;  !! <-TIS_LNK_AXE_VRS_Code
OVA_TIS_LNK_NLO_NOD_CKO_Owner_1: OPTIONAL OWNER; !! <-TIS_LNK_NLO_NOD_CKO_Owner_1
OVA_TIS_LNK_NLO_NOD_CK_1: OPTIONAL CK4;       !! <-TIS_LNK_NLO_NOD_CK_1
OVA_TIS_LNK_NLO_NOD_VRS_CODE: OPTIONAL VERSCODE; !! <-TIS_LNK_NLO_NOD_VRS_CODE
OVA_TIS_LNK_NLO_RPT_CK_1: OPTIONAL CK2;       !! <-TIS_LNK_NLO_RPT_CK_1
OVA_TIS_LNK_NLO_RPT_VRS_Code_1: OPTIONAL VERSCODE; !! <-TIS_LNK_NLO_RPT_VRS_Code_1
OVA_TIS_LNK_NLO_VRS_Code_1: OPTIONAL VERSCODE; !! <-TIS_LNK_NLO_VRS_Code_1
OVA_TIS_LNK_NLO_NOD_CKO_Owner_2: OPTIONAL OWNER; !! <-TIS_LNK_NLO_NOD_CKO_Owner_2
OVA_TIS_LNK_NLO_NOD_CK_2: OPTIONAL CK4;       !! <-TIS_LNK_NLO_NOD_CK_2
OVA_TIS_LNK_NLO_NOD_VRS_CODE_2: OPTIONAL VERSCODE; !! <-TIS_LNK_NLO_NOD_VRS_CODE_2
OVA_TIS_LNK_NLO_RPT_CK_2: OPTIONAL CK3;       !! <-TIS_LNK_NLO_RPT_CK_2
OVA_TIS_LNK_NLO_RPT_VRS_Code_2: OPTIONAL VERSCODE; !!<-TIS_LNK_NLO_RPT_VRS_Code_2
OVA_TIS_LNK_NLO_VRS_Code_2: OPTIONAL VERSCODE; !! <-TIS_LNK_NLO_VRS_Code_2
OVA_TIS_LNK_LKG_BASEID: OPTIONAL  BASEID;     !! <-TIS_LNK_LKG_BASEID
OVA_TIS_LNK_LKG_CKO_Owner: OPTIONAL CKO;      !! <-TIS_LNK_LKG_CKO_Owner
OVA_TIS_LNK_LKG_CK:  OPTIONAL     CK2;        !! <-TIS_LNK_LKG_CK
OVA_TIS_LNK_LKG_SCA_SLT_CTK_CKO_Owner: OPTIONAL CKO; !! <- TIS_LNK_LKG_SCA_SLT_CTK_CKO_Owner
OVA_TIS_LNK_LKG_SCA_SLT_CTT_LNG_Code: OPTIONAL LANGCODE; !! <-TIS_LNK_LKG_SCA_SLT_CTT_LNG_Code
OVA_TIS_LNK_LKG_SCA_SLT_CTT_TextID: OPTIONAL TEXTID;  !! <-TIS_LNK_LKG_SCA_SLT_CTT_TextID
OVA_TIS_LKG_BaseID:  OPTIONAL     BASEID;     !! <-TIS_LKG_BaseID
OVA_TIS_LKG_CKO_Owner: OPTIONAL   CKO;        !! <-TIS_LKG_CKO_Owner
OVA_TIS_LKG_CK:      OPTIONAL     CK2;        !! <-TIS_LKG_CK
OVA_TIS_LKG_SCA_SLT_CTK_CKO_Owner: OPTIONAL CKO; !! <-TIS_LKG_SCA_SLT_CTK_CKO_Owner
OVA_TIS_LKG_SCA_SLT_CTT_LNG_Code: OPTIONAL LANGCODE; !! <-TIS_LKG_SCA_SLT_CTT_LNG_Code
OVA_TIS_LKG_SCA_SLT_CTT_TextID: OPTIONAL TEXTID; !! <-TIS_LKG_SCA_SLT_CTT_TextID
OVA_TIS_Orientation_Code:         ORICODE;    !! <-TIS_Orientation_Code
OVA_TIS_VRS_Code:                 VERSCODE;   !! <-TIS_VRS_Code
OVA_TIS_RefDate:                  DATEDOM;    !! <-TIS_RefDate
OVA_TIS_BeginValidity:            DATEDOM;    !! <-TIS_BeginValidity
OVA_TIS_EndValidity: OPTIONAL     DATEDOM;    !! <-TIS_EndValidity
OVA_TIS_Comment:     OPTIONAL     COMMENT;    !! <-TIS_Comment
OVA_TIS_ODO_Owner:                OWNER;      !! <-TIS_ODO_Owner
OVA_TIS_OrigSubDBID:              DBID;       !! <-TIS_OrigSubDBID
OVA_TIS_DAO_Owner:                OWNER;      !! <-TIS_DAO_Owner
OVA_TIS_DataOwner:                DATAOWNER;  !! <-TIS_DataOwner
OVA_TIS_CreateDate:               DATEDOM;    !! <-TIS_CreateDate
OVA_TIS_ChangeDate:  OPTIONAL     DATEDOM;    !! <-TIS_ChangeDate
OVA_TIS_CHO_Owner:                OWNER;      !! <-TIS_CHO_Owner
OVA_TIS_ChangeUser:               ORAUSER;    !! <-TIS_ChangeUser
OVA_TIS_ITS_Code:                 ITSCODE;    !! <-TIS_ITS_Code
OVA_TIS_IntegrityDate:            DATEDOM;    !! <-TIS_IntegrityDate
OVA_TIS_XST_XferSetID: OPTIONAL   BASEID;     !! <-TIS_XST_XferSetID
OVA_VAF_BaseID:                   BASEID;     !! <-VAF_BaseID
OVA_VAF_Version:                  VERSION;    !! <-VAF_Version
OVA_VAF_TIS_BaseID:               BASEID;     !! <-VAF_TIS_BaseID
OVA_VAF_TIS_Version:              VERSION;    !! <-VAF_TIS_Version
OVA_VAF_TIS_CKO_Owner:            OWNER;      !! <-VAF_TIS_CKO_Owner
OVA_VAF_TIS_CK:                   CK5;        !! <-VAF_TIS_CK
OVA_VAF_TIS_VRS_Code:             VERSCODE;   !! <-VAF_TIS_VRS_Code
OVA_VAF_ACCEPTANCE_Code:          ACCCODE;    !! <-VAF_ACCEPTANCE_Code
OVA_VAF_COMPLETE_Code:            COMCODE;    !! <-VAF_COMPLETE_Code
OVA_VAF_COHERENCE_Code:           COHCODE;    !! <-VAF_COHERENCE_Code
OVA_VAF_TVC_BaseID:               BASEID;     !! <-VAF_TVC_BaseID
OVA_VAF_TVC_Version:              VERSION;    !! <-VAF_TVC_Version
OVA_VAF_TVC_TCL_BaseID:           BASEID;     !! <-VAF_TVC_TCL_BaseID
OVA_VAF_TVC_TCL_Version:          VERSION;    !! <-VAF_TVC_TCL_Version
OVA_VAF_TVC_TCL_CKO_Owner:        OWNER;      !! <-VAF_TVC_TCL_CKO_Owner
OVA_VAF_TVC_TCL_CK:               CK5;        !! <-VAF_TVC_TCL_CK
OVA_VAF_TVC_TCL_VRS_Code:         VERSCODE;   !! <-VAF_TVC_TCL_VRS_Code
OVA_VAF_TVC_SCA_VCT_CTK_BaseID:   BASEID;     !! <-VAF_TVC_SCA_VCT_CTK_BaseID
OVA_VAF_TVC_SCA_VCT_CTK_CKO_Owner: OWNER;     !! <-VAF_TVC_SCA_VCT_CTK_CKO_Owner
OVA_VAF_TVC_SCA_VCT_CTT_LNG_Code: LANGCODE;   !! <-VAF_TVC_SCA_VCT_CTT_LNG_Code
OVA_VAF_TVC_SCA_VCT_CTT_TextID:   TEXTID;     !! <-VAF_TVC_SCA_VCT_CTT_TextID
OVA_VAF_Comment:     OPTIONAL     COMMENT;    !! <-VAF_Comment
OVA_VAF_RefDate_A:                DATEDOM;    !! <-VAF_RefDate_A
OVA_VAF_BeginValidity_A:          DATEDOM;    !! <-VAF_BeginValidity_A
OVA_VAF_EndValidity_A: OPTIONAL   DATEDOM;    !!<-VAF_EndValidity_A
OVA_TVA_BASEID:                   BASEID;     !! <-TVA_BASEID
OVA_TVA_VERSION:                  VERSION;    !! <-TVA_VERSION
OVA_TVA_VAF_BASEID:               BASEID;     !! <-TVA_VAF_BASEID
OVA_TVA_VAF_VERSION:              VERSION;    !! <-TVA_VAF_VERSION
OVA_TVA_VAF_TIS_BaseID:           BASEID;     !! <-TVA_VAF_TIS_BaseID
OVA_TVA_VAF_TIS_Version:          VERSION;    !! <-TVA_VAF_TIS_Version
OVA_TVA_VAF_TIS_CKO_Owner:        OWNER;      !! <-TVA_VAF_TIS_CKO_Owner
OVA_TVA_VAF_TIS_CK:               CK5;        !! <-TVA_VAF_TIS_CK
OVA_TVA_VAF_TIS_VRS_Code:         VERSCODE;   !! <-TVA_VAF_TIS_VRS_Code
OVA_TVA_VAF_TVC_BaseID:           BASEID;     !! <-TVA_VAF_TVC_BaseID
```

```
      OVA_TVA_VAF_TVC_Version:          VERSION;    !! <-TVA_VAF_TVC_Version
      OVA_TVA_VAF_TVC_TCL_BaseID:       BASEID;     !! <-TVA_VAF_TVC_TCL_BaseID
      OVA_TVA_VAF_TVC_TCL_Version:      VERSION;    !! <-TVA_VAF_TVC_TCL_Version
      OVA_TVA_VAF_TVC_TCL_CKO_Owner:    OWNER;      !! <-TVA_VAF_TVC_TCL_CKO_Owner
      OVA_TVA_VAF_TVC_TCL_CK:           CK5;        !! <-TVA_VAF_TVC_TCL_CK
      OVA_TVA_VAF_TVC_TCL_VRS_Code:     VERSCODE;   !! <-TVA_VAF_TVC_TCL_VRS_Code
      OVA_TVA_VAF_TVC_SCA_VCT_CTK_BaseID: BASEID;   !! <-TVA_VAF_TVC_SCA_VCT_CTK_BaseID
      OVA_TVA_VAF_TVC_SCA_VCT_CTK_CKO_Owner: OWNER; !! <-TVA_VAF_TVC_SCA_VCT_CTK_CKO_Owner
      OVA_TVA_VAF_TVC_SCA_VCT_CTT_LNG_Code:LANGCODE; !! <-TVA_VAF_TVC_SCA_VCT_CTT_LNG_Code
      OVA_TVA_VAF_TVC_SCA_VCT_CTT_TextID: TEXTID;   !! <-TVA_VAF_TVC_SCA_VCT_CTT_TextID
      OVA_TVA_VAF_BeginValidity_A:      DATEDOM;    !! <-TVA_VAF_BeginValidity_A
      OVA_TVA_VAF_VRS_Code:             VERSCODE;   !! <-TVA_VAF_VRS_Code
      OVA_TVA_TIMESTAMP:                DATEDOM;    !! <-TVA_TIMESTAMP
      OVA_TVA_INDEX:                    NUM4;       !! <-TVA_INDEX
      OVA_TVA_VALUE_EFFECTIF: OPTIONAL  NUM20_10;   !! <-TVA_VALUE_EFFECTIF
      OVA_TVA_ORIENTATION_Code:         ORICODE;    !! <-TVA_ORIENTATION_Code
      OVA_TVA_PLAUSIBILITY_Code:        PLACODE;    !! <-TVA_PLAUSIBILITY_Code
      OVA_TVA_CHANGE_Code               CHACODE;    !! <-TVA_CHANGE_Code
      OVA_TVA_VALUE_THEORETIC: OPTIONAL NUM20_10;   !! <-TVA_VALUE_THEORETIC
      OVA_TVA_FREETEXT:    OPTIONAL     COMMENT;    !! <-TVA_FREETEXT
      OVA_TVA_CODE:        OPTIONAL     TEXT*2;     !! <-TVA_CODE
      OVA_TVA_SCA_CTK_BASEID: OPTIONAL  BASEID;     !! <-TVA_SCA_CTK_BASEID
      OVA_TVA_SCA_VCT_CTK_CKO_Owner:    OWNER;      !! <-TVA_SCA_VCT_CTK_CKO_Owner
      OVA_TVA_SCA_VCT_CTT_LNG_Code:     LANGCODE;   !! <-TVA_SCA_VCT_CTT_LNG_Code
      OVA_TVA_SCA_VCT_CTT_TextID:       TEXTID;     !! <-TVA_SCA_VCT_CTT_TextID
      OVA_TVA_TYPADTEXT:   OPTIONAL     ADTEXT;     !! <-TVA_TYPADTEXT
      OVA_TVA_COMMENT:     OPTIONAL     COMMENT;    !! <-TVA_COMMENT
END Traffic_Values;
```

```
    TABLE Traffic_Values_Abs =
    !! (D:‘Absolute Verkehrswerte‘ F:‘Valeurs de trafic absolues‘)
      TVA_BASEID:                        BASEID;                  !! PK
      TVA_VERSION:                       VERSION;                 !! PK
      TVA_VAF_BASEID:                    BASEID      //Kernel//;  !! FKL(1)->VAF
      TVA_VAF_VERSION:                   VERSION     //Kernel//;  !! FKL(1)->VAF
      TVA_VAF_TIS_BaseID:                BASEID      //Kernel//;  !! FKL(.)->VAF
      TVA_VAF_TIS_Version:               VERSION     //Kernel//;  !! FKL(.)->VAF
      TVA_VAF_TIS_CKO_Owner:             OWNER       //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TIS_CK                     CK5         //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TIS_VRS_Code:              VERSCODE    //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TVC_BaseID:                BASEID      //Kernel//;  !! FKL(1)->VAF
      TVA_VAF_TVC_Version:               VERSION     //Kernel//;  !! FKL(1)->VAF
      TVA_VAF_TVC_TCL_BaseID:            BASEID      //Kernel//;  !! FKL(.)->VAF
      TVA_VAF_TVC_TCL_Version:           VERSION     //Kernel//;  !! FKL(.)->VAF
      TVA_VAF_TVC_TCL_CKO_Owner:         OWNER       //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TVC_TCL_CK:                CK5         //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TVC_TCL_VRS_Code:          VERSCODE    //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TVC_SCA_VCT_CTK_BaseID:    BASEID      //Kernel//;  !! FKL(.)->VAF
      TVA_VAF_TVC_SCA_VCT_CTK_CKO_Owner: OWNER       //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TVC_SCA_VCT_CTT_LNG_Code:LANGCODE      //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_TVC_SCA_VCT_CTT_TextID:    TEXTID      //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_BeginValidity_A:           DATEDOM     //Kernel//;  !! UKC(1)->VAF
      TVA_VAF_VRS_Code:                  VERSCODE    //Kernel//;  !! UKC(1)->VAF
      TVA_TIMESTAMP:                     DATEDOM     //Kernel//;
      TVA_INDEX:                         NUM4        //Kernel//;  !! UKC
      TVA_VALUE_EFFECTIF:    OPTIONAL    NUM20_10    //Kernel//;
      TVA_ORIENTATION_Code:              ORICODE     //Kernel//;
      TVA_PLAUSIBILITY_Code:             PLACODE     //Kernel//;
      TVA_CHANGE_Code:                   CHACODE     //Kernel//;
      TVA_VALUE_THEORETIC:   OPTIONAL    NUM20_10    //Kernel//;
      TVA_FREETEXT:          OPTIONAL    COMMENT     //Kernel//;
      TVA_CODE:              OPTIONAL    TEXT*2      //Kernel//;
      TVA_SCA_CTK_BASEID:    OPTIONAL    BASEID      //Kernel//;  !! FKL(2)->SCA
      TVA_SCA_VCT_CTK_CKO_Owner:         OWNER       //Kernel//;  !! FKC(2)->SCA
      TVA_SCA_VCT_CTT_LNG_Code:          LANGCODE    //Kernel//;  !! FKC(2)->SCA
      TVA_SCA_VCT_CTT_TextID:            TEXTID      //Kernel//;  !! FKC(2)->SCA
      TVA_TYPADTEXT:         OPTIONAL    ADTEXT      //Kernel//;
      TVA_COMMENT:           OPTIONAL    COMMENT     //Kernel//;
      TVA_ODO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
      TVA_OrigSubDBID:                   DBID;                    !!                SN-GEN-A3
      TVA_DAO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
      TVA_DataOwner:                     DATAOWNER;               !!                SN-GEN-A5
      TVA_CreateDate:                    DATEDOM;                 !!                SN-GEN-A8
      TVA_ChangeDate:      OPTIONAL      DATEDOM;                 !!                SN-GEN-A9
      TVA_CHO_Owner:                     OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
      TVA_ChangeUser:                    ORAUSER;                 !!                SN-GEN-A11
      TVA_ITS_Code:                      ITSCODE;                 !!                SN-GEN-A6
      TVA_IntegrityDate:                 DATEDOM;                 !!                SN-GEN-A7
      TVA_XST_XfrSetID:    OPTIONAL      BASEID;                  !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
        TVA_BASEID, TVA_VERSION;
        TVA_VAF_BASEID, TVA_VAF_VERSION, TVA_TIMESTAMP;
        TVA_VAF_BASEID, TVA_VAF_VERSION, TVA_INDEX;
    END Absolute_Traffic_Values;

!!=============================================================================================
!! Attribut description
!!    TVA_BASEID:             Primary Key
!!    TVA_VERSION:            Primary Key
!!    TVA_VAF_BASEID:         Logical foreign Key to Traffic Variation Curve
!!    TVA_VAF_VERSION:        Version on the traffic variation curve
!!    TVA_VAF_TIS_CKO_Owner, TVA_VAF_TIS_CK, TVA_VAF_TIS_VRS_Code
!!    TVA_VAF_TVC_TCL_CKO_Owner, TVA_VAF_TVC_TCL_CK, TVA_VAF_TVC_TCL_VRS_Code,
!!    TVA_VAF_TVC_SCA_VCT_CTK_CKO_Owner, TVA_VAF_TVC_SCA_VCT_CTT_LNG_Code,
!!    TVA_VAF_TVC_SCA_VCT_CTT_TextID, TVA_VAF_BeginValidity_A
!!    TVA_VRS_Code:
!!                            Necessary attributs to identify a Traffic Value
!!                            Classification
!!    TVA_TIMESTAMP:          Date of measurement
!!    TVA_INDEX:              Sequence number of the measure within a set of measures
!!    TVA_VALUE_EFFECTIF:     Effective absolute value
!!    TVA_ORIENTATION_CODE:   Orientation of the measurement
!!    TVA_PLAUSIBILITY_CODE:  Plausibility for user of system
!!    TVA_CHANGE_CODE:        Modification Method
!!    TVA_VALUE_THEORETIC:    Theoretical Value
!!    TVA_FREETEXT:           Comment
!!    TVA_CODE:               Free comment
!!    TVA_SCA_CTK_BASEID:     Free text catalog
```

```
!!    TVA_SCA_CTK_CKO_Owner, TVA_SCA_CTT_LNG_Code, TVA_SCA_CTT_TextID:
!!                            Conceptual Key of text of a text catalog
!!    TVA_TYPADTEXT:          Free text to text catalog
!!=============================================================================================
```

```
    TABLE Traf_values_rels =
      TVR_BASEID:                          BASEID;                  !! PK
      TVR_VERSION:                         VERSION;                 !! PK
      TVR_VAF_BASEID:                      BASEID      //Kernel//;  !! FKL(1)->VAF
      TVR_VAF_VERSION:                     VERSION     //Kernel//;  !! FKL(1)->VAF
      TVR_VAF_TIS_BaseID:                  BASEID      //Kernel//;  !! FKL(.)->VAF
      TVR_VAF_TIS_Version:                 VERSION     //Kernel//;  !! FKL(.)->VAF
      TVR_VAF_TIS_CKO_Owner:               OWNER       //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TIS_CK                       CK5         //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TIS_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TVC_BaseID:                  BASEID      //Kernel//;  !! FKL(1)->VAF
      TVR_VAF_TVC_Version:                 VERSION     //Kernel//;  !! FKL(1)->VAF
      TVR_VAF_TVC_TCL_BaseID:              BASEID      //Kernel//;  !! FKL(.)->VAF
      TVR_VAF_TVC_TCL_Version:             VERSION     //Kernel//;  !! FKL(.)->VAF
      TVR_VAF_TVC_TCL_CKO_Owner:           OWNER       //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TVC_TCL_CK:                  CK5         //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TVC_TCL_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TVC_SCA_VCT_CTK_BaseID:      BASEID      //Kernel//;  !! FKL(.)->VAF
      TVR_VAF_TVC_SCA_VCT_CTK_CKO_Owner: OWNER         //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TVC_SCA_VCT_CTT_LNG_Code:LANGCODE        //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_TVC_SCA_VCT_CTT_TextID:      TEXTID      //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_BeginValidity_A:             DATEDOM     //Kernel//;  !! UKC(1)->VAF
      TVR_VAF_VRS_Code:                    VERSCODE    //Kernel//;  !! UKC(1)->VAF
      TVR_INDEX:                           NUM4        //Kernel//;  !! UKC
      TVR_VALUE_EFFECTIF:    OPTIONAL      NUM20_10    //Kernel//;
      TVR_ORIENTATION_Code:                ORICODE     //Kernel//;
      TVR_PLAUSIBILITY_Code:               PLACODE     //Kernel//;
      TVR_CHANGE_Code:                     CHACODE     //Kernel//;
      TVR_VALUE_THEORETIC:   OPTIONAL      NUM20_10    //Kernel//;
      TVR_FREETEXT:          OPTIONAL      COMMENT     //Kernel//;
      TVR_CODE:              OPTIONAL      TEXT*2      //Kernel//;
      TVR_SCA_CTK_BASEID:    OPTIONAL      BASEID      //Kernel//;  !! FKL(2)->SCA
      TVR_SCA_VCT_CTK_CKO_Owner:           OWNER       //Kernel//;  !! FKC(2)->SCA
      TVR_SCA_VCT_CTT_LNG_Code:            LANGCODE    //Kernel//;  !! FKC(2)->SCA
      TVR_SCA_VCT_CTT_TextID:              TEXTID      //Kernel//;  !! FKC(2)->SCA
      TVR_TYPADTEXT:         OPTIONAL      ADTEXT      //Kernel//;
      TVR_COMMENT:           OPTIONAL      COMMENT     //Kernel//;
      TVR_ODO_Owner:                       OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
      TVR_OrigSubDBID:                     DBID;                    !!                SN-GEN-A3
      TVR_DAO_Owner:                       OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
      TVR_DataOwner:                       DATAOWNER;               !!                SN-GEN-A5
      TVR_CreateDate:                      DATEDOM;                 !!                SN-GEN-A8
      TVR_ChangeDate:      OPTIONAL        DATEDOM;                 !!                SN-GEN-A9
      TVR_CHO_Owner:                       OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
      TVR_ChangeUser:                      ORAUSER;                 !!                SN-GEN-A11
      TVR_ITS_Code:                        ITSCODE;                 !!                SN-GEN-A6
      TVR_IntegrityDate:                   DATEDOM;                 !!                SN-GEN-A7
      TVR_XST_XfrSetID:    OPTIONAL        BASEID;                  !! FKL(.)->XST

    CONSTRAINT
      UNIQUE
        TVR_BASEID, TVR_VERSION;
        TVR_VAF_BASEID, TVR_VAF_VERSION, TVR_INDEX;
    END traf_values_rels;

!!=============================================================================================
!! Attribut description
!!    TVR_BASEID:              Primary Key
!!    TVR_VERSION:             Primary Key
!!    TVR_TVA_BASEID           Logical foreign Key to Traf values abs.
!!    TVR_VAF_BASEID:          Logical foreign Key to Traffic Variation Curve
!!    TVR_VAF_VERSION:         Version on the traffic variation curve
!!    TVR_VAF_TIS_CKO_Owner, TVR_VAF_TIS_CK, TVR_VAF_TIS_VRS_Code
!!    TVR_VAF_TVC_TCL_CKO_Owner, TVR_VAF_TVC_TCL_CK, TVR_VAF_TVC_TCL_VRS_Code,
!!    TVR_VAF_TVC_SCA_VCT_CTK_CKO_Owner, TVR_VAF_TVC_SCA_VCT_CTT_LNG_Code,
!!    TVR_VAF_TVC_SCA_VCT_CTT_TextID, TVR_VAF_BeginValidity_A
!!    TVR_VRS_Code:
!!                             Necessary attributs to identify a Traffic Value
!!                             Classification
!!    TVR_INDEX:               Sequence number of the measure within a set of measures
!!    TVR_VALUE_EFFECTIF:      Effective absolute value
!!    TVR_ORIENTATION_CODE:    Orientation of the measurement
!!    TVR_PLAUSIBILITY_CODE:   Plausibility for user of system
!!    TVR_CHANGE_CODE:         Modification Method
!!    TVR_VALUE_THEORETIC:     Theoretical Value
!!    TVR_FREETEXT:            Comment
!!    TVR_CODE:                Free comment
!!    TVR_SCA_CTK_BASEID:      Free text catalog
!!    TVR_SCA_CTK_CKO_Owner, TVR_SCA_CTT_LNG_Code, TVR_SCA_CTT_TextID:
!!                             Conceptual Key of text of a text catalog
```

```
!!    TVR_TYPADTEXT:          Free text to text catalog
!!=============================================================================
```

```
TABLE Doc_Usages =
!! (D: Dokument Nutzung, F: Usage des documents)
  DUS_ObjBaseID:                    MSK         //Kernel//;   !! PK
  DUS_ObjTypeName:                  TEXT*20;
  DUS_Doc_BaseID:                   MSK         //Kernel//;   !! PK
  DUS_Doc_Version:                  VERS        //Kernel//;   !! PK
  DUS_ODO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A2
  DUS_OrigSubDBID:                  DBID;                     !!                  SN-GEN-A3
  DUS_DAO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A4
  DUS_DataOwner:                    DATAOWNER;                !!                  SN-GEN-A5
  DUS_CreateDate:                   DATEDOM;                  !!                  SN-GEN-A8
  DUS_ChangeDate:      OPTIONAL     DATEDOM;                  !!                  SN-GEN-A9
  DUS_CHO_Owner:                    OWNER;                    !! FKL(.)->OWN    SN-GEN-A10
  DUS_ChangeUser:                   ORAUSER;                  !!                  SN-GEN-A11
  DUS_ITS_Code:                     ITSCODE;                  !!                  SN-GEN-A6
  DUS_IntegrityDate:                DATEDOM;                  !!                  SN-GEN-A7
  DUS_XST_XfrSetID:    OPTIONAL     BASEID;                   !! FKL(.)->XST
CONSTRAINT
  UNIQUE
    DUS_ObjBaseID, DUS_Doc_BaseID, DUS_Doc_Version; !!PK
END Doc_Usage;

TABLE SimpleSubjUsages =
!! Simplified Subject Usage connected with a STRADA object
!! (Without administration data which are inherited from the object with OBJ_BASEID)
  SSU_SBJ_BASEID:                   BASEID      //Kernel//;   !! PK
  SSU_OBJ_BASEID:                   BASEID      //Kernel//;   !! PK
  SSU_OBJ_TYPNAME:     OPTIONAL     TEXT*20     //Kernel//;
CONSTRAINT
  UNIQUE
    SBJ_BASEID, OBJ_BASEID; !! PK
END SimpleSubjUsage;
```

```
TABLE Segmentation_PMS =
!! (D:‘PMS-Segmentierung ‘ F:‘Segmentation PMS‘, Guide STRADA-PMS /GEN-SN640940)
  SEG_Baseid:                       BASEID;                 !! PK             SN-GEN-I1
  SEG_Version:                      VERSION;                !! PK             SN-GEN-I2
  SEG_CKO_Owner:                    OWNER       //Kernel//; !! UKC->OWN       GI-PMS-I1
  SEG_CK:                           TEXT*5      //Kernel//; !! UKC            GI-PMS-I2
  SEG_Statut_Code:                  STCODE      //Kernel//; !!                GI-PMS-A01
  SEG_Name:           OPTIONAL      ADTEXT      //Kernel//; !!                GI-PMS-A02
  SEG_VRS_Code:                     VERSCODE    //Kernel//; !!                SN-GEN-A1
  SEG_RefDate:                      DATEDOM     //Kernel//; !!                SN-GEN-D3
  SEG_BeginValidity:                DATEDOM     //Kernel//; !!                SN-GEN-D1
  SEG_EndValidity:    OPTIONAL      DATEDOM     //Kernel//; !!                SN-GEN-D2
  SEG_Comment:        OPTIONAL      COMMENT     //Kernel//;
  SEG_ODO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
  SEG_OrigSubDBID:                  DBID;                   !!                SN-GEN-A3
  SEG_DAO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
  SEG_DataOwner:                    DATAOWNER;              !!                SN-GEN-A5
  SEG_CreateDate:                   DATEDOM;                !!                SN-GEN-A8
  SEG_ChangeDate:     OPTIONAL      DATEDOM;                !!                SN-GEN-A9
  SEG_CHO_Owner:                    OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
  SEG_ChangeUser:                   ORAUSER;                !!                SN-GEN-A11
  SEG_ITS_Code:                     ITSCODE;                !!                SN-GEN-A6
  SEG_IntegrityDate:                DATEDOM;                !!                SN-GEN-A7
  SEG_XST_XfrSetID:   OPTIONAL      BASEID;                 !! FKL(.)->XST
CONSTRAINT
  SEG_BeginValidity < SEG_EndValidity;
   !! The PMS Segmentation describes an activity. Therefore, no more than one version
   !! exists.
  (IF SEG_RefDate < SEG_BeginValidity THEN SEG_VRS_Code == P ELSE SEG_VRS_Code == E);
   !! The PMS Segmentation describes an activity. Therefore, the EndValidity has to be
   !! defined.
  UNIQUE
    SEG_BaseID, SEG_Version; !! PK
  UNIQUE
    SEG_CKO_OWNER, SEG_CK; !! UKC
END Segmentation_PMS;
```

```
TABLE Decoupage_PMS =
!! (D:‘ PMS-Objekt-Bildung‘ F:‘Découpage PMS‘, Guide STRADA-PMS /GEN-SN640940 )
  DEC_Baseid:                       BASEID;                  !! PK             SN-GEN_I1
  DEC_Version:                      VERSION;                 !! PK             SN-GEN-I2
  DEC_SEG_Baseid:                   BASEID;     //Kernel//;  !! FKL(1)->SEG    GI-PMS-I1.0
  DEC_SEG_CKO_Owner:                OWNER       //Kernel//;  !! UKC(1)->SEG    GI-PMS-I1.1
  DEC_SEG_CK:                       TEXT*5      //Kernel//;  !! UKC(1)->SEG    GI-PMS-I1.2
  DEC_SEG_VRS_Code:                 VERSCODE    //Kernel//;  !! UKC(1)->SEG    GI-PMS-I1.3
  DEC_CKO_Owner:                    OWNER       //Kernel//;  !! UKC(.)->OWN    GI-PMS-I2
  DEC_CK:                           TEXT*5      //Kernel//;  !! UKC            GI-PMS-I3
  DEC_Decoupage_Code:               DECCODE     //Kernel//;  !!                GI-PMS-A01
  DEC_Statetat_Code:                DECSTCODE   //Kernel//;  !!                GI-PMS-A02
  DEC_Nom:             OPTIONAL     ADTEXT      //Kernel//;  !!                GI-PMS-A03
  DEC_CondMinLong:     OPTIONAL     PNUM5_1     //Kernel//;  !!                GI-PMS-A04
  DEC_CondMaxLong:     OPTIONAL     PNUM5_1     //Kernel//;  !!                GI-PMS-A05
  DEC_ConditionAge     OPTIONAL     PNUM4       //Kernel//;  !!                GI-PMS-A06
  DEC_DureeAge         OPTIONAL     PNUM2       //Kernel//;  !!                GI-PMS-A07
  DEC_LimiteIndice     OPTIONAL     PNUM1_1     //Kernel//;  !!                GI-PMS-A08
  DEC_StatCoher_Code   OPTIONAL     DECSTCOH    //Kernel//;  !!                GI-PMS-A09
  DEC_VRS_Code:                     VERSCODE    //Kernel//;  !!                SN-GEN-A1
  DEC_RefDate:                      DATEDOM     //Kernel//;  !!                SN-GEN-D3
  DEC_BeginValidity:                DATEDOM     //Kernel//;  !!                SN-GEN-D1
  DEC_EndValidity:     OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
  DEC_Comment:         OPTIONAL     COMMENT     //Kernel//;
  DEC_ODO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
  DEC_OrigSubDBID:                  DBID;                    !!                SN-GEN-A3
  DEC_DAO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
  DEC_DataOwner:                    DATAOWNER;               !!                SN-GEN-A5
  DEC_CreateDate:                   DATEDOM;                 !!                SN-GEN-A8
  DEC_ChangeDate:      OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
  DEC_CHO_Owner:                    OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
  DEC_ChangeUser:                   ORAUSER;                 !!                SN-GEN-A11
  DEC_ITS_Code:                     ITSCODE;                 !!                SN-GEN-A6
  DEC_IntegrityDate:                DATEDOM;                 !!                SN-GEN-A7
  DEC_XST_XfrSetID:    OPTIONAL     BASEID;                  !! FKL(.)->XST
CONSTRAINT
  DEC_Version != 1;
   !! The 'PMS decoupage' describes an activity. Therefore, no more than one version
   !! exists.
  DEC_EndValidity IS DEFINED;
   !! The 'PMS decoupage' describes an activity. Therefore, the EndValidity has to be
   !! defined.
  DEC_BeginValidity < DEC_EndValidity;
   !! The BeginValidity has to be earlier than the EndValidity.
  (IF DEC_RefDate < DEC_BeginValidity THEN DEC_VRS_Code == P ELSE DEC_VRS_Code == E);
   !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
  (IF DEC_VRS_Code == E THEN DEC_SEG_VRS_Code == E);
   !! If the treated 'PMS decoupage' is an effective object, his master object has to be
   !! also effective.
  DEC_CondMinLongueur < DEC_CondMaxLongueur;
   !! The minimum length of the attached PMS-objects has to be smaller than the maximum
   !! length.
  UNIQUE
    DEC_BaseID, DEC_Version; !! PK
  UNIQUE
    DEC_SEG_CKO_Owner, DEC_SEG_CK, DEC_SEG_VRS_Code, DEC_CKO_Owner, DEC_CK; !! UKC
END Decoupage_PMS;
```

```
TABLE Segment_Chaussee =
!! (D:‘Fahrbahnsegment‘, F:‘Segment de chaussée‘, Guide STRADA-PMS /GEN-SN640940 )
   SCH_Baseid:                      BASEID;                  !! PK             SN-GEN-I1
   SCH_Version:                     VERSION;                 !! PK             SN-GEN-I2
   SCH_SEG_Baseid:                  BASEID;     //Kernel//;  !! FKL(1)->SEG    GI-PMS-I1.0
   SCH_SEG_CKO_Owner:               OWNER       //Kernel//;  !! UKC(1)->SEG    GI-PMS-I1.1
   SCH_SEG_CK:                      TEXT*5      //Kernel//;  !! UKC(1)->SEG    GI-PMS-I1.2
   SCH_SEG_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(1)->SEG    GI-PMS-I1.3
   SCH_CK:                          TEXT*8      //Kernel//;  !! UKC            GI-PMS-I2
   SCH_AXE_CKO_Owner:               OWNER       //Kernel//;  !! UKC(2,3)->RPT  GI-PMS-A01.11
   SCH_AXE_CK:                      CK3         //Kernel//;  !! UKC(2,3)->RPT  GI-PMS-A01.12
   SCH_AXE_POS_Code:                POSCODE     //Kernel//;  !! UKC(2,3)->RPT  GI-PMS-A01.13
   SCH_AXE_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(2,3)->RPT  GI-PMS-A01.14
   SCH_RPT_BaseID_1:                BASEID      //Kernel//;  !! FKL(2)->RPT    GI-PMS-A01.20
   SCH_RPT_CK_1:                    CK2         //Kernel//;  !! UKC(2)->RPT    GI-PMS-A01.21
   SCH_RPT_VRS_Code_1:              VERSCODE    //Kernel//;  !! UKC(2)->RPT    GI-PMS-A01.22
   SCH_RPDistBegin:                 U_ABS       //Kernel//;  !!                GI-PMS-A01.3
   SCH_LatDistBegin:                NUM2_3;     //Kernel//;  !!                GI-PMS-A01.4
   SCH_RPT_BaseID_2:                BASEID      //Kernel//;  !! FKL(3)->RPT    GI-PMS-A02.10
   SCH_RPT_CK_2:                    CK2         //Kernel//;  !! UKC(3)->RPT    GI-PMS-A02.11
   SCH_RPT_VRS_Code_2:              VERSCODE    //Kernel//;  !! UKC(3)->RPT    GI-PMS-A02.12
   SCH_RPDistEnd:                   U_ABS       //Kernel//;  !!                GI-PMS-A02.2
   SCH_LatDistEnd:                  NUM2_3;     //Kernel//;  !!                GI-PMS-A02.3
   SCH_PosLatMoy:                   NUM2_2;     //Kernel//;  !!                GI-PMS-A03
   SCH_LargeurSeg:                  NUM3_2;     //Kernel//;  !!                GI-PMS-A04
   SCH_LKG_BaseID_1:   OPTIONAL     BASEID      //Kernel//;  !! FKL(5)->LKG    GI-PMS-A05.0
   SCH_LKG_CKO_Owner_1: OPTIONAL    OWNER       //Kernel//;  !! FKC(5)->LKG    GI-PMS-A05.1
   SCH_LKG_CK_1:       OPTIONAL     CK2         //Kernel//;  !! FKC(5)->LKG    GI-PMS-A05.2
   SCH_LKG_VRS_Code_1: OPTIONAL     VERSCODE    //Kernel//;  !! FKC(5)->LKG    GI-PMS-A05.3
   SCH_LKG_BaseID_2:   OPTIONAL     BASEID      //Kernel//;  !! FKL(6)->LKG    GI-PMS-A06.0
   SCH_LKG_CKO_Owner_2: OPTIONAL    OWNER       //Kernel//;  !! FKC(6)->LKG    GI-PMS-A06.1
   SCH_LKG_CK_2:       OPTIONAL     CK2         //Kernel//;  !! FKC(6)->LKG    GI-PMS-A06.2
   SCH_LKG_VRS_Code_2: OPTIONAL     VERSCODE    //Kernel//;  !! FKC(6)->LKG    GI-PMS-A06.3
   SCH_FacteurFonc:    OPTIONAL     NUM2_3      //Kernel//;  !!                GI-PMS-A07
   SCH_TJM:            OPTIONAL     NUM6        //Kernel//;  !!                GI-PMS-A08
   SCH_PCTPL:          OPTIONAL     NUM3_2      //Kernel//;  !!                GI-PMS-A09
   SCH_TraficCumule:   OPTIONAL     NUM8        //Kernel//;  !!                GI-PMS-A10
   SCH_MultEssieux:    OPTIONAL     NUM5        //Kernel//;  !!                GI-PMS-A11
   SCH_RepartDirec:    OPTIONAL     NUM5        //Kernel//;  !!                GI-PMS-A12
   SCH_TauxAccident:   OPTIONAL     NUM2_2      //Kernel//;  !!                GI-PMS-A13
   SCH_ClasseClim_Code:OPTIONAL     SCHCCLCD    //Kernel//;  !!                GI-PMS-A14
   SCH_SituSeg_Code:   OPTIONAL     SITUCODE    //Kernel//;  !!                GI-PMS-A15
   SCH_TypeSeg_Code:   OPTIONAL     TYPCHCODE   //Kernel//;  !!                GI-PMS-A16
   SCH_PosChauss_Code: OPTIONAL     POSCHCODE   //Kernel//;  !!                GI-PMS-A17
   SCH_TypeStruc_Code: OPTIONAL     TYSTRCODE   //Kernel//;  !!                GI-PMS-A18
   SCH_ClasseTrafic_Code:OPTIONAL   CLTRFCODE   //Kernel//;  !!                GI-PMS-A19
   SCH_DebClasseAge:   OPTIONAL     NUM4        //Kernel//;  !!                GI-PMS-A20
   SCH_FinClasseAge:   OPTIONAL     NUM4        //Kernel//;  !!                GI-PMS-A21
   SCH_Drainage:       OPTIONAL     OUINON      //Kernel//;  !!                GI-PMS-A22
   SCH_DimGel:         OPTIONAL     OUINON      //Kernel//;  !!                GI-PMS-A23
   SCH_LigneTC:        OPTIONAL     OUINON      //Kernel//;  !!                GI-PMS-A24
   SCH_DerivAlt_Code:  OPTIONAL     DERIVCODE   //Kernel//;  !!                GI-PMS-A25
   SCH_ValUnitRempl:   OPTIONAL     NUM6_1      //Kernel//;  !!                GI-PMS-A26
   SCH_PCTCoutMaint:   OPTIONAL     NUM3_2      //Kernel//;  !!                GI-PMS-A27
   SCH_VRS_Code:                    VERSCODE    //Kernel//;  !!                SN-GEN-A1
   SCH_RefDate:                     DATEDOM     //Kernel//;  !!                SN-GEN-D3
   SCH_BeginValidity:               DATEDOM     //Kernel//;  !!                SN-GEN-D1
   SCH_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
   SCH_Comment:        OPTIONAL     COMMENT     //Kernel//;
   SCH_ODO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
   SCH_OrigSubDBID:                 DBID;                    !!                SN-GEN-A3
   SCH_DAO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
   SCH_DataOwner:                   DATAOWNER;               !!                SN-GEN-A5
   SCH_CreateDate:                  DATEDOM;                 !!                SN-GEN-A8
   SCH_ChangeDate:     OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
   SCH_CHO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
   SCH_ChangeUser:                  ORAUSER;                 !!                SN-GEN-A11
   SCH_ITS_Code:                    ITSCODE;                 !!                SN-GEN-A6
   SCH_IntegrityDate:               DATEDOM;                 !!                SN-GEN-A7
   SCH_XST_XfrSetID:   OPTIONAL     BASEID;                  !! FKL(.)->XST
CONSTRAINT
  SCH_Version != 1;
   !! The PMS segment is an activity. Therefore, no more than one version exists.
  SCH_EndValidity IS DEFINED;
   !! The PMS segment is an activity. Therefore, the EndValidity has to be defined.
  SCH_BeginValidity < SCH_EndValidity;
   !! The BeginValidity has to be earlier than the EndValidity.
  (IF SCH_RefDate < SCH_BeginValidity THEN SCH_VRS_Code == P ELSE SCH_VRS_Code == E);
   !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
```

```
      (IF SCH_VRS_Code == E THEN SCH_SEG_VRS_Code == E AND SCH_AXE_VRS_Code == E AND SCH_RPT_VRS_Code_1
== E AND SCH_RPT_VRS_Code_2 == E AND SCH_LKG_VRS_Code_1 == E AND SCH_LKG_VRS_Code_2 == E);
       !! If the treated PMS segment is an effective object, all his master objects have to be
       !! also effective.
      SCH_RPT_DistBegin < SCH_RPT_1.RPT_SectorL;
       !! The distance from the reference point to the location on the road axis has to be
       !! smaller than the corresponding sector length
      SCH_RPT_DistEnd < SCH_RPT_2.RPT_SectorL;
       !! The distance from the reference point to the location on the road axis has to be
       !! smaller than the corresponding sector length
      SCH_RPT_1.RPT_AXE_BaseID == SCH_AXE.AXE_BaseID;
       !! The first reference point has to be on the indicated axis.
      SCH_RPT_2.RPT_AXE_BaseID == SCH_AXE.AXE_BaseID;
       !! The second reference point has to be on the indicated axis.
      SCH_RPT_CK_1 < SCH_RPT_CK_2;
       !! The second reference point has to be after the first reference point, seen from the
       !! direction of the indicated axis.
      SCH_LKG_1.LKG_SCA_SLT_CTK_CKO_Owner ==   AND LKG_SCA_SLT_CTT_TextID ==
       !! Reference to a link groupe representing a classification defined by a canton
      SCH_LKG_2.LKG_SCA_SLT_CTK_CKO_Owner ==   AND LKG_SCA_SLT_CTT_TextID ==
       !! Reference to a link groupe representing a administrative subdivision for road
       !! maintenance
      SCH_PCT_PL < 100.00;
       !! Percentage of heavy trucks cannot pass the 100 hundred percent threshold.
      SCH_PCTCoutMaint < 100.00;
       !! Percentage of cost of maintenance work relative to a total subsitution of the PMS
       !! segment.
      UNIQUE
        SCH_BaseID, SCH_Version; !! PK
      UNIQUE
        SCH_SEG_CKO_Owner, SCH_SEG_CK, SCH_SEG_VRS_Code, SCH_CKO_OWNER, SCH_CK; !! UKC
    END Segment_Chaussee;
```

```
TABLE Objet_PMS =
!! (D:‘PMS-Objekt‘, F:‘Objet PMS‘, Guide STRADA-PMS /GEN-SN640940 )
   OBJ_Baseid:                      BASEID;                  !! PK             SN-GEN-I1
   OBJ_Version:                     VERSION;                 !! PK             SN-GEN-I2
   OBJ_DEC_Baseid:                  BASEID;                  !! UKC(1)->DEC    GI-PMS-I1.0
   OBJ_DEC_SEG_CKO_Owner:           OWNER       //Kernel//;  !! UKC(1)->DEC    GI-PMS-I1.11
   OBJ_DEC_SEG_CK:                  TEXT*5      //Kernel//;  !! UKC(1)->DEC    GI-PMS-I1.12
   OBJ_DEC_SEG_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->DEC    GI-PMS-I1.13
   OBJ_DEC_CKO_Owner:               OWNER       //Kernel//;  !! UKC(1)->DEC    GI-PMS-I1.2
   OBJ_DEC_CK:                      TEXT*5      //Kernel//;  !! UKC(1)->DEC    GI-PMS-I1.3
   OBJ_DEC_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(1)->DEC    GI-PMS-I1.4
   OBJ_SCH_Baseid:                  BASEID;     //Kernel//;  !! FKL(2)->SCH    GI-PMS-I2.0
   OBJ_SCH_CK:                      TEXT*8      //Kernel//;  !! UKC(2)->SCH    GI-PMS-I2.1
   OBJ_No:                          TEXT*3      //Kernel//;  !! UKC            GI-PMS-I3
   OBJ_Nom;                         TEXT*10     //Kernel//;  !!                GI-PMS-A01
   OBJ_AXE_CKO_Owner:               OWNER       //Kernel//;  !! FKC(3,4)->RPT  GI-PMS-A02.11
   OBJ_AXE_CK:                      CK3         //Kernel//;  !! FKC(3,4)->RPT  GI-PMS-A02.12
   OBJ_AXE_POS_Code:                POSCODE     //Kernel//;  !! FKC(3,4)->RPT  GI-PMS-A02.13
   OBJ_AXE_VRS_Code:                VERSCODE    //Kernel//;  !! FKC(3,4)->RPT  GI-PMS-A02.14
   OBJ_RPT_BaseID_1:                BASEID      //Kernel//;  !! FKL(3)->RPT    GI-PMS-A02.20
   OBJ_RPT_CK_1:                    CK2         //Kernel//;  !! FKC(3)->RPT    GI-PMS-A02.21
   OBJ_RPT_VRS_Code_1:              VERSCODE    //Kernel//;  !! FKC(3)->RPT    GI-PMS-A02.22
   OBJ_RPDistBegin:                 U_ABS       //Kernel//;  !!                GI-PMS-A02.3
   OBJ_LatDistBegin:                NUM2_3;     //Kernel//;  !!                GI-PMS-A02.4
   OBJ_RPT_BaseID_2:                BASEID      //Kernel//;  !! FKL(4)->RPT    GI-PMS-A03.10
   OBJ_RPT_CK_2:                    CK2         //Kernel//;  !! FKC(4)->RPT    GI-PMS-A03.11
   OBJ_RPT_VRS_Code_2:              VERSCODE    //Kernel//;  !! FKC(4)->RPT    GI-PMS-A03.12
   OBJ_RPDistEnd:                   U_ABS       //Kernel//;  !!                GI-PMS-A03.2
   OBJ_LatDistEnd:                  NUM2_3;     //Kernel//;  !!                GI-PMS-A03.3
   OBJ_WidthBegin:                  NUM2_3;     //Kernel//;  !!                GI-PMS-A04
   OBJ_WidthEnd:                    NUM2_3;     //Kernel//;  !!                GI-PMS-A05
   OBJ_RPT_BaseID_3:                BASEID      //Kernel//;  !! FKL(6)->RPT    GI-PMS-A06.10
   OBJ_RPT_CK_3:                    CK2         //Kernel//;  !! FKC(6)->RPT    GI-PMS-A06.11
   OBJ_RPT_VRS_Code_3:              VERSCODE    //Kernel//;  !! FKC(6)->RPT    GI-PMS-A06.12
   OBJ_RPDistBegZone:               U_ABS       //Kernel//;  !!                GI-PMS-A06.2
   OBJ_LatDistBegZone:              NUM2_3;     //Kernel//;  !!                GI-PMS-A06.3
   OBJ_RPT_BaseID_4:                BASEID      //Kernel//;  !! FKL(7)->RPT    GI-PMS-A07.10
   OBJ_RPT_CK_4:                    CK2         //Kernel//;  !! FKC(7)->RPT    GI-PMS-A07.11
   OBJ_RPT_VRS_Code_4:              VERSCODE    //Kernel//;  !! FKC(7)->RPT    GI-PMS-A07.12
   OBJ_RPDistEndZone:               U_ABS       //Kernel//;  !!                GI-PMS-A07.2
   OBJ_LatDistEndZone:              PNUM2_3;    //Kernel//;  !!                GI-PMS-A07.3
   OBJ_LargeurZone:                 PNUM2_3;    //Kernel//;  !!                GI-PMS-A08
   OBJ_ChantierProp:   OPTIONAL     TEXT*4      //Kernel//;  !!                GI-PMS-A12.1
   OBJ_ChantierNo:     OPTIONAL     PNUM4       //Kernel//;  !!                GI-PMS-A12.2
   OBJ_Priorite:       OPTIONAL     PNUM4_2     //Kernel//;  !!                GI-PMS-A09
   OBJ_AnneePose:      OPTIONAL     PNUM4       //Kernel//;  !!                GI-PMS-A10
   OBJ_EpaisseRevet:   OPTIONAL     PNUM2_3     //Kernel//;  !!                GI-PMS-A11
   OBJ_IndiceAccident: OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A13
   OBJ_TauxAccident:   OPTIONAL     PNUM2_2     //Kernel//;
   OBJ_TJM:            OPTIONAL     PNUM6       //Kernel//;  !!                GI-PMS-A14
   OBJ_PCTPL:          OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A15
   OBJ_TraficCumule:   OPTIONAL     PNUM8       //Kernel//;  !!                GI-PMS-A16
   OBJ_MultEssieux:    OPTIONAL     PNUM5       //Kernel//;  !!                GI-PMS-A17
   OBJ_RepartDirec:    OPTIONAL     PNUM5       //Kernel//;  !!                GI-PMS-A18
   OBJ_SituSeg_Code:   OPTIONAL     SITUCODE    //Kernel//;  !!                GI-PMS-A19
   OBJ_TypeObjet_Code: OPTIONAL     TYPCHCODE   //Kernel//;  !!                GI-PMS-A20
   OBJ_PosChauss_Code: OPTIONAL     POSCHCODE   //Kernel//;  !!                GI-PMS-A21
   OBJ_Drainage:       OPTIONAL     OUINON      //Kernel//;  !!                GI-PMS-A22
   OBJ_DimGel:         OPTIONAL     OUINON      //Kernel//;  !!                GI-PMS-A23
   OBJ_TypeStruc_Code: OPTIONAL     TYSTRCODE   //Kernel//;  !!                GI-PMS-A24
   OBJ_SCA_PLT_CTK_BaseID: OPTIONAL BASEID      //Kernel//;  !! FKL(9)->SCA    GI-PMS-A25.0
   OBJ_SCA_PLT_CTK_CKO_Owner: OPTIONAL OWNER    //Kernel//;  !! FKC(9)->SCA    GI-PMS-A25.1
   OBJ_SCA_PLT_CTT_LNG_Code: OPTIONAL LANGCODE  //Kernel//;  !! FKC(9)->SCA    GI-PMS-A25.2
   OBJ_SCA_PLT_CTT_TextID: OPTIONAL TEXTID      //Kernel//;  !! FKC(9)->SCA    GI-PMS-A25.3
   OBJ_TypAdText:      OPTIONAL     ADTEXT      //Kernel//;
   OBJ_AppreciaResp:   OPTIONAL     PNUM3       //Kernel//;  !!                GI-PMS-A26
   OBJ_VRS_Code:                    VERSCODE    //Kernel//;  !!                SN-GEN-A1
   OBJ_RefDate:                     DATEDOM     //Kernel//;  !!                SN-GEN-D3
   OBJ_BeginValidity:               DATEDOM     //Kernel//;  !!                SN-GEN-D1
   OBJ_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
   OBJ_Comment:        OPTIONAL     COMMENT     //Kernel//;
   OBJ_ODO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
   OBJ_OrigSubDBID:                 DBID;                    !!                SN-GEN-A3
   OBJ_DAO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
   OBJ_DataOwner:                   DATAOWNER;               !!                SN-GEN-A5
   OBJ_CreateDate:                  DATEDOM;                 !!                SN-GEN-A8
   OBJ_ChangeDate:     OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
   OBJ_CHO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
   OBJ_ChangeUser:                  ORAUSER;                 !!                SN-GEN-A11
```

```
       OBJ_ITS_Code:                   ITSCODE;                !!               SN-GEN-A6
       OBJ_IntegrityDate:              DATEDOM;                !!               SN-GEN-A7
       OBJ_XST_XfrSetID:   OPTIONAL    BASEID;                 !! FKL(.)->XST
    CONSTRAINT
      OBJ_Version != 1;
 !! The PMS-Object describes an activity. Therefore, no more than one version exists.
      OBJ_EndValidity IS DEFINED;
       !! The PMS-Object describes an activity. Therefore, the EndValidity has to be defined.
      OBJ_BeginValidity < OBJ_EndValidity;
       !! The BeginValidity has to be earlier than the EndValidity.
      (IF OBJ_RefDate < OBJ_BeginValidity THEN OBJ_VRS_Code == P ELSE OBJ_VRS_Code == E);
       !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
      (IF OBJ_VRS_Code == E THEN OBJ_DEC_SEG_VRS_Code == E AND OBJ_DEC_VRS_Code == E AND
OBJ_AXE_VRS_Code == E AND OBJ_RPT_VRS_Code_1 == E AND OBJ_RPT_VRS_Code_2 == E AND OBJ_RPT_VRS_Code_3 ==
E AND OBJ_PRO_VRS_Code == E)
       !! If the treated PMS-Object is an effective object, all his master objects have to be
       !! also effective.
      OBJ_RPT_DistBegin < OBJ_RPT_1.RPT_SectorL;
       !! The distance from the reference point to the location on the road axis has to be
       !! smaller than the corresponding sector length
      OBJ_RPT_DistEnd < OBJ_RPT_2.RPT_SectorL;
       !! The distance from the reference point to the location on the road axis has to be
       !! smaller than the corresponding sector length
      OBJ_RPT_DistBeginZone < OBJ_RPT_3.RPT_SectorL;
       !! The distance from the reference point to the location on the road axis has to be
       !! smaller than the corresponding sector length
      OBJ_RPT_DistEndZone < OBJ_RPT_4.RPT_SectorL;
       !! The distance from the reference point to the location on the road axis has to be
       !! smaller than the corresponding sector length
      OBJ_RPT_1.RPT_AXE_BaseID == OBJ_AXE.AXE_BaseID;
       !! The first reference point has to be on the indicated axis.
      OBJ_RPT_2.RPT_AXE_BaseID == OBJ_AXE.AXE_BaseID;
       !! The second reference point has to be on the indicated axis.
      OBJ_RPT_3.RPT_AXE_BaseID == OBJ_AXE.AXE_BaseID;
       !! The first reference point which is used to define the first location of the analyzed
       !! zone has to be on the indicated axis.
      OBJ_RPT_4.RPT_AXE_BaseID == OBJ_AXE.AXE_BaseID;
       !! The second reference point which is used to define the second location of the
       !! analyzed zone has to be on the indicated axis.
      OBJ_RPT_CK_1 < OBJ_RPT_CK_2;
       !! The second reference point has to be after the first reference point, seen from the
       !! direction of the indicated axis.
      OBJ_RPT_CK_3 < OBJ_RPT_CK_4;
       !! The second reference point of the analyzed zone has to be after the first reference
       !! point of the analyzed zone, seen from the direction of the indicated axis.
      OBJ_SCH.SCH_SEG_BaseID == OBJ_DEC.DEC_SEG_BaseID;
       !! The PMS-Segmentation on which the construction of the referred PMS-Segment is based
       !! has to be identic to the one used to define the "decoupage-PMS"
      UNIQUE
        OBJ_BaseID, OBJ_Version; !! PK
      UNIQUE
        OBJ_DEC_SEG_CKO_Owner, OBJ_DEC_SEG_CK, OBJ_DEC_SEG_VRS_Code, OBJ_DEC_CK, OBJ_SCH_CK,
        OBJ_No; !! UKC
    END Objet_PMS;
```

```
    TABLE Donnee_Etat =
    !! (D:‘Zustandsdaten‘, F:‘Données d'ètat relevées‘, Guide STRADA-PMS /GEN-SN640940 )
       ETA_Baseid:                      BASEID;                  !! PK             SN-GEN-I1
       ETA_Version:                     VERSION;                 !! PK             SN-GEN-I2
       ETA_OBJ_Baseid:                  BASEID;     //Kernel//;  !! FKL(1)->OBJ    GI-PMS-I1.0
       ETA_OBJ_DEC_Baseid:              BASEID;     //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.10
       ETA_OBJ_DEC_SEG_CKO_Owner:       OWNER       //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.11
       ETA_OBJ_DEC_SEG_CK:              TEXT*5      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.12
       ETA_OBJ_DEC_SEG_VRS_Code:        VERSCODE    //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.13
       ETA_OBJ_DEC_CKO_Owner:           OWNER       //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.14
       ETA_OBJ_DEC_CK:                  TEXT*5      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.15
       ETA_OBJ_DEC_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.16
       ETA_OBJ_SCH_Baseid:              BASEID;     //Kernel//;  !! FKL(1)->OBJ    GI-PMS-I1.20
       ETA_OBJ_SCH_CK:                  TEXT*8      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.21
       ETA_OBJ_SCH_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.22
       ETA_OBJ_No:                      TEXT*3      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.3
       ETA_OBJ_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.4
       ETA_SCA_CLT_CTK_BaseID:          BASEID      //Kernel//;  !! FKL(2)->SCA    GI-PMS-I2.0
       ETA_SCA_CLT_CTK_CKO_Owner:       OWNER       //Kernel//;  !! FKC(2)->SCA    GI-PMS-I2.1
       ETA_SCA_CLT_CTT_LNG_Code:        LANGCODE    //Kernel//;  !! FKC(2)->SCA    GI-PMS-I2.2
       ETA_SCA_CLT_TextID:              EXTID       //Kernel//;  !! FKC(2)->SCA    GI-PMS-I2.3
       ETA_TypAdText:      OPTIONAL     ADTEXT      //Kernel//;
       ETA_ParaMCT:                     PARAMCT     //Kernel//;  !! UKC            GI-PMS-I3
       ETA_ValeurMax:      OPTIONAL     PNUM4_4     //Kernel//;  !!                GI-PMS-A01
       ETA_ValeurMin:      OPTIONAL     PNUM4_4     //Kernel//;  !!                GI-PMS-A02
       ETA_ValeurMoy:      OPTIONAL     PNUM4_4     //Kernel//;  !!                GI-PMS-A03
       ETA_ValeurFreq:     OPTIONAL     PNUM4_4     //Kernel//;  !!                GI-PMS-A04
       ETA_EcartType:      OPTIONAL     PNUM4_4     //Kernel//;  !!                GI-PMS-A05
       ETA_Percentil:      OPTIONAL     PNUM4_4     //Kernel//;  !!                GI-PMS-A06
       ETA_VRS_Code:                    VERSCODE    //Kernel//;  !!                SN-GEN-A1
       ETA_RefDate:                     DATEDOM     //Kernel//;  !!                SN-GEN-D3
       ETA_BeginValidity:               DATEDOM     //Kernel//;  !! UKC            GI-PMS-I4
       ETA_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
       ETA_Comment:        OPTIONAL     COMMENT     //Kernel//;
       ETA_ODO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
       ETA_OrigSubDBID:                 DBID;                    !!                SN-GEN-A3
       ETA_DAO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
       ETA_DataOwner:                   DATAOWNER;               !!                SN-GEN-A5
       ETA_CreateDate:                  DATEDOM;                 !!                SN-GEN-A8
       ETA_ChangeDate:     OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
       ETA_CHO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
       ETA_ChangeUser:                  ORAUSER;                 !!                SN-GEN-A11
       ETA_ITS_Code:                    ITSCODE;                 !!                SN-GEN-A6
       ETA_IntegrityDate:               DATEDOM;                 !!                SN-GEN-A7
       ETA_XST_XfrSetID:   OPTIONAL     BASEID;                  !! FKL(.)->XST
    CONSTRAINT
      ETA_BeginValidity < ETA_EndValidity;
       !! The BeginValidity has to be earlier than the EndValidity.
      (IF ETA_RefDate < ETA_BeginValidity THEN ETA_VRS_Code == P ELSE ETA_VRS_Code == E);
       !! If the RefDate is earlier than the BeginValidity, then it is a predicted state.
     (IF ETA_VRS_Code == E THEN ETA_OBJ_DEC_SEG_VRS_Code == E AND ETA_OBJ_DEC_VRS_Code == E AND
ETA_OBJ_SCH_VRS_Code == E AND ETA_OBJ_VRS_Code_ == E)
       !! If the treated PMS-Object is an effective object, all his master objects have to be
       !! also effective.
      ETA_ValeurMin < ETA_ValeurMoy < ETA_ValeurMax;
       !! The average value has to be within the interval [minimal value, maximal value]
      ETA_ValeurMin < ETA_ValeurFreq < ETA_ValeurMax;
       !! The most frequent value has to be within the interval [minimal value, maximal value]
      ETA_ValeurMin < ETA_Percentil < ETA_ValeurMax;
       !! The percentil value has to be within the interval [minimal value, maximal value]
      ETA_EcartType < (ETA_ValeurMax - ETA_ValeurMin) / 2;
       !! The standard deviation has to be smaller than half the difference between the minimal
       !! and the maximal value
      UNIQUE
        ETA_BaseID, ETA_Version; !! PK
      UNIQUE
        ETA_DEC_SEG_CKO_Owner, ETA_DEC_SEG_CK, ETA_DEC_SEG_VRS_Code, ETA_DEC_CKO_Owner,
        ETA_DEC_CK, ETA_DEC_VRS_Code, ETA_OBJ_No, ETA_SCH_CK, ETA_SCA_CLT_CTK_CKO_Owner,
        ETA_SCA_CLT_CTT_LNG_Owner, ETA_SCA_CLT_TextID, ETA_ParaMCT, ETA_BeginValidity; !! UKC
    END Donnee_Etat;
```

```
    TABLE Analyse_PMS =
    !! (D:‘ PMS-Analyse‘ F:‘Analyse PMS‘, Guide STRADA-PMS /GEN-SN640940 )
       ANA_Baseid:                    BASEID;                  !! PK             SN-GEN-I1
       ANA_Version:                   VERSION;                 !! PK             SN-GEN-I1
       ANA_CKO_Owner:                 OWNER;      //Kernel//;  !! FKL(.)->OWN    GI-PMS-I1
       ANA_CK:                        TEXT*8      //Kernel//;  !! UKC            GI-PMS-I2
       ANA_Methode_Code:              METHCODE    //Kernel//;  !!                GI-PMS-A01
       ANA_NomUsuel:       OPTIONAL   ADTEXT      //Kernel//;  !!                GI-PMS-A02
       ANA_MethodeCalc:    OPTIONAL   TEXT*16     //Kernel//;  !!                GI-PMS-A03
       ANA_StatUtil_Code:             STUTCODE    //Kernel//;  !!                GI-PMS-A04
       ANA_VRS_Code:                  VERSCODE    //Kernel//;  !!                SN-GEN-A1
       ANA_RefDate:                   DATEDOM     //Kernel//;  !!                SN-GEN-D3
       ANA_BeginValidity:             DATEDOM     //Kernel//;  !!                SN-GEN-D1
       ANA_EndValidity:    OPTIONAL   DATEDOM     //Kernel//;  !!                SN-GEN-D2
       ANA_Comment:        OPTIONAL   COMMENT     //Kernel//;
       ANA_ODO_Owner:                 OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
       ANA_OrigSubDBID:               DBID;                    !!                SN-GEN-A3
       ANA_DAO_Owner:                 OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
       ANA_DataOwner:                 DATAOWNER;               !!                SN-GEN-A5
       ANA_CreateDate:                DATEDOM;                 !!                SN-GEN-A8
       ANA_ChangeDate:     OPTIONAL   DATEDOM;                 !!                SN-GEN-A9
       ANA_CHO_Owner:                 OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
       ANA_ChangeUser:                ORAUSER;                 !!                SN-GEN-A11
       ANA_ITS_Code:                  ITSCODE;                 !!                SN-GEN-A6
       ANA_IntegrityDate:             DATEDOM;                 !!                SN-GEN-A7
       ANA_XST_XfrSetID:   OPTIONAL   BASEID;                  !! FKL(.)->XST
    CONSTRAINT
      ANA_Version != 1;
!! The Analyse_PMS describes an activity. Therefore, no more than one version exists.
      ANA_EndValidity IS DEFINED;
       !! The Analyse_PMS describes an activity. Therefore, the EndValidity has to be defined.
      ANA_BeginValidity < ANA_EndValidity;
       !! The BeginValidity has to be earlier than the EndValidity.
      (IF ANA_RefDate < ANA_BeginValidity THEN ANA_VRS_Code == P ELSE ANA_VRS_Code == E);
       !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
      UNIQUE
        ANA_BaseID, ANA_Version; !! PK
      UNIQUE
        ANA_CKO_Owner, ANA_CK; !! UKC
    END Analyse_PMS;
```

```
   TABLE Variante_Analyse_PMS =
   !! (D:‘Variante PMS-Analyse‘ F:‘Variante d'analyse PMS‘, Guide STRADA-PMS /GEN-SN640940 )
      VAR_Baseid:                      BASEID;                  !! PK             SN-GEN-I1
      VAR_Version:                     VERSION;                 !! PK             SN-GEN-I2
      VAR_ANA_Baseid:                  BASEID;                  !! FKL(1)->ANA    GI-PMS-I1.0
      VAR_ANA_CKO_Owner:  OWNER;       //Kernel//; !!           UKC(1)->ANA       GI-PMS-I1.1
      VAR_ANA_CK:                      TEXT*8      //Kernel//;  !! UKC(1)->ANA    GI-PMS-I1.2
      VAR_ANA_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(1)->ANA    GI-PMS-I1.3
      VAR_CK:                          TEXT*4      //Kernel//;  !! UKC            GI-PMS-I2
      VAR_AnneeDebut:                  NUM4        //Kernel//;  !!                GI-PMS-A01
      VAR_NombreAnnee:                 NUM2        //Kernel//;  !!                GI-PMS-A02
      VAR_CondAnalyse_Code:            CONDCODE    //Kernel//;  !!                GI-PMS-A03
      VAR_CritAnalyse_Code:            CRITCODE    //Kernel//;  !!                GI-PMS-A04
      VAR_TauxRench:      OPTIONAL     NUM3_2      //Kernel//;  !!                GI-PMS-A05
      VAR_TauxActual:     OPTIONAL     NUM3_2      //Kernel//;  !!                GI-PMS-A06
      VAR_TauxVarTrafic: OPTIONAL      NUM3_2      //Kernel//;  !!                GI-PMS-A07
      VAR_VRS_Code:                    VERSCODE    //Kernel//;  !!                SN-GEN-A1
      VAR_RefDate:                     DATEDOM     //Kernel//;  !!                SN-GEN-D3
      VAR_BeginValidity:  DATEDOM      //Kernel//; !!              SN-GEN-D1
      VAR_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
      VAR_Comment:        OPTIONAL     COMMENT     //Kernel//;
      VAR_ODO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
      VAR_OrigSubDBID:                 DBID;                    !!                SN-GEN-A3
      VAR_DAO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
      VAR_DataOwner:                   DATAOWNER;               !!                SN-GEN-A5
      VAR_CreateDate:                  DATEDOM;                 !!                SN-GEN-A8
      VAR_ChangeDate:     OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
      VAR_CHO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
      VAR_ChangeUser:                  ORAUSER;                 !!                SN-GEN-A11
      VAR_ITS_Code:                    ITSCODE;                 !!                SN-GEN-A6
      VAR_IntegrityDate:  DATEDOM;                  !!           SN-GEN-A7
      VAR_XST_XfrSetID:   OPTIONAL     BASEID;                  !! FKL(.)->XST
   CONSTRAINT
     VAR_Version != 1;
!! The Variante_Analyse_PMS describes an activity. Therefore, no more than one version
!! exists.
     VAR_EndValidity IS DEFINED;
      !! The Variante_Analyse_PMS describes an activity. Therefore, the EndValidity has to be
      !! defined.
     VAR_BeginValidity < VAR_EndValidity;
      !! The BeginValidity has to be earlier than the EndValidity.
     (IF VAR_RefDate < VAR_BeginValidity THEN VAR_VRS_Code == P ELSE VAR_VRS_Code == E);
      !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
     (IF VAR_VRS_Code == E THEN VAR_ANA_VRS_Code == E);
      !! If the treated Variante_Analyse_PMS is an effective object, his master object has to
      !! be also effective.
     UNIQUE
       VAR_BaseID, VAR_Version; !! PK
     UNIQUE
       VAR_ANA_CKO_Owner, VAR_ANA_CK, VAR_ANA_VRS_Code, VAR_CK; !! UKC
   END Variante_Analyse_PMS;
```

```
TABLE Analyse_PMS_Decoupage_PMS =
!! (D:‘PMS-Analyse PMS-Decoupage‘ F:‘Analyse PMS Découpage PMS‘,GEN-SN640940 )
   ADE_ANA_Baseid:                 BASEID;     //Kernel//;  !! FKL(1)->ANA
   ADE_ANA_CKO_OWNER:  OWNER;      //Kernel//; !!           UKC(1)->ANA
   ADE_ANA_CK:                     TEXT*8      //Kernel//;  !! UKC(1)->ANA
   ADE_ANA_VRS_Code:               VERSCODE    //Kernel//;  !! UKC(1)->ANA
   ADE_DEC_Baseid:                 BASEID;     //Kernel//;  !! FKL(2)->DEC
   ADE_DEC_SEG_Baseid: BASEID;     //Kernel//; !!           UKC(2)->DEC
   ADE_DEC_SEG_CKO_Owner:          OWNER       //Kernel//;  !! UKC(2)->DEC
   ADE_DEC_SEG_CK:                 TEXT*5      //Kernel//;  !! UKC(2)->DEC
   ADE_DEC_SEG_VRS_Code:           VERSCODE    //Kernel//;  !! UKC(2)->DEC
   ADE_DEC_CKO_Owner:  OWNER       //Kernel//; !!           UKC(2)->DEC
   ADE_DEC_CK:                     TEXT*5      //Kernel//;  !! UKC(2)->DEC
   ADE_DEC_VRS_Code:               VERSCODE    //Kernel//;  !!
CONSTRAINT
  UNIQUE
    ADE_ANA_CKO_Owner, ADE_ANA_CK, ADE_ANA_VRS_Code, ADE_DEC_SEG_CKO_Owner,
    ADE_ANA_DEC_SEG_CK, ADE_DEC_SEG_VRS_Code, ADE_DEC_CKO_OWNER, ADE_DEC_CK; !! UKC
END Analyse_PMS_Decoupage_PMS;
```

```
    TABLE Periode =
    !! (D:‘ Periode‘ F:‘Période de l'analyse‘, Guide STRADA-PMS /GEN-SN640940 )
       PAN_Baseid:                      BASEID;                  !! PK             SN-GEN-I1
       PAN_Version:                     VERSION;                 !! PK             SN-GEN-I2
       PAN_VAR_Baseid:                  BASEID;                  !! FKL(1)->VAR    GI-PMS-I1.0
       PAN_VAR_ANA_CKO_Owner:           OWNER;      //Kernel//;  !! UKC(1)->VAR    GI-PMS-I1.1
       PAN_VAR_ANA_CK:                  TEXT*8      //Kernel//;  !! UKC(1)->VAR    GI-PMS-I1.2
       PAN_VAR_ANA_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->VAR    GI-PMS-I1.3
       PAN_VAR_CK:                      TEXT*4      //Kernel//;  !! UKC(1)->VAR    GI-PMS-I1.4
       PAN_VAR_VRS_Code:                VERSCODE    //Kernel//;  !! UKC(1)->VAR    GI-PMS-I1.5
       PAN_Annee:                       NUM4        //Kernel//;  !! UKC            GI-PMS-I2
       PAN_BudgetFixe:                  NUM5        //Kernel//;  !!                GI-PMS-A01
       PAN_BudgetPropose: OPTIONAL      NUM5        //Kernel//;  !!                GI-PMS-A02
       PAN_BudgetEntre:    OPTIONAL     NUM5        //Kernel//;  !!                GI-PMS-A03
       PAN_EtatReseauVise: OPTIONAL     NUM1_1      //Kernel//;  !!                GI-PMS-A04
       PAN_EtatResAtt:     OPTIONAL NUM1_1          //Kernel//;  !!                GI-PMS-A05
       PAN_VRS_Code:                    VERSCODE    //Kernel//;  !!                SN-GEN-A1
       PAN_RefDate:                     DATEDOM     //Kernel//;  !!                SN-GEN-D3
       PAN_BeginValidity:  DATEDOM      //Kernel//; !!               SN-GEN-D1
       PAN_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
       PAN_Comment:        OPTIONAL     COMMENT     //Kernel//;
       PAN_ODO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
       PAN_OrigSubDBID:                 DBID;                    !!                SN-GEN-A3
       PAN_DAO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
       PAN_DataOwner:                   DATAOWNER;               !!                SN-GEN-A5
       PAN_CreateDate:                  DATEDOM;                 !!                SN-GEN-A8
       PAN_ChangeDate:     OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
       PAN_CHO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
       PAN_ChangeUser:                  ORAUSER;                 !!                SN-GEN-A11
       PAN_ITS_Code:                    ITSCODE;                 !!                SN-GEN-A6
       PAN_IntegrityDate:  DATEDOM;                  !!             SN-GEN-A7
       PAN_XST_XfrSetID:   OPTIONAL     BASEID;                  !! FKL(.)->XST
    CONSTRAINT
      PAN_Version != 1;
 !! No more version than one version exists of a period.
      PAN_EndValidity IS DEFINED;
       !! The Variante_Analyse_PMS describes an activity. Therefore, the EndValidity has to be
       !! defined.
      PAN_BeginValidity < PAN_EndValidity;
       !! The BeginValidity has to be earlier than the EndValidity.
      (IF PAN_RefDate < PAN_BeginValidity THEN PAN_VRS_Code == P ELSE PAN_VRS_Code == E);
       !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
      (IF PAN_VRS_Code == E THEN PAN_VAR_ANA_VRS_Code == E AND PAN_VAR_VRS_Code == E);
       !! If the treated Variante_Analyse_PMS is an effective object, his master object has to
       !! be also effective.
      PAN_BudgetFixe IS DEFINED OR PAN_EtatReseauVise IS DEFINED;
       !! One of the these attributs has to contain a value.
      UNIQUE
        PAN_BaseID, PAN_Version; !! PK
        PAN_VAR_ANA_CKO_Owner, PAN_VAR_ANA_CK, PAN_VAR_ANA_VRS_Code, PAN_VAR_CK,
        PAN_VAR_VRS_Code, PAN_Annee; !! UKC
    END Periode;
```

```
TABLE Mesure_Type =
!! (D:‘Massnahmentyp‘ F:‘Mesure-type‘, Guide STRADA-PMS /GEN-SN640940 )
   MET_Baseid:                      BASEID;                  !! PK             SN-GEN-I1
   MET_Version:                     VERSION;                 !! PK             SN-GEN-I1
   MET_CKO_Owner                    OWNER       //Kernel//;  !! FKL(.)->OWN    GI-PMS-I1
   MET_SCA_PRT_CTK_BaseID:          BASEID      //Kernel//;  !! FKL(1)->SCA    GI-PMS-I2.0
   MET_SCA_PRT_CTK_CKO_Owner:       OWNER       //Kernel//;  !! UKC(1)->SCA    GI-PMS-I2.1
   MET_SCA_PRT_CTT_LNG_Code:        LANGCODE    //Kernel//;  !! UKC(1)->SCA    GI-PMS-I2.2
   MET_SCA_PRT_TextID: EXTID        //Kernel//; !!           UKC(1)->SCA       GI-PMS-I2.3
   MET_TypAdText:      OPTIONAL     ADTEXT      //Kernel//;
   MET_TypeMesure_Code:             TYMESCODE   //Kernel//;  !! UKC            GI-PMS-I3
   MET_Designation:                 ADTEXT      //Kernel//;  !!                GI-PMS-A01
   MET_CatBudget_Code: OPTIONAL     BUDGCODE    //Kernel//;  !!                GI-PMS-A02
   MET_Unite_Code:     OPTIONAL     UNCODE      //Kernel//;  !!                GI-PMS-A03
   MET_Productivite:   OPTIONAL     PNUM06_2    //Kernel//;  !!                GI-PMS-A04
   MET_CoutPetiteQte: OPTIONAL      PNUM4_2     //Kernel//;  !!                GI-PMS-A05
   MET_LimPetiteQte:   OPTIONAL     PNUM6_2     //Kernel//;
   MET_CoutGrandeQuant:OPTIONAL     PNUM4_2     //Kernel//;  !!                GI-PMS-A06
   MET_LimGrandeQuant: OPTIONAL     PNUM6_2     //Kernel//;
   MET_Delta_I1:       OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A07
   MET_I1_Code:        OPTIONAL     INDCODE     //Kernel//;  !!                GI-PMS-A08
   MET_Delta_I2:       OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A09
   MET_I2_Code:        OPTIONAL     INDCODE     //Kernel//;  !!                GI-PMS-A10
   MET_Delta_I3:       OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A11
   MET_I3_Code:        OPTIONAL     INDCODE     //Kernel//;  !!                GI-PMS-A12
   MET_Delta_I4:       OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A13
   MET_I4_Code:        OPTIONAL     INDCODE     //Kernel//;  !!                GI-PMS-A14
   MET_Delta_I5:       OPTIONAL     PNUM3_2     //Kernel//;  !!                GI-PMS-A15
   MET_I5_Code:        OPTIONAL     INDCODE     //Kernel//;  !!                GI-PMS-A16
   MET_LimInf_I1:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimSup_I1:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimInf_I2:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimSup_I2:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimInf_I3:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimSup_I3:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimInf_I4:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimSup_I4:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimInf_I5:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_LimSup_I5:      OPTIONAL     PNUM1_1     //Kernel//;
   MET_VRS_Code:                    VERSCODE    //Kernel//;  !!                SN-GEN-A1
   MET_RefDate:                     DATEDOM     //Kernel//;  !!                SN-GEN-D3
   MET_BeginValidity:  DATEDOM      //Kernel//; !!              SN-GEN-D1
   MET_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;  !!                SN-GEN-D2
   MET_Comment:        OPTIONAL     COMMENT     //Kernel//;
   MET_ODO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A2
   MET_OrigSubDBID:                 DBID;                    !!                SN-GEN-A3
   MET_DAO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A4
   MET_DataOwner:                   DATAOWNER;               !!                SN-GEN-A5
   MET_CreateDate:                  DATEDOM;                 !!                SN-GEN-A8
   MET_ChangeDate:     OPTIONAL     DATEDOM;                 !!                SN-GEN-A9
   MET_CHO_Owner:                   OWNER;                   !! FKL(.)->OWN    SN-GEN-A10
   MET_ChangeUser:                  ORAUSER;                 !!                SN-GEN-A11
   MET_ITS_Code:                    ITSCODE;                 !!                SN-GEN-A6
   MET_IntegrityDate:               DATEDOM;                 !!                SN-GEN-A7
   MET_XST_XfrSetID:   OPTIONAL     BASEID;                  !! FKL(.)->XST
CONSTRAINT
  MET_BeginValidity < MET_EndValidity;
   !! The BeginValidity has to be earlier than the EndValidity.
  (IF MES_RefDate < MES_BeginValidity THEN MES_VRS_Code == P ELSE MES_VRS_Code == E);
   !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity
  MET_LimInf_I1 < MET_LimSup_I1;
   !! The inferior limit I1 is smaller than the superior limit I1.
  MET_LimInf_I2 < MET_LimSup_I2;
   !! The inferior limit I2 is smaller than the superior limit I2.
  MET_LimInf_I3 < MET_LimSup_I3;
   !! The inferior limit I3 is smaller than the superior limit I3.
  MET_LimInf_I4 < MET_LimSup_I4;
   !! The inferior limit I4 is smaller than the superior limit I4.
  MET_LimInf_I5 < MET_LimSup_I5;
   !! The inferior limit I5 is smaller than the superior limit I5.
  UNIQUE
    MET_BaseID, MET_Version; !! PK
    MET_CKO_Owner, MET_SCA_PRT_CTK_CKO_Owner, MET_SCA_PRT_CTT_LNG_Code, MET_SCA_PRT_TextID,
    MET_TypeMesure_Code; !! UKC
END Mesure_Type;
```

```
    TABLE Mesure_prevue =
    !! (D:‘Vorgesehene Massnahme ‘ F:‘Mesure prévue‘, Guide STRADA-PMS /GEN-SN640940 )
       MES_OBJ_Baseid:                  BASEID;                  !! FKL(1)->OBJ    SN-GEN-I1
       MES_OBJ_DEC_Baseid: BASEID;                   !!          UKC(1)->OBJ       GI-PMS-I1.0
       MES_OBJ_SEG_CKO_Owner:           OWNER       //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.11
       MES_OBJ_SEG_CK:                  TEXT*5      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.12
       MES_OBJ_SEG_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.13
       MES_OBJ_DEC_CKO_Owner:           OWNER       //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.21
       MES_OBJ_DEC_CK:                  TEXT*5      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.22
       MES_OBJ_DEC_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.23
       MES_OBJ_SCH_Baseid: BASEID;      //Kernel//; !!           FKL(1)->OBJ       GI-PMS-I1.30
       MES_OBJ_SCH_CK:                  TEXT*8      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.31
       MES_OBJ_NO:                      TEXT*3      //Kernel//;  !! UKC(1)->OBJ    GI-PMS-I1.4
       MES_PAN_Baseid:                  BASEID;                  !! FKL(2)->VAR    GI-PMS-I2.0
       MES_PAN_VAR_ANA_CKO_OWNER:       OWNER;      //Kernel//;  !! UKC(2)->VAR    GI-PMS-I2.11
       MES_PAN_VAR_ANA_CK: TEXT*8       //Kernel//; !!           UKC(2)->VAR       GI-PMS-I2.12
       MES_PAN_VAR_ANA_VRS_Code:        VERSCODE    //Kernel//;  !! UKC(2)->VAR    GI-PMS-I2.13
       MES_PAN_VAR_CK:                  TEXT*4      //Kernel//;  !! UKC(2)->VAR    GI-PMS-I2.2
       MES_PAN_VAR_VRS_Code:            VERSCODE    //Kernel//;  !! UKC(2)->VAR    GI-PMS-I2.3
       MES_PAN_Annee:                   PNUM4       //Kernel//;  !! UKC(2)->VAR    GI-PMS-I2.4
       MES_MET_Baseid:                  BASEID;                  !! FKL(3)->MET    GI-PMS-I3.0
       MES_MET_CKO_Owner:  OWNER        //Kernel//; !!           UKC(3)->MET       GI-PMS-I3.1
       MES_MET_SCA_PRT_CTK_BaseID:      BASEID      //Kernel//;  !! FKL(.)->MET    GI-PMS-I3.20
       MES_MET_SCA_PRT_CTK_CKO_Owner: OWNER         //Kernel//;  !! UKC(3)->MET    GI-PMS-I3.21
       MES_MET_SCA_PRT_CTT_LNG_Code:    LANGCODE    //Kernel//;  !! UKC(3)->MET    GI-PMS-I3.22
       MES_MET_SCA_PRT_TextID:          EXTID       //Kernel//;  !! UKC(3)->MET    GI-PMS-I3.23
       MES_MET_TypeMesure_Code:         TYMESCODE   //Kernel//;  !! UKC(3)->MET    GI-PMS-I3.3
       MES_Statut_Code:                 STMODCODE   //Kernel//;  !!                GI-PMS-A01
       MES_CoutMeSure:                  PNUM9       //Kernel//;  !!                GI-PMS-A02
       MES_CoutTravSuppl:  OPTIONAL     PNUM6       //Kernel//;  !! UKC(2)->VAR    GI-PMS-A03
    CONSTRAINT
      MES_Version != 1;
 !! The planified measure is an planified activity. Therefore, no more than one version
 !! exists.
      MES_EndValidity IS DEFINED;
       !! The planified measure is an planified activity. Therefore, the EndValidity has to be
       !! defined.
      (IF MET_RefDate < MET_BeginValidity THEN MET_VRS_Code == P ELSE MET_VRS_Code == E);
       !! If the RefDate is earlier than the BeginValidity, then it is a plannified activity.
      (IF MET_VRS_Code == E THEN MES_PAN_VAR_VRS_Code == E);
       !! If the planified measure is an effective object, his master object has to
       !! be also effective.
      UNIQUE
        MES_OBJ_SEG_CKO_Owner, MES_OBJ_SEG_CK, MES_OBJ_SEG_VRS_Code, MES_OBJ_DEC_CKO_Owner,
        MES_OBJ_DEC_CK, MES_OBJ_DEC_VRS_Code, MES_OBJ_SCH_CK, MES_OBJ_NO,
        MES_PAN_VAR_ANA_CKO_OWNER, MES_PAN_VAR_ANA_CK, MES_PAN_VAR_ANA_VRS_Code,
        MES_PAN_VAR_CK, MES_PAN_VAR_VRS_Code, MES_PAN_Annee, MES_SCA_PRT_CTK_CKO_Owner,
        MES_SCA_PRT_CTT_LNG_Code, MES_SCA_PRT_TextID, MES_TypeMesure_Code; !! UKC
    END Mesure_prevue;
```

```
!! All JOKER-entity types


    TABLE Joker_Types =
    !! (D:‘Joker Objekttyp ‘ F:‘Type d'objet Joker‘, Analysis STRADA-DB/Joker /GEN-SN640940)
       JOT_Baseid:                      BASEID;                 !! PK             SN-GEN-I1
       JOT_Version:                     VERSION;                !! PK             SN-GEN-I2
       JOT_JOT_Baseid:                  BASEID      //Kernel//; !! FKL(3)->JOT    AN-JOK-A4.01
       JOT_JOT_Version:                 VERSION     //Kernel//; !! FKL(3)->JOT    AN-JOK-A4.02
       JOT_JOT_CKO_Owner:               OWNER       //Kernel//; !! FKC(3)->JOT    AN-JOK-A4.1
       JOT_JOT_CK:                      CK2         //Kernel//; !! FKC(3)->JOT    AN-JOK-A4.2
       JOT_CKO_Owner:                   OWNER       //Kernel//; !! UKC(1)->OWN    AN-JOK-I1
       JOT_CK:                          CK2         //Kernel//; !! UKC(1)         AN-JOK-I2
       JOT_LBL_Baseid:                  BASEID;                 !! FKL(2)->LBL    AN-JOK-A1.0
       JOT_LBL_Version:                 VERSION;                !! FKL(2)->LBL    AN-JOK-A1.0
       JOT_JOT_MAND:                    TEXT*1      //Kernel//; !!                AN-JOK-A5
       JOT_SpatialCode:                 TEXT*2      //Kernel//; !!                AN-JOK-A3
       JOT_VRS_Code:                    VERSCODE    //Kernel//; !!                SN-GEN-A1
       JOT_RefDate:                     DATEDOM     //Kernel//; !!                SN-GEN-D3
       JOT_BeginValidity:  DATEDOM      //Kernel//; !!              SN-GEN-D1
       JOT_EndValidity:    OPTIONAL     DATEDOM     //Kernel//; !!                SN-GEN-D2
       JOT_Comment:        OPTIONAL     COMMENT     //Kernel//;
       JOT_ODO_Owner:                   OWNER;                  !! FKL(.)->OWN    SN-GEN-A2
       JOT_OrigSubDBID:                 DBID;                   !!                SN-GEN-A3
       JOT_DAO_Owner:                   OWNER;                  !! FKL(.)->OWN    SN-GEN-A4
       JOT_DataOwner:                   DATAOWNER;              !!                SN-GEN-A5
       JOT_CreateDate:                  DATEDOM;                !!                SN-GEN-A8
       JOT_ChangeDate:     OPTIONAL     DATEDOM;                !!                SN-GEN-A9
       JOT_CHO_Owner:                   OWNER;                  !! FKL(.)->OWN    SN-GEN-A10
       JOT_ChangeUser:                  ORAUSER;                !!                SN-GEN-A11
       JOT_ITS_Code:                    ITSCODE;                !!                SN-GEN-A6
       JOT_IntegrityDate:               DATEDOM;                !!                SN-GEN-A7
       JOT_XST_XfrSetID:   OPTIONAL     BASEID;                 !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
        JOT_Baseid, JOT_Version;  !! PK
        JOT_CKO_Owner, JOT_CK;    !! UKC
    END Joker_Types;
```

```
TABLE Joker_Attributes =
!! (D:‘Joker Attribute ’ F:‘Attributs Joker‘,GEN-SN640940)
   JAT_Baseid:                       BASEID;                  !! PK             SN-GEN-I1
   JAT_Version:                      VERSION;                 !! PK             SN-GEN-I2
   JAT_JOT_Baseid:                   BASEID      //Kernel//;  !! FKL(1)->JOT
   JAT_JOT_Version:                  VERSION     //Kernel//;  !! FKL(1)->JOT
   JAT_JOT_CKO_Owner:                OWNER       //Kernel//;  !! FKC(1)->JOT
   JAT_JOT_CK:                       CK2         //Kernel//;  !! FKC(1)->JOT
   JAT_XXX_Baseid:     OPTIONAL      BASEID      //Kernel//;  !! FKL(2)->(CAT OR COD)
   JAT_SCA_CAT_Baseid: OPTIONAL      BASEID      //Kernel//;  !!
   JAT_SCA_CAT_CKO_Owner: OPTIONAL   OWNER       //Kernel//;  !!
   JAT_SCA_CAT_CK:     OPTIONAL      CK4         //Kernel//;  !!
   JAT_SEQ:                          NUM5        //Kernel//;  !!
   JAT_Type:                         DATATYPE    //Kernel//;
   JAT_Mand:                         TEXT*2      //Kernel//;
   JAT_DefValue:       OPTIONAL      TEXT*72     //Kernel//;
   JAT_MaxLength:      OPTIONAL      NUM5        //Kernel//;
   JAT_Format:         OPTIONAL      NUM5        //Kernel//;
   JAT_Numn:           OPTIONAL      NUM5        //Kernel//;
   JAT_Numm:           OPTIONAL      NUM5        //Kernel//;
   JAT_MinValue:       OPTIONAL      TEXT*38     //Kernel//;
   JAT_MaxValue:       OPTIONAL      TEXT*38     //Kernel//;
CONSTRAINT
  UNIQUE
    JAT_Baseid, JAT_Version;  !! PK
END Joker_Attributes;
```

```
TABLE Joker_Define =
!! (D: 'Reihenfolge der Attribute pro Jokertyp, F: 'Ordre des attributs par type de Joker',
!!  GEN-SN640940)
   JOD_SEQ:                        NUM5;
   JOD_Type:                       DATATYPE;
END Joker_Define;

TABLE Joker_Help =
!! (D: 'Joker-Hilfe, F: 'Aide pour le Joker', GEN-SN640940)
   JHP_JOT_Baseid:                 BASEID;                  !! PK            SN-GEN-I1
   JHP_JOT_Version:                VERSION;                 !! PK            SN-GEN-I2
   JHP_JOT_CKO_Owner:              OWNER       //Kernel//;  !! UKC->JOT
   JHP_JOT_CK:                     CK2         //Kernel//;  !! UKC->JOT
   JHP_LNG_Code:                   LANGCODE;                !! PK, UKC
   JHP_Text:                       TEXT*2000
CONSTRAINT
  UNIQUE
    JHP_JOT_Baseid, JHP_JOT_Version, JHP_LNG_Code;  !! PK
    JHP_JOT_CKO_Owner, JHP_JOT_CK, JHP_LNG_Code;    !! UKC
END Joker_Help;

TABLE Joker_Labels =
!! (D: 'Joker Anschriften', F: 'Libellé Joker', Analysis STRADA-DB/GEN-SN640940)
   JAL_JAT_Baseid:                 BASEID;                  !! PK            SN-GEN-I1
   JAL_JAT_Version:                VERSION;                 !! PK            SN-GEN-I2
   JAL_LNG_Code:                   LANGCODE;                !!
   JAL_Text:                       TEXT*20;                 !!
   JAL_Text_Long:     OPTIONAL     TEXT*72;                 !!
CONSTRAINT
  UNIQUE
    JAL_JAT_Baseid, JAL_JAT_Version;  !! PK
END Joker_Labels;
```

```
TABLE Jokers =
    !! (D:‘Joker-Objekte ‘ F:‘Objets Joker‘, Analysis STRADA-DB/Joker /GEN-SN640940)
       JOK_Baseid:                       BASEID;                    !! PK             SN-GEN-I1
       JOK_Version:                      VERSION;                   !! PK             SN-GEN-I2
       JOK_JOT_Baseid:                   BASEID      //Kernel//;    !! FKL(1)->JOT    AN-JOK-I1.01
       JOK_JOT_Version:                  VERSION     //Kernel//;    !! FKL(1)->JOT    AN-JOK-I1.02
       JOK_JOT_CKO_Owner:                OWNER       //Kernel//;    !! FKC(1)->JOT    AN-JOK-I1.11
       JOK_JOT_CK:                       CK2         //Kernel//;    !! FKC(1)->JOT    AN-JOK-I1.12
       JOK_JOK_Baseid:     OPTIONAL      BASEID      //Kernel//;    !! FKL(2)->JOK    AN-JOK-A2.01
       JOK_JOK_Version:    OPTIONAL      VERSION     //Kernel//;    !! FKL(2)->JOK    AN-JOK-A2.02
       JOK_JOK_CKO_Owner:  OPTIONAL      OWNER       //Kernel//;    !! FKC(2)->JOK    AN-JOK-A2.11
       JOK_JOK_CK:         OPTIONAL      CK2         //Kernel//;    !! FKC(2)->JOK    AN-JOK-A2.12
       JOK_PRO_Baseid:     OPTIONAL      BASEID      //Kernel//;    !! FKL(3)->PRO    AN-JOK-A5.0
       JOK_PRO_CKO_Owner:  OPTIONAL      OWNER       //Kernel//;    !! FKC(3)->PRO    AN-JOK-A5.1
       JOK_PRO_CK:         OPTIONAL      CK2         //Kernel//;    !! FKC(3)->PRO    AN-JOK-A5.2
       JOK_PRO_VRS_Code:   OPTIONAL      VRSCODE     //Kernel//;    !! FKC(3)->PRO    AN-JOK-A5.3
       JOK_LNK_Baseid:     OPTIONAL      BASEID      //Kernel//;    !! FKL(4)->LNK    AN-JOK-A14.0
       JOK_LNK_AXE_CKO_Owner: OPTIONAL   OWNER       //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.11
       JOK_LNK_AXE_CK:     OPTIONAL      CK3         //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.12
       JOK_LNK_AXE_POS_Code: OPTIONAL    POSCODE     //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.13
       JOK_LNK_AXE_VRS_Code: OPTIONAL    VERSCODE    //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.14
       JOK_LNK_NLO_NOD_CKO_Owner_1: OPTIONAL OWNER   //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.15
       JOK_LNK_NLO_NOD_CK_1: OPTIONAL    CK4         //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.16
       JOK_LNK_NLO_NOD_VRS_CODE: OPTIONAL VERSCODE   //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.17
       JOK_LNK_NLO_RPT_CK_1: OPTIONAL    CK2         //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.18
       JOK_LNK_NLO_RPT_VRS_Code_1: OPTIONAL VERSCODE //Kernel//; !! FKC(4)->LNK    AN-JOK-A14.19
       JOK_LNK_NLO_VRS_Code_1: OPTIONAL VERSCODE     //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.20
       JOK_LNK_NLO_NOD_CKO_Owner_2: OPTIONAL OWNER   //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.21
       JOK_LNK_NLO_NOD_CK_2: OPTIONAL    CK4         //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.22
       JOK_LNK_NLO_NOD_VRS_CODE_2: OPTIONAL VERSCODE //Kernel//; !! FKC(4)->LNK    AN-JOK-A14.23
       JOK_LNK_NLO_RPT_CK_2: OPTIONAL    CK3         //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.24
       JOK_LNK_NLO_RPT_VRS_Code_2: OPTIONAL VERSCODE //Kernel//; !! FKC(4)->LNK    AN-JOK-A14.25
       JOK_LNK_NLO_VRS_Code_2: OPTIONAL VERSCODE     //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.26
       JOK_LNK_LKG_CKO_Owner: OPTIONAL   OWNER       //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.27
       JOK_LNK_LKG_CK:     OPTIONAL      CK4         //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.28
       JOK_LNK_LKG_SCA_SLT_CTK_CKO_Owner: OPTIONAL OWNER //Kernel//; !! FKC(4)->LNK  AN-JOK-A14.29
       JOK_LNK_LKG_SCA_SLT_CTT_LNG_Code: OPTIONAL LANGCODE //Kernel//; !! FKC(4)->LNK AN-JOK-A14.30
       JOK_LNK_LKG_SCA_SLT_CTT_TextID: OPTIONAL TEXTID //Kernel//; !! FKC(4)->LNK  AN-JOK-A14.31
       JOK_LNK_LKG_VRS_Code: OPTIONAL    VERSCODE    //Kernel//;    !! FKC(4)->LNK    AN-JOK-A14.32
       JOK_AXE_CKO_Owner:  OPTIONAL      OWNER       //Kernel//;    !! FKC(5)->AXE    AN-JOK-A7.11
       JOK_AXE_CKO:        OPTIONAL      CK3         //Kernel//;    !! FKC(5)->AXE    AN-JOK-A7.12
       JOK_AXE_VRS_Code:   OPTIONAL      VERSCODE    //Kernel//;    !! FKC(5)->AXE    AN-JOK-A7.13
       JOK_RPT_Baseid1:    OPTIONAL      BASEID      //Kernel//;    !! FKL(6)->RPT    AN-JOK-A7
       JOK_RPT_CK_1:       OPTIONAL      CK2         //Kernel//;    !! FKC(6)->RPT    AN-JOK-A7.4
       JOK_RPT_VRS_Code_1: OPTIONAL      VERSCODE    //Kernel//;    !! FKC(6)->RPT    AN-JOK-A7.41
       JOK_RPT_Baseid_2:   OPTIONAL      BASEID      //Kernel//;    !! FKL(7)->RPT    AN-JOK-A11
       JOK_RPT_CK_2:       OPTIONAL      CK2         //Kernel//;    !! FKC(7)->RPT    AN-JOK-A11.4
       JOK_RPT_VRS_Code_2: OPTIONAL      VERSCODE    //Kernel//;    !! FKC(7)->RPT    AN-JOK-A11.41
       JOK_SCA_CTK_Baseid1: OPTIONAL     BASEID      //Kernel//;    !! FKL(8)->SCA    AN-JOK-A16.10
       JOK_SCA_CTK_CKO_Owner1: OPTIONAL  OWNER       //Kernel//;    !! FKC(8)->SCA    AN-JOK-A16.11
       JOK_SCA_CTT_LNG_Code1: OPTIONAL   LANGCODE    //Kernel//;    !! FKC(8)->SCA    AN-JOK-A16.12
       JOK_SCA_CTT_TextID1: OPTIONAL     TEXTID      //Kernel//;    !! FKC(8)->SCA    AN-JOK-A16.13
       JOK_SCA_CTK_Baseid2: OPTIONAL     BASEID      //Kernel//;    !! FKL(9)->SCA    AN-JOK-A16.20
       JOK_SCA_CTK_CKO_Owner2: OPTIONAL  OWNER       //Kernel//;    !! FKC(9)->SCA    AN-JOK-A16.21
       JOK_SCA_CTT_LNG_Code2: OPTIONAL   LANGCODE    //Kernel//;    !! FKC(9)->SCA    AN-JOK-A16.22
       JOK_SCA_CTT_TextID2: OPTIONAL     TEXTID      //Kernel//;    !! FKC(9)->SCA    AN-JOK-A16.23
       JOK_SCA_CTK_Baseid3: OPTIONAL     BASEID      //Kernel//;    !! FKL(10)->SCA   AN-JOK-A16.30
       JOK_SCA_CTK_CKO_Owner3: OPTIONAL  OWNER       //Kernel//;    !! FKC(10)->SCA   AN-JOK-A16.31
       JOK_SCA_CTT_LNG_Code3: OPTIONAL   LANGCODE    //Kernel//;    !! FKC(10)->SCA   AN-JOK-A16.32
       JOK_SCA_CTT_TextID3: OPTIONAL     TEXTID      //Kernel//;    !! FKC(10)->SCA   AN-JOK-A16.33
       JOK_CTP_TYPE1:      OPTIONAL      BASEID      //Kernel//;    !!                AN-JOK-A15.11
       JOK_COD_CODE1:      OPTIONAL      TEXT*2      //Kernel//;    !!                AN-JOK-A15.12
       JOK_CTP_TYPE2:      OPTIONAL      BASEID      //Kernel//;    !!                AN-JOK-A15.21
       JOK_COD_CODE2:      OPTIONAL      TEXT*2      //Kernel//;    !!                AN-JOK-A15.22
       JOK_CTP_TYPE3:      OPTIONAL      BASEID      //Kernel//;    !!                AN-JOK-A15.31
       JOK_COD_CODE3:      OPTIONAL      TEXT*2      //Kernel//;    !!                AN-JOK-A15.32
       JOK_CKO_Owner:                    OWNER       //Kernel//;    !! UKC(1)->OWN    AN-JOK-I1
       JOK_CK:                           CK5         //Kernel//;    !! UKC(1)         AN-JOK-I2
       JOK_NAME:           OPTIONAL      NAME        //Kernel//;    !!                AN-JOK-A1
       JOK_RPDistBegin:    OPTIONAL      U_ABS       //Kernel//;    !!                AN-JOK-A8
       JOK_RPDistEnd:      OPTIONAL      U_ABS       //Kernel//;    !!                AN-JOK-A12
       JOK_LatDistBegin:   OPTIONAL      NUM2_2      //Kernel//;    !!                AN-JOK-A9
       JOK_LatDistEnd:     OPTIONAL      NUM2_2      //Kernel//;    !!                AN-JOK-A13
       JOK_WidthBegin:     OPTIONAL      NUM2_3      //Kernel//;    !!                AN-JOK-A10
       JOK_WidthEnd:       OPTIONAL      NUM2_3      //Kernel//;    !!                AN-JOK-A14
       JOK_Date_1:         OPTIONAL      DATEDOM     //Kernel//;    !!                AN-JOK-AV1.1
       JOK_Date_2:         OPTIONAL      DATEDOM     //Kernel//;    !!                AN-JOK-AV1.2
       JOK_Date_3:         OPTIONAL      DATEDOM     //Kernel//;    !!                AN-JOK-AV1.3
       JOK_Num_1:          OPTIONAL      NUM38       //Kernel//;    !!                AN-JOK-AV2.1
```

```
      JOK_Num_2:          OPTIONAL     NUM38       //Kernel//;   !!                AN-JOK-AV2.2
      JOK_Num_3:          OPTIONAL     NUM38       //Kernel//;   !!                AN-JOK-AV2.3
      JOK_Num_4:          OPTIONAL     NUM38       //Kernel//;   !!                AN-JOK-AV2.4
      JOK_Char_1:         OPTIONAL     TEXT*72     //Kernel//;   !!                AN-JOK-AV3.1
      JOK_Char_2:         OPTIONAL     TEXT*72     //Kernel//;   !!                AN-JOK-AV3.2
      JOK_Char_3:         OPTIONAL     TEXT*72     //Kernel//;   !!                AN-JOK-AV3.3
      JOK_Char_4:         OPTIONAL     TEXT*72     //Kernel//;   !!                AN-JOK-AV3.4
      JOK_VRS_Code:                    VERSCODE    //Kernel//;   !!                SN-GEN-A1
      JOK_RefDate:                     DATEDOM     //Kernel//;   !!                SN-GEN-D3
      JOK_BeginValidity:               DATEDOM     //Kernel//;   !!                SN-GEN-D1
      JOK_EndValidity:    OPTIONAL     DATEDOM     //Kernel//;   !!                SN-GEN-D2
      JOK_Comment:        OPTIONAL     COMMENT     //Kernel//;
      JOK_ODO_Owner:                   OWNER;                    !! FKL(.)->OWN    SN-GEN-A2
      JOK_OrigSubDBID:                 DBID;                     !!                SN-GEN-A3
      JOK_DAO_Owner:                   OWNER;                    !! FKL(.)->OWN    SN-GEN-A4
      JOK_DataOwner:                   DATAOWNER;                !!                SN-GEN-A5
      JOK_CreateDate:                  DATEDOM;                  !!                SN-GEN-A8
      JOK_ChangeDate:     OPTIONAL     DATEDOM;                  !!                SN-GEN-A9
      JOK_CHO_Owner:                   OWNER;                    !! FKL(.)->OWN    SN-GEN-A10
      JOK_ChangeUser:                  ORAUSER;                  !!                SN-GEN-A11
      JOK_ITS_Code:                    ITSCODE;                  !!                SN-GEN-A6
      JOK_IntegrityDate:               DATEDOM;                  !!                SN-GEN-A7
      JOK_XST_XfrSetID:   OPTIONAL     BASEID;                   !! FKL(.)->XST
    CONSTRAINT
      UNIQUE
        JOK_Baseid, JOK_Version;  !! PK
        JOK_JOT_CKO_Owner, JOK_JOT_CK, JOK_CKO_Owner, JOK_CK;    !! UKC
    END Jokers;



END STRADA_DB.

END STRADA_2_04.                !! End of the model STRADA
!! ====================================================================


!! ====================================================================
FORMAT STRADA;
!! ====================================================================
```