



Fakultät Verkehrswissenschaften „Friedrich List“  
Institut für Verkehrstelematik  
Professur für Verkehrsleitsysteme und -prozessautomatisierung  
Prof. Dr.-Ing. J. Krimmling

Martin Lehmann, Thomas Albrecht

Bericht

„Erarbeitung eines XML-basierten Schemas zur Darstellung der  
sicherungstechnischen Streckenausrüstung in einem Fahrsimulator“

Basierend auf einer Hauptseminararbeit von Martin Lehmann

Kontakte: Martin.Lehmann3 @ mailbox.tu-dresden.de

Thomas.K.Albrecht @ tu-dresden.de

(Telefon: +49 / 351 / 46 33 67 65)

# Inhaltsverzeichnis

ABKÜRZUNGSVERZEICHNIS.....	3
<b>1 EINLEITUNG.....</b>	<b>4</b>
1.1 MOTIVATION .....	4
1.2 EINFÜHRUNG ZUM FAHRSIMULATOR .....	4
1.3 EINFÜHRUNG ZU RAILML.....	5
1.4 ZIELSTELLUNG.....	6
<b>2 DERZEITIGER AUFBAU DES RAILML-SCHEMAS .....</b>	<b>7</b>
2.1 ANMERKUNG .....	7
2.2 RAILML-GRUNDAUFBAU .....	7
2.3 INFRASTRUKTURSHEMA.....	8
2.3.1 <i>Grundschemata</i> .....	8
2.3.2 <i>&lt;line&gt;</i> .....	9
2.3.3 <i>&lt;track&gt;</i> .....	10
2.3.4 <i>&lt;ocsElements&gt;</i> .....	10
2.3.5 <i>&lt;signal&gt;</i> .....	11
2.3.6 <i>&lt;trainProtectionElement&gt;</i> .....	12
<b>3 ERWEITERUNGSVORSCHLAG &lt;INTERLOCKING&gt; .....</b>	<b>14</b>
3.1 EINZUHALTENDE RAILML-ENTWURFSRICHTLINIEN.....	14
3.2 GRUNDLEGENDE DEFINITIONEN .....	15
3.3 NEUES TEILSCHEMA <INTERLOCKING> MIT SEINEN ELEMENTEN .....	16
3.3.1 <i>Einordnung von &lt;interlocking&gt;</i> .....	16
3.3.2 <i>&lt;mainSignalBox&gt;</i> .....	16
3.3.3 <i>&lt;subSignalBox&gt; und &lt;ownControlRange&gt;</i> .....	17
3.3.4 <i>&lt;route&gt;</i> .....	18
3.3.5 <i>&lt;elements&gt;</i> .....	19
3.3.6 <i>&lt;flankProtElements&gt;</i> .....	20
3.4 ERWEITERUNG BESTEHENDER ELEMENTE.....	20
3.4.1 <i>&lt;signal&gt;</i> .....	20
3.4.2 <i>&lt;trainProtectionElement&gt;</i> .....	26

<b>4</b>	<b>PRAXISBEISPIEL</b> .....	<b>28</b>
4.1	FESTLEGUNGEN UND SCHEMATISCHER PLAN.....	28
4.2	DATEN IM BEREICH <INTERLOCKING>.....	29
4.2.1	<i>Einordnung des Stellwerkes</i> .....	29
4.2.2	<i>Fahrstraßentabelle</i> .....	29
4.2.3	<i>Durchrutschwegtabelle</i> .....	31
4.3	DATEN IM BEREICH <LINES>.....	31
4.3.1	<i>Signalliste</i> .....	31
4.3.2	<i>Zugbeeinflussung</i> .....	32
<b>5</b>	<b>ZUSAMMENFASSUNG UND AUSBLICK</b> .....	<b>33</b>
	<b>QUELLENVERZEICHNIS</b> .....	<b>34</b>
	<b>ANHANG A – NEUE UND VERÄNDERTE ELEMENTE IN RAILML</b> .....	<b>35</b>
	<b>ANHANG B – RAILML-CODE FÜR DAS BEISPIEL IN KAPITEL 4</b> .....	<b>42</b>

## Abkürzungsverzeichnis

BSK	Betriebsstellenkennzahl
BZ	Betriebszentrale
CORBA	Common Object Request Broker Architecture
ESTW	Elektronisches Stellwerk
ESTW-A	Abgesetztes ESTW
ESTW-Z	ESTW-Zentrale
ETCS	European Train Control System
Gü	Geschwindigkeitsüberwachung
Ks	Kombinationssignal
LST	Leit- und Sicherungstechnik
LZB	Linienförmige Zugbeeinflussung
OCS	Operation Control System (Leit -und Sicherungstechnik)
PZB	Punktförmige Zugbeeinflussung
vMax	Hier ist immer die zulässige Streckengeschwindigkeit gemeint.
W3C	World Wide Web Consortium
XML	Extensible Markup Language (erweiterbare Auszeichnungssprache)

# 1 Einleitung

## 1.1 Motivation

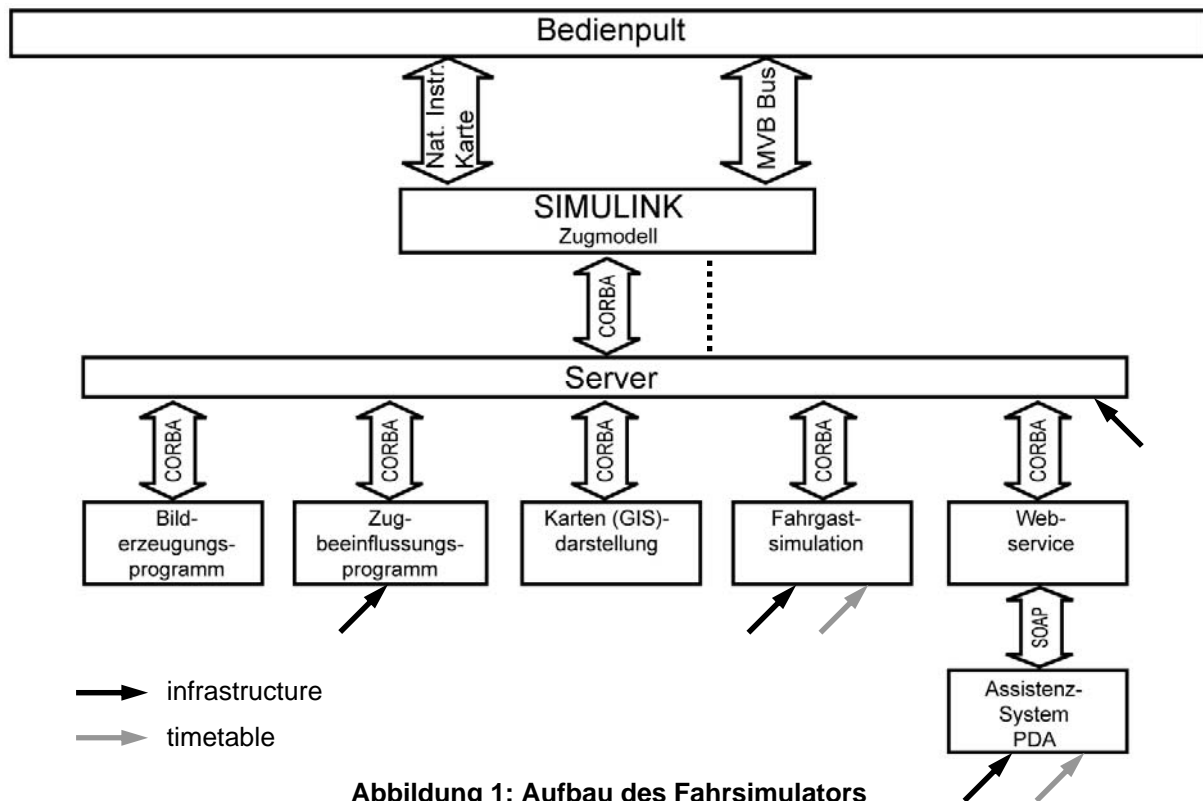
Der am Lehrstuhl Verkehrsleitsysteme und -prozessautomatisierung vorhandene Fahrsimulator wird seit mehr als fünf Jahren zur Entwicklung und Erprobung von Systemen zur Triebfahrzeugführerunterstützung eingesetzt. Für solche Systeme gewinnt die Berücksichtigung der aktuellen Betriebssituation zunehmend an Bedeutung. Hierzu muss der Fahrsimulator mit einer synchron ablaufenden Betriebssimulation gekoppelt werden. Zwischen beiden Systemen muss eine Vielzahl von Daten zum Zustand der Leit- und Sicherungstechnik ausgetauscht werden. Da der Datenaustausch innerhalb des Fahrsimulators sowie im Fahrerassistenzsystem ENAFlex-S bereits den RailML-Standard umsetzt, sollte dieser auch bei der Kopplung Fahr-/ Betriebssimulation zum Einsatz kommen.

Das aktuelle RailML-Schema bietet hierfür kaum Möglichkeiten. Eine Erweiterung um ein Teilschema, welches dieser Problematik Rechnung trägt wird seit längerem in den entsprechenden Foren (im RailML.org-Forum, bei Treffen der RailML.org-Initiative und an den verkehrswissenschaftlichen Lehrstühlen) diskutiert. Diese Arbeit soll für die Einbindung der Sicherungstechnik in RailML einen Beitrag leisten und einen Erweiterungsvorschlag unterbreiten.

## 1.2 Einführung zum Fahrsimulator

Der am Lehrstuhl vorhandene Eisenbahn-Fahrsimulator ist ein Server-Client-System. Die einzelnen Komponenten der Simulation laufen in verschiedenen Programmen unabhängig von einander ab und kommunizieren über eine CORBA-Schnittstelle mit dem Fahrsimulatorserver. Das Zugmodell (Simulink-Programm) ist über den MVB Bus mit einem realen Bedienpult der Baureihe 424 verbunden. Die Fahrbefehle des Bedienpultes werden im Zugmodell verarbeitet und die benötigten Daten an die Programme weitergegeben. Ebenso steuert das Zugmodell über den MVB Bus die Anzeigen und Leuchtmelder des Bedienpultes an.

Das System bzw. die einzelnen Programme des Fahrsimulators benötigen verschiedene Grunddaten. Im Moment werden Infrastruktur- und Fahrplandaten in RailML-Dateien bereitgestellt, die Nutzung von RailML für Fahrzeugdaten ist in Vorbereitung. In Abbildung 1 ist die Grundstruktur des Fahrsimulators und der Dateneinsatz im RailML-Format dargestellt.



**Abbildung 1: Aufbau des Fahrsimulators**

### 1.3 Einführung zu RailML

Mit steigender Zahl an Computerprogrammen im Eisenbahnbetriebs- und Eisenbahninfrastrukturbereich stieg auch die Zahl der Schnittstellen zwischen den entsprechenden Programmen. Um diesen Informationstausch zu vereinfachen, bedarf es standardisierter Datenformate. Für die Entwicklung eines solchen Formates entstand 2002 die RailML.org-Initiative. Die Initiative besteht aus gleichberechtigten Partnern von Eisenbahnunternehmen, Softwareunternehmen, unabhängigen Forschungsinstituten und universitären Einrichtungen. Deren Open Source Projekt brachte das XML-Schema RailML hervor [2].

Das gewählte Format XML spielt heute eine bedeutende Rolle für den standardisierten Datenaustausch. Die Spezifikationen für XML werden vom W3C herausgegeben. XML ist eine Metasprache, mit deren Hilfe die Anwender spezifische Sprachen definieren können. RailML stellt eine solche anwendungsspezifische Sprache dar, mit der man eisenbahnspezifische Daten beschreiben kann [8].

Seit den Anfängen von RailML sind verschiedene Versionen veröffentlicht worden. Diese Arbeit basiert auf Version 1.1 vom 21.10.2007 [7].

## 1.4 Zielstellung

Es soll ein Vorschlag für ein Schema im Bereich Sicherheitstechnik - mit dem Hauptaugenmerk auf die Signalisierung - ausgearbeitet werden. Die Anforderungen einer Kopplung von Fahrsimulation und Betriebssimulation sind bei der Konzeption des Vorschlages zu berücksichtigen. Im Vordergrund steht dabei vor allem, dass dem Fahrsimulator in Zukunft die nötigen Daten für die Auswahl der Signalbegriffe übergeben werden können, damit diese entsprechend der betrieblichen Situation dem zur Darstellung der Umgebung genutzten Video überlagert werden können. Die Daten sind abhängig von den eingestellten Fahrstraßen bzw. Blockbelegungen. Der Vorschlag für die RailML-Erweiterung soll nun die sinnvolle Auswahl des zu überblendenden Signalbegriffes ermöglichen. Sicherungstechnische Grunddaten sind dafür zwingend notwendig.

Aus diesem Grund soll der Vorschlag ein neues Teilschema `<interlocking>` enthalten aber auch bereits existierende Elemente erweitern. Die zu dieser Thematik im RailML.org-Wiki bereits veröffentlichten Anmerkungen, wurden im Erweiterungsvorschlag weitestgehend berücksichtigt [9].

Abschließend soll der Einsatz des ausgearbeiteten Schemas an einem Praxisbeispiel demonstriert werden.

Das RailML-Schema wird mit dem Programm XMLSpy von Altova bearbeitet. Die Abbildungen der Schemata in dieser Arbeit sind Screenshots aus diesem Programm.

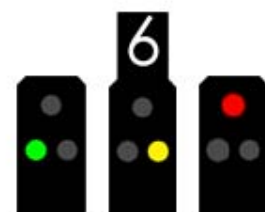
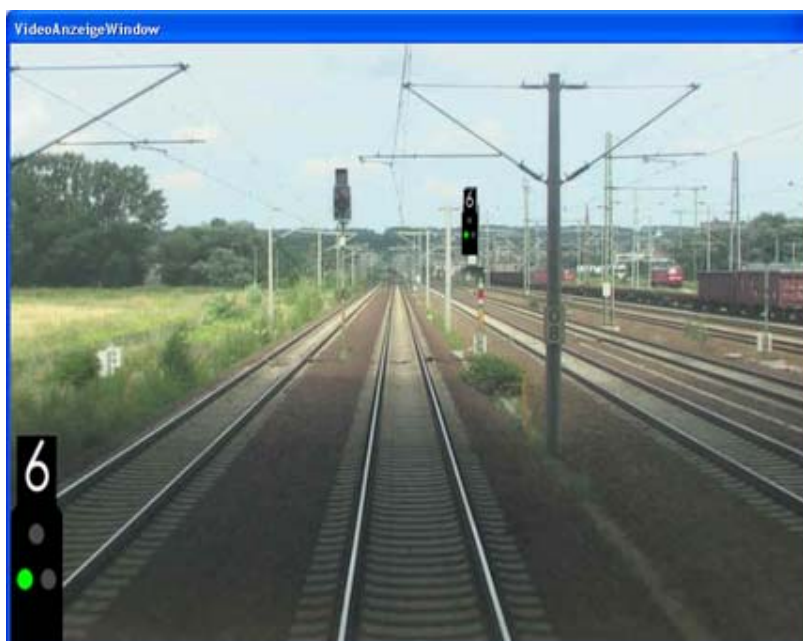


Abbildung 2: Anzeige des Videotools mit überblendetem Signalbegriff (links), weitere mögl. Signalbegriffe, die überblendet werden können (rechts)

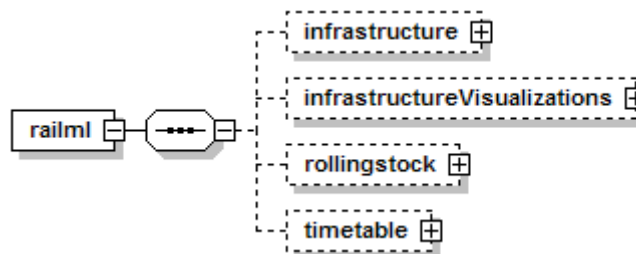
## 2 Derzeitiger Aufbau des RailML-Schemas

### 2.1 Anmerkung

Ziel dieses Kapitels ist es, den grundsätzlichen Aufbau von RailML darzustellen, damit der später dargestellte Erweiterungsvorschlag nachvollzogen und eingeordnet werden kann. Komponenten von RailML, die hierfür irrelevant oder von geringer Bedeutung sind, werden nicht betrachtet.

### 2.2 RailML-Grundaufbau

Die aktuelle RailML-Version 1.1 besteht aus vier Teilschemata, wie in Abbildung 3 dargestellt.



**Abbildung 3: Gliederung des RailML-Schemas in die Teilschemata**

Die Eisenbahninfrastruktur wird mit einer Vielzahl an Unterelementen in dem Teilschema `<infrastructure>` abgebildet.

Zur visuellen Darstellung von Infrastruktur in Simulations- oder Planungswerkzeugen werden Visualisierungsdaten benötigt, die getrennt von dem eigentlichen Infrastrukturbereich im Teilschema `<infrastructureVisualizations>` bereitgestellt werden.

Das Schema `<timetable>` widmet sich der Fahr- und Betriebsplanung und das Schema `<rollingstock>` dem rollenden Material (angetriebene wie auch nicht angetriebene Fahrzeuge) [8].

Im Folgenden wird auf das Subschema `<infrastructure>` detailliert eingegangen. Die drei anderen genannten Teilschemata werden nicht weiter betrachtet. In den Abbildungen zur Veranschaulichung der Elemente wird darauf verzichtet, die Elementarattribute mit abzubilden. Allgemeine Attribute, wie `<name>` (ausgeschriebener Name des Objekt), `<ID>` (eindeutige Bezeichnung, Identifier) und `<type>` (Bauart, Ausführung), werden nicht jedes Mal genannt, da sie bei fast allen Elementen vorhanden sind und immer gleich angewendet werden. Sonderattribute werden an den entsprechenden Stellen genannt und erläutert.



Generell ist zum Aufbau zu bemerken, dass Elemente durch bestimmte Typdefinitionen beschrieben werden. In Tabelle 1 sind die unterschiedlichen Typdefinitionen aufgelistet.

**Tabelle 1: Allg. Typdefinitionen**

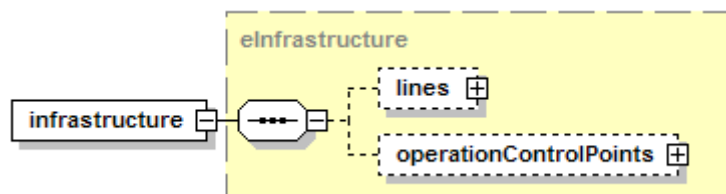
Nomenklatur	Beschreibung	Beispiel
eXxxYyy	Komplexe Typdefinition, die die Grundstruktur eines Elementes beschreibt.	<eInfrastructure>
tXxxYyy	Komplexe oder einfache Typdefinition, die Basis für eine weitere Typdefinition ist.	<tGenericID>
aXxxYyy	Definiert eine Gruppe von Attributen, die einer Typdefinition zugeordnet wird.	<aSignal>

Einfache Typdefinitionen enthalten generell nur Attribute, wohingegen komplexe Definitionen mindestens für ein weiteres Unterelement als Wurzel dienen und zusätzliche Attribute enthalten können [8].

## 2.3 Infrastrukturschema

### 2.3.1 Grundschema

Das Infrastrukturschema teilt sich in zwei Subschemata, wie in Abbildung 4 dargestellt.



**Abbildung 4: Aufbau des Elementes <infrastructure>**

Im Bereich <operationControlPoints> befindet sich eine Liste der Bahnhöfe und Haltepunkte. Die Informationen, die hier hinterlegt werden können, sind <ID>, <name> und <number> (Kennzahl) der Betriebsstellen und entsprechende betriebliche Aspekte. Halte werden in diesem Teilschema in ihrer Funktion als Schnittstelle zu Bahnkunden dargestellt, sicherungstechnische Informationen sind aus diesem Grund im Unterschema <operationControlPoints> nicht enthalten.

Der überwiegende Teil an Infrastrukturdaten befindet sich im Bereich `<lines>`. Dieses Subschema enthält Informationen zu den Strecken. Dabei wird ein Knoten-Kanten-Modell zugrunde gelegt. Es werden Punkte entlang der Kilometrierungsachse definiert, die Änderungen mindestens eines Attributwertes beinhalten. Die Angaben sind immer richtungsabhängig. Das Streckenschema enthält auch Elemente, die keine Änderungen bezüglich des weiteren Streckenverlaufes enthalten, z. B. Signale, Weichen, Kreuzungen oder Gleisschaltmittel. Diese Elemente werden auch als Knoten modelliert. In den folgenden Teilabschnitten werden diverse Unterelemente von `<lines>` vorgestellt, auf die sich das sicherungstechnische Schema bezieht und die deswegen für das Verständnis der Erweiterung erforderlich sind.

### 2.3.2 `<line>`

Das Element `<line>` wird durch den Typ `eLines` dem Element `<lines>` zugeordnet (siehe Abbildung 5). In `<lines>` können eine Vielzahl von Strecken mit gleichen Eigenschaften hinterlegt werden. Die gemeinsamen Eigenschaften der Streckenabschnitte werden `<infraAttrGroup>` gespeichert, z. B.: Elektrifizierung, Eigentümer, Energieversorgung, Achslast, etc.

Das Element `<line>` beinhaltet eine Auflistung mehrerer Gleise, die im Element `<tracks>` hinterlegt sind. Generell sollen somit parallele Gleise zusammengefasst werden, die einer Bahnstrecke angehören. In `<lineDescr>` (description in dt. Beschreibung) können Erklärungen oder Kommentare zur Strecke erfolgen und `<infraAttrGroupID>` dient zur Referenzierung übergeordneter Infrastrukturmerkmale.

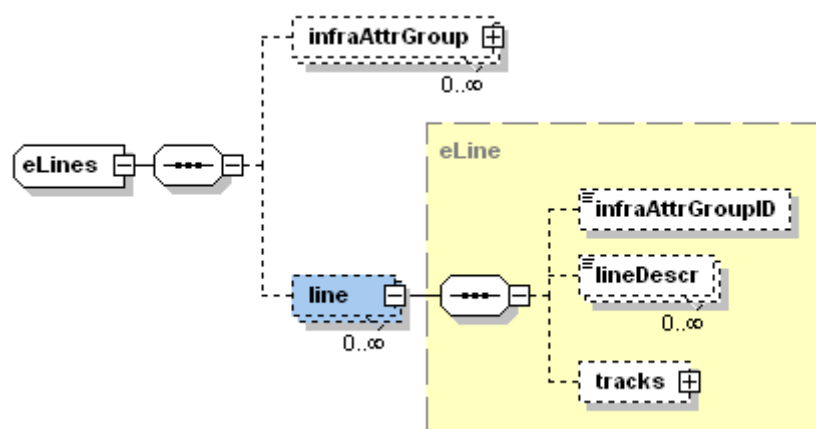


Abbildung 5: Aufbau des Elementes `<lines>`

### 2.3.3 <track>

Das Gleiselement `<track>` bildet einen Gleisabschnitt mit einem definierten Start- und Endpunkt ab. Das übergeordnete Element `<tracks>` beinhaltet die Liste der Gleisabschnitte. In Abbildung 6 ist die Einordnung von `<track>` und seiner Unterelemente dargestellt.

In `<trackTopology>` sind immer der Startpunkt `<trackBegin>` und der Endpunkt `<trackEnd>` enthalten. Es können auch noch weitere topologische Eigenschaften hinterlegt werden: `<connections>` (Verbindung zu anderen Gleisen), `<mileageChanges>` (Kilometrierungsprünge), `<crossSections>` (Kreuzungsbereiche) und `<borders>` (Grenzen). Jedem Gleis können bestimmte Gleiselemente in `<trackElements>` zugeordnet werden, unter anderem: `<speedChanges>` (Geschwindigkeitswechsel), `<gradientChanges>` (Steigungswechsel), `<tunnels>` (Tunnel), `<levelCrossings>` (Bahnübergänge) und noch einige weitere.

Das für diese Arbeit relevante Unterelement von `<track>` ist `<ocsElements>` (Elemente der Leit- und Sicherungstechnik).

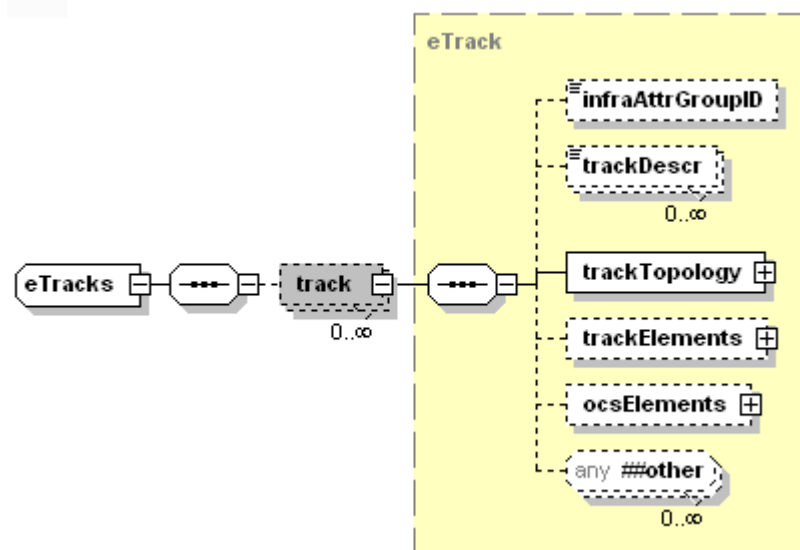


Abbildung 6: Aufbau des Elementes `<tracks>`

### 2.3.4 <ocsElements>

Das Element enthält die Beschreibung streckenseitiger Komponenten der Leit- und Sicherungstechnik, dabei werden in den Unterelementen `<signals>` einzelne Signale oder Signalgruppen, in `<trainDetectionElements>` Zugerfassungselemente wie Achszähler, Gleisstromkreise o. ä., in `<balises>` Balisen oder Balisengruppen und in `<trainProtectionElements>` streckenseitige

Komponenten der Zugbeeinflussung hinterlegt (siehe Abbildung 7). Sicherungstechnische Abhängigkeiten oder Zugehörigkeiten zu bestimmten Stellwerken werden nicht abgebildet. Es handelt sich um eine reine Auflistung und Klassifizierung von Elementen. Diverse Attribute bieten die Möglichkeit, die Elemente allgemeingültig zu beschreiben. In jedem Fall sind die Attribute Streckenposition, Name und ID zu benutzen, wohingegen zusätzlich beschreibende Attribute optional sind.

Im Folgenden wird auf die Elemente `<signal>` und `<trainProtectionElement>` eingegangen, da diese in der Infrastrukturdatei des Fahrsimulators genutzt werden.

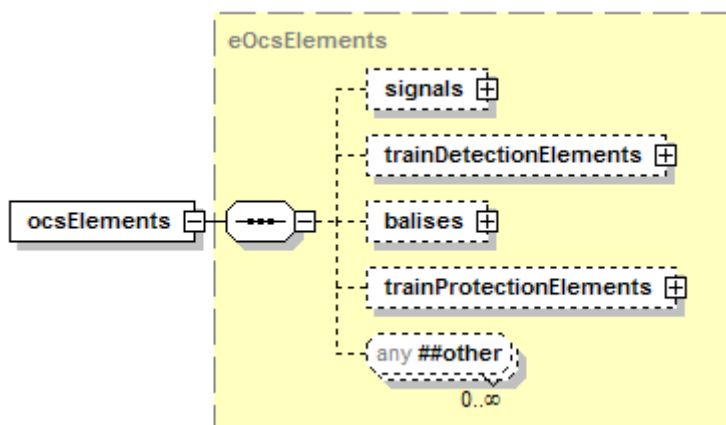


Abbildung 7: Aufbau des Elementes `<ocsElements>`

### 2.3.5 `<signal>`

Das Element `<signal>` hat nur das Unterelement `<geoCoord>`, in dem die geografische Lage (geografische Breite und Länge) des jeweiligen Signals gespeichert werden kann. Es kann der Liste `<signals>` in Form von einzelnen Signalen oder in Signalgruppen untergeordnet werden (siehe Abbildung 8). Das Signalelement besitzt eine Vielzahl von Attributen zur Beschreibung, diese werden in Tabelle 2 vorgestellt.

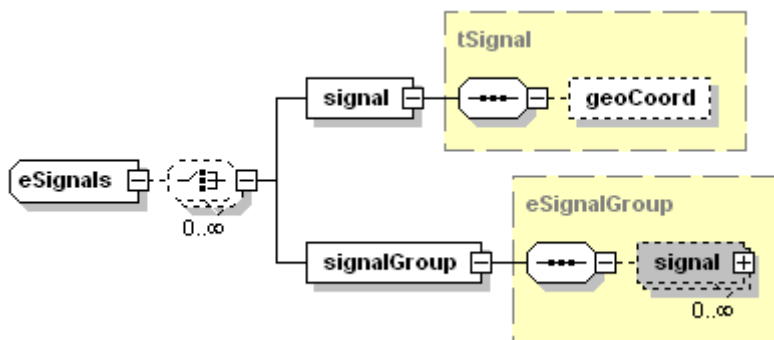


Abbildung 8: Aufbau des Elementes `<signals>`

**Tabelle 2: Attribute des Elementes <signal>**

Attribut	Erklärung	Wert
id	Identifikationsnummer	String
name	Signalname	String
pos	Streckenposition	Meterangabe
absPos	absolute Streckenposition	Meterangabe
absPosOffset	Streckenpositionsversatz	Meterangabe
dir	Richtung	none, up, down, both
sight	Sichtweite	Meterangabe
type	Art	main, distant, combined
function	Funktion	exit, home, blocking
sigSystem	Signalsystem	HI, Ks, HV (DB) L, N (SBB)
switchable	Schaltbarkeit	Boolesch
maskableRoute		Boolesch
maskableATC		Boolesch
virtual	Physisch vorhanden	Boolesch
signalBoxID	Stellwerksidentifikationsnummer	Generische Id
stationID	Bahnhofsidentifikationsnummer	Generische Id
distNearestDangerPoint	Gefahrpunktabstand	Meterangabe
trackDist	Abstand zum Gleis	Meterangabe
height	Signalhöhe	Meterangabe

Die erhebliche Anzahl an Attributen verdeutlicht die Komplexität des Elements in der RailML-Struktur. Trotz der Gestaltungsmöglichkeiten durch die vorhandenen Attribute sind verschiedene Eigenschaften und Sachverhalte von Signalen bisher nicht berücksichtigt, vor allem die fehlende Dokumentationsmöglichkeit der Signalbegriffe, die angezeigt werden können, ist hier hervorzuheben. Gerade im Bezug der Entwicklung eines sicherungstechnischen Teilschemas gilt es dies zu ergänzen. Im Kapitel 3 wird hierzu ein Lösungsansatz vorgeschlagen.

### 2.3.6 <trainProtectionElement>

In diesem Element können streckenseitige Komponenten der Zugbeeinflussung hinterlegt werden. Das Element hat, ähnlich dem Signal, nur das Unterelement <geoCoord> und einige Attribute zur Beschreibung. Die wichtigste Eigenschaft ist neben der Bezeichnung <ID> die Längslage <pos> im Gleis. Weitere Attribute sind

`<dir>` (Richtung), `<medium>` (Übertragungsart z. B. mechanisch, magnetisch, Funk usw.), `<system>` (technisches System z. B. PZB oder ETCS L1), `<model>` (Modelltypbezeichnung), `<absPos>` (absolute Streckenposition) und `<absPosOffset>` (Streckenpositionsversatz).

Eine Zuordnung des Elementes zu einer Steuereinheit erfolgt nicht. Das Problem wird ebenfalls im folgenden Kapitel aufgegriffen.

## 3 Erweiterungsvorschlag <interlocking>

### 3.1 Einzuhaltende RailML-Entwurfsrichtlinien

Das Erweitern eines Schemas unterliegt den allgemeinen Entwurfsrichtlinien, die von der RailML.org-Initiative aufgestellt wurden. Sie wurden in der „Bildungsvorschriften für RailML-Teilschemata“ [8] veröffentlicht. Es sind sowohl allgemeine Anforderungen als auch spezielle Designgrundsätze zu berücksichtigen.

#### allgemeine Anforderungen [8]:

- Die Gültigkeit bestehender Versionen des Schemas darf nicht eingeschränkt werden.
- Zukünftige Erweiterungen sollen nicht behindert oder erschwert werden.
- Es sollen eindeutige Bezeichnungen gewählt werden und bereits verwendete Bezeichnungen übernommen werden.
- Das Schema ist so allgemeingültig zu halten, dass es keine Anwender ausschließt.
- Die RailML-Designgrundsätze sind einzuhalten.

#### Designgrundsätze [8]:

- Das Schema soll wohl geformt und anhand von standardisierten Schemata validierbar sein (siehe auch Bildungsvorschriften W3C für XML-Schemata [10]).
- Alle Elemente des Schemas sind ausschließlich in Englisch zu erstellen.
- Aus einem Wort bestehende Bezeichner werden kleingeschrieben.
- Aus mehreren Worten bestehende Bezeichner werden in der Kamel-Schreibweise gebildet (Bsp.: „signalBox“).
- Bezeichner mit nur zwei Buchstaben werden mit Großbuchstaben gekennzeichnet (Bsp.: „ID“).
- Selbst definierten Datentypen wird der Zusatz Type angehängt.
- Bezeichnernamen sollen keine Abkürzungen oder Komposita enthalten.
- Es sollen nur Akronyme, die allgemein in der Computer- bzw. Bahntechnik anerkannt sind, verwendet werden.
- Auch für Akronyme soll die Kamel-Schreibweise benutzt werden.

### **3.2 Grundlegende Definitionen**

Die obersten Instanzen der Beschreibung der Leit- und Sicherungstechnik sind Stellwerke. Stellwerke sind Bestandteile der ortsfesten Infrastruktur. Elemente wie z. B. die Fahrstraße oder der Durchrutschweg sind zwar keine physischen Infrastruktureinheiten, jedoch enthalten sie Referenzen auf Infrastrukturelemente wie z. B. Weichen, Signale und Gleisabschnitte.

Es wird daher vorgeschlagen, das neu zu entwickelnde <interlocking> (LST)-Schema als Unterelement des Elementes <infrastructure> einzuführen.

Zentrale Aufgabe der Sicherungstechnik ist es, Zugfahrten und Rangierfahrten in den Fahrwegen zu sichern. Grundsätzlich unterscheidet man in Deutschland die Technologien Fahrstraße für Bahnhöfe und Blockinformation für die freie Strecke. Die Vereinheitlichung und Vereinfachung der Technik und auch der Betriebsführung in den letzten Jahren hat dazu geführt, dass in modernen ESTW auch die freie Strecke mit der Technologie Fahrstraße gesichert wird [1]. Eine ähnliche Tendenz ist auch in anderen europäischen Ländern zu beobachten.

Aus diesem Grund wird im Erweiterungsvorschlag nicht zwischen Streckenblock und Fahrstraße unterschieden. Der Streckenblock wird wie eine Fahrstraße behandelt.

Die Fahrstraßenabbildung folgt dem tabellarischen Prinzip, d.h. dass alle Fahrwegelemente- und Flankenschutzelemente einer einzelnen Fahrstraße in einer Auflistung zusammengefasst werden [6].

Der hier vorgestellte Erweiterungsvorschlag hat nicht den Anspruch auf Vollständigkeit, sondern soll der Aufgabe gerecht werden „ein Schema im Bereich Signalisierung zu erarbeiten, mit dem die Anforderungen der Kopplung Fahr Simulator/ Betriebssimulation erfüllt werden können“ [Themenblatt der Hauptseminararbeit].



### 3.3 Neues Teilschema <interlocking> mit seinen Elementen

#### 3.3.1 Einordnung von <interlocking>

Das Element <interlocking> wird als Unterelement von <infrastructure> parallel zum Strecken- und Betriebsstellenelement angeordnet. Das englische Wort „interlocking“ kann mit der deutschen Bezeichnung Leit- und Sicherungstechnik (LST) gleichgesetzt werden. Der komplexe Typ <eInfrastructure> wurde wie in Abbildung 9 zu sehen verändert.

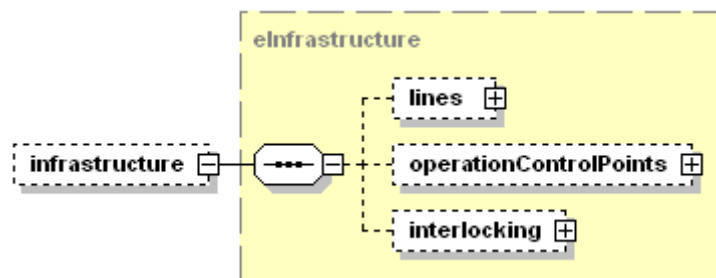


Abbildung 9: Veränderung des Elementes <infrastructure>

#### 3.3.2 <mainSignalBox>

Das Element <mainSignalBox> (Hauptstellwerk) (siehe Abbildung 10) enthält die oberste Instanz des Stellwerks, ihm können weitere Elemente vom Typ <subSignalBox> hinzugefügt werden.

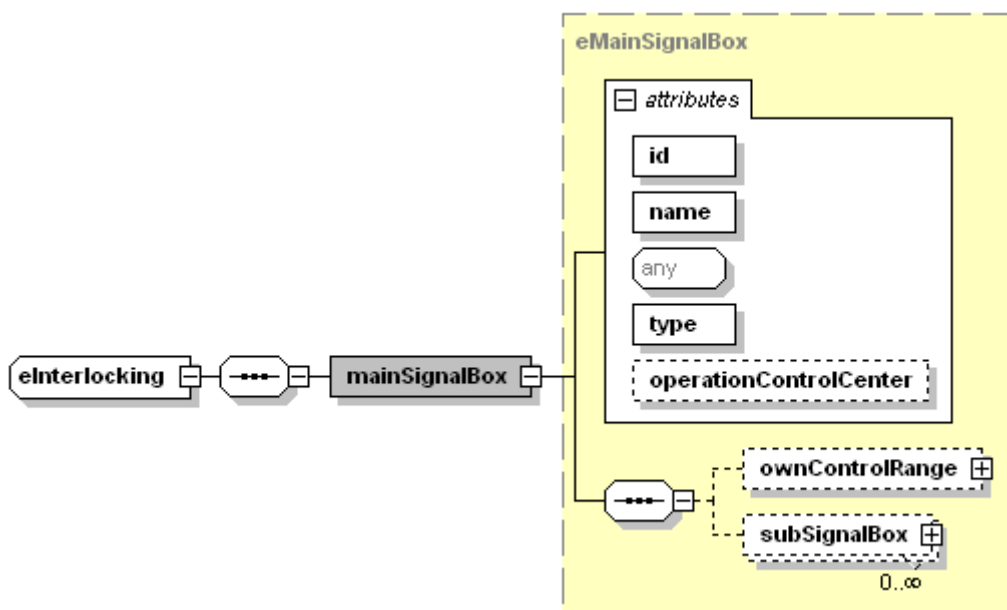


Abbildung 10: Neues Element <mainSignalBox>

Im Fall eines ESTW entspricht die ESTW-Zentrale dem Hauptstellwerk. Der Zentrale untergeordnet sind die ESTW-A. Bei mechanischen bzw. elektromechanischen Stellwerken ist das Fahrdienstleiterstellwerk das Hauptstellwerk und ihm untergeordnet sind diverse Weichenwärterstellwerke. Bei Relaisstellwerken existiert meist nur ein Stellwerk, in einem solchen Fall wird keine Liste mit untergeordneten Stellwerken angelegt. Das Attribut <type> beschreibt die eingesetzte Stellwerkstechnik. Hat das Hauptstellwerk einen eigenen Stellbereich, so ist dieser im Element <ownControlRange> (eigener Stellbezirk) einzutragen. Ist das nicht der Fall, wie z. B. bei einem ESTW-Z, braucht <ownControlRange> nicht gepflegt werden.

Wird ein Hauptstellwerk einer Betriebszentrale zugeordnet, sollte die BZ-Zugehörigkeit in dem Attribut <operationControlCenter> hinterlegt werden. Auf die nachfolgend beschriebene Struktur hat diese Zuordnung keinen Einfluss, da nur das Stellwerk selbst abgebildet wird, und nicht die Technik, die auf das Stellwerk einwirkt [5].

In Deutschland sind Stelleinheiten (Signale, Weichen, Gleisfreimeldeabschnitte, etc.) üblicherweise einer Betriebsstelle zugeordnet, die durch die sogenannte Betriebsstellenkennzahl BSK eindeutig identifiziert werden kann. Den Stellelementen werden bei der Zuordnung zum ESTW-A die entsprechenden BSK zugeordnet. In der vorliegenden RailML-Implementierung sind Stelleinheiten den jeweiligen Gleisen <track> zugeordnet. In dieser Arbeit wird auf ein Attribut BSK für das ESTW-A und die Stelleinheiten zunächst verzichtet. Es ist in späteren Arbeiten zu prüfen, wie diese Zuordnung optimal dokumentiert werden sollte, wenn RailML für den Stellwerksentwurf eingesetzt wird.

### 3.3.3 <subSignalBox> und <ownControlRange>

Die beiden Elemente besitzen die gleichen Unterelemente und unterscheiden sich lediglich darin, dass das Element <ownControlRange> keine Attribute besitzt. Aus diesem Grund ist in Abbildung 11 nur der Ordner <subSignalBox> dargestellt.

Die Unterelemente enthalten die Listen der Fahrstraßen (in <routes>), Freimeldeabschnitte (in <trackSections>), Durchrutschwege und Gefahrenpunkteabstände<sup>1</sup> (in <overlaps>) und Nahbedienbereiche (in <localOpAreas>).

---

<sup>1</sup> Wird im Folgenden von Durchrutschwegen gesprochen sind immer auch die Gefahrenpunkteabstände impliziert.

In dieser Arbeit liegt der Fokus der Entwicklung auf dem Element <routes>, also in der Hinterlegung der Fahrstraßen. Die Elemente <trackSections> und <overlaps> werden nur kurz besprochen und sind nicht vollständig entwickelt worden, da sie für Anwendung des Fahrsimulators nur geringe Bedeutung besitzen. Die Nahbedienbereiche sind hier nur der Vollständigkeit halber genannt. Sie besitzen für den Fahrsimulator keine Relevanz und werden ebenfalls nicht weiter betrachtet.

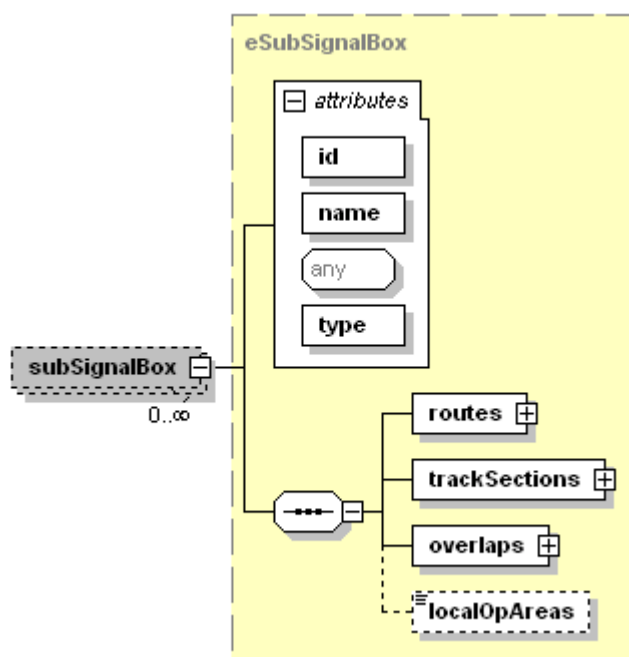


Abbildung 11: Neues Element <subSignalBox>

### 3.3.4 <route>

Die Liste <routes> enthält die <route> Einzelelemente. In Abbildung 12 ist der Aufbau dargestellt. Die Attribute <id> und <vMax> sind obligatorisch. Das Attribut <vMax> ist dabei die höchstzulässige Geschwindigkeit, die sich aus der Gleis- bzw. Weichengeometrie ergibt. Die maßgebende Geschwindigkeit für eine Fahrstraße ergibt sich aus dem Minimum dieser Geschwindigkeit und der durch die Durchrutschwege <overlaps> bestimmten Geschwindigkeit. Die korrekte Ermittlung der restriktivsten Geschwindigkeit muss in der Anwendung geschehen, die die RailML-Daten verarbeitet und nutzt.

Die Attribute <setupTime> (Fahrstraßenbildezeit), <releaseTime> (Fahrstraßenauflösezeit), <length> (Länge) und <type> (Art z. B.: Rangierfahrstraße, Zugfahrstraße, Streckenblock usw.) sind fakultativ. Weitere Attribute von Fahrstraßen sind für die zu untersuchende Anwendung nicht notwendig, sie müssen gegebenenfalls in weiterführenden Arbeiten ergänzt werden.

Der Anfang der Fahrstraßen wird in <routeStart> hinterlegt und enthält das Attribut <refID> mit der Referenz auf ein bestimmtes Objekt. Über das Attribut <type> kann die Art des Fahrstraßenstartpunktes definiert werden (z. B. Signal, Sperrsignal, Trapeztafel, ...). Das Fahrstraßenziel <routeDestination> ist genauso aufgebaut. Beide Elemente sind vom Typ <tRouteElement>.

Die Gesamtheit aller Durchrutschwege eines Stellwerkes wird als Durchrutschweg-tabelle im Unterelement <overlaps> des Elementes <subSignalBox> bzw. <ownControlRange> hinterlegt. Die für die Fahrstraße relevanten Durchrutschwege<sup>2</sup> werden für das Element <route> in seinem Unterelement <overlaps> referenziert. Äquivalent wird mit den Gleisfreimeldeabschnitten (im Element <trackSections>) verfahren. Die Unterelemente <elements> und <flankProtElements> werden in den folgenden Abschnitten eingeführt.

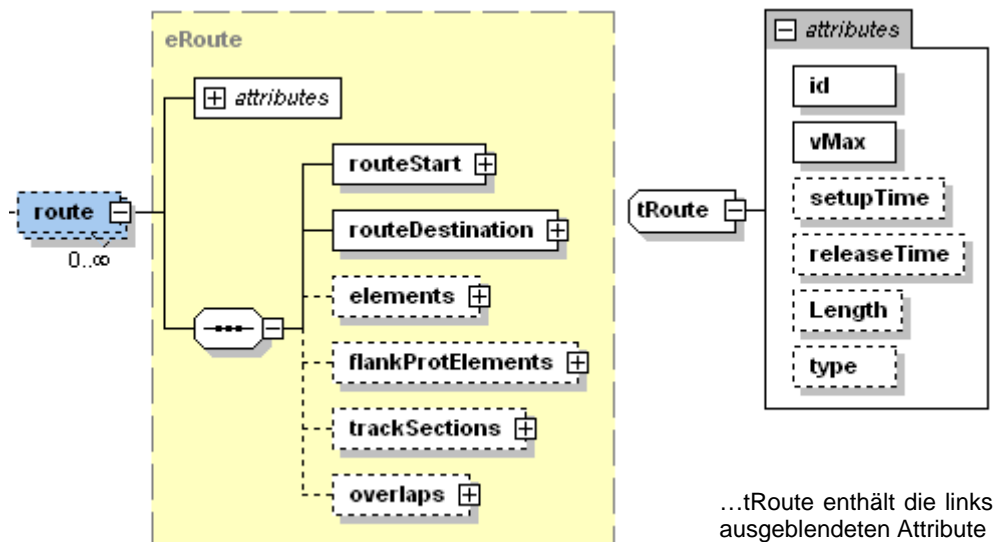


Abbildung 12: Neues Element <route>

### 3.3.5 <elements>

Der Ordner enthält eine Liste der Elemente, die sich in der Fahrstraße befinden. Das entsprechende Element kann ausgewählt werden und wird über das Attribut <refID> referenziert. Den Auswahlelementen sind weitere Attribute zugeordnet, um die Lage (bei einer Weiche) oder den Typ (bei einem Signal) festzulegen. Mögliche Auswahlelemente sind: <switch> (Weiche), <signal> (Signal), <crossing> (Kreuzung), <levelCrossing> (Bahnübergang), <derailer> (Gleissperre) und <other> (sonstige). In Abbildung 13 ist das Element dargestellt.

<sup>2</sup> Mindestens ein Durchrutschweg muss referenziert werden. Bei Verwendung von Wahldurchrutschwegen müssen alle referenziert werden.

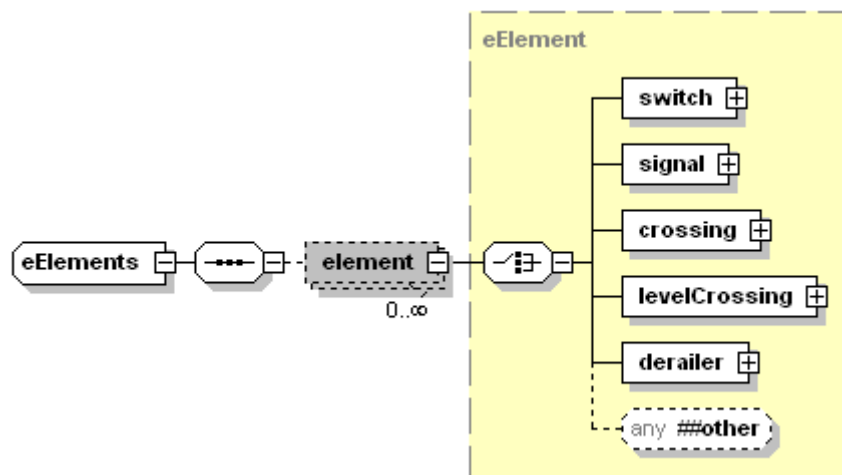


Abbildung 13 Neues Element &lt;elements&gt;

### 3.3.6 <flankProtElements>

Das Element ist ähnlich wie das eben beschriebene Element aufgebaut (siehe Abbildung 14). Es enthält die Liste der Flankenschutzelemente, die zu der Fahrstraße gehören. Sollte es keinen Flankenschutz geben (z. B. bei Rangierstraßen), braucht das Element nicht gepflegt zu werden.

Die Auswahlelemente sind <switch> und <derailer>, wie schon in <element> und <redSignal> (Schutz durch Halt zeigendes Signal).

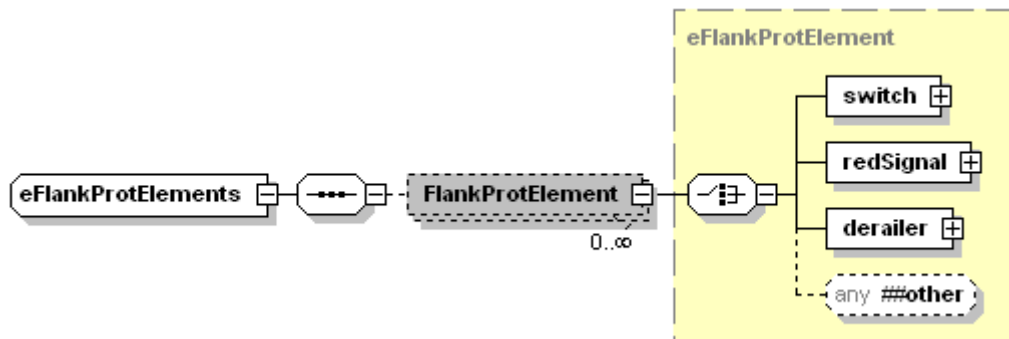



Abbildung 14 Neues Element &lt;flankProtElements&gt;

## 3.4 Erweiterung bestehender Elemente

### 3.4.1 <signal>

Die Festlegung der Funktion und Art eines Signals und auch die Zuordnung zu einem Signalsystem werden im vorliegenden RailML-Standard unterstützt. Dadurch können Rückschlüsse auf die Art der Abschnittssignalisierung (Ein-, Zwei- oder Mehrabschnittssignalisierung) und damit die zulässigen Signalbegriffe gezogen werden. Die konkreten Geschwindigkeiten, die für die jeweiligen Abschnitte signalisiert werden können, sind jedoch nicht darstellbar. Das Beispiel in Tabelle 3 veranschaulicht die Problematik.

**Tabelle 3: Signalelement vor der Erweiterung**

<b>&lt;signal&gt;</b>			Werte	Bemerkungen
	Mehrab-			
	schnittssignal	id:	64P1	
	mit Zs3v für	name:	Ausfahrtsignal P1	
	60/40 km/h	type:	combined	Kombinationssignal
	und Zs3 für	function:	Exit	Ausfahrtsignal
80 km/h	sigSystem:	Ks	Ks-System der DB	

Es handelt sich um ein Ausfahrtsignal, welches sowohl Vor- als auch Hauptsignal-eigenschaften besitzt und zum Signalsystem Ks der DB gehört, also eine Zweiabschnittssignalisierung realisiert. Die folgenden drei Signalbegriffe können aus diesen Angaben abgeleitet werden:

- Hp0:                    rotes Dauerlicht            Halt
- Ks1:                    grünes Dauerlicht            Fahrt mit vMax
- Ks2:                    gelbes Dauerlicht            Halt erwarten

Das Signal aus Tabelle 3 kann weiterhin noch die folgenden Signalbegriffe anzeigen, die nicht aus den gegebenen Werten geschlossen werden können:

- Ks1+Zs3                grünes Dauerlicht            Fahrt mit  $v = \text{Zahl}_{\text{weiß}} \times 10 \text{ km/h}$   
mit weißer Zahl
- Ks1+Zs3v                grünes Blinklicht            Fahrt mit vMax und Vorankündigung  
mit gelber Zahl                Fahrt mit  $v = \text{Zahl}_{\text{gelb}} \times 10 \text{ km/h}$
- Ks1+Zs3+Zs3v            grünes Blinklicht            Fahrt mit  $v = \text{Zahl}_{\text{weiß}} \times 10 \text{ km/h}$  und  
mit weißer und                Vorankündigung Fahrt mit  
gelber Zahl                         $v = \text{Zahl}_{\text{gelb}} \times 10 \text{ km/h}$
- Ks2 +Zs3                gelbes Dauerlicht            Fahrt mit  $v = \text{Zahl}_{\text{weiß}} \times 10 \text{ km/h}$  und  
mit weißer Zahl                Vorankündigung Halt am nächsten  
Signal

Um Geschwindigkeitsbegriffe abzubilden wurde zunächst die Möglichkeit eines Unterelements „Zusatzanzeiger“ in Betracht gezogen, das mögliche Geschwindigkeitsbegriffe explizit enthalten hätte. Bei den unterschiedlichen Bahnverwaltungen existiert jedoch eine nahezu unüberschaubare Anzahl solcher Begriffe und Zusatzanzeiger [3]. Eine direkte Implementierung der Zusatzanzeiger hätte zur Folge, dass sämtliche auf die Daten zugreifenden Programme eine Interpretation der Begriffe durchführen müssten und dazu eine große Menge an nationalen Regeln vorhalten müssten. Dies hätte nicht den Designgrundsätzen von RailML entsprochen.

Es wurde deshalb entschieden, die den Geschwindigkeitsbegriffen zugrunde liegende sicherungstechnische Logik als Vorlage für eine Implementierung zu nutzen. Damit kann eine Vielzahl von Signalsystemen einheitlich abgebildet werden, was auch dem universellen Charakter von RailML entspricht.

Dem Element <signal> wurde deshalb ein neues Unterelement <signalingBlocks> hinzugefügt, welches die einzelnen Signalisierungsabschnitte beinhaltet.

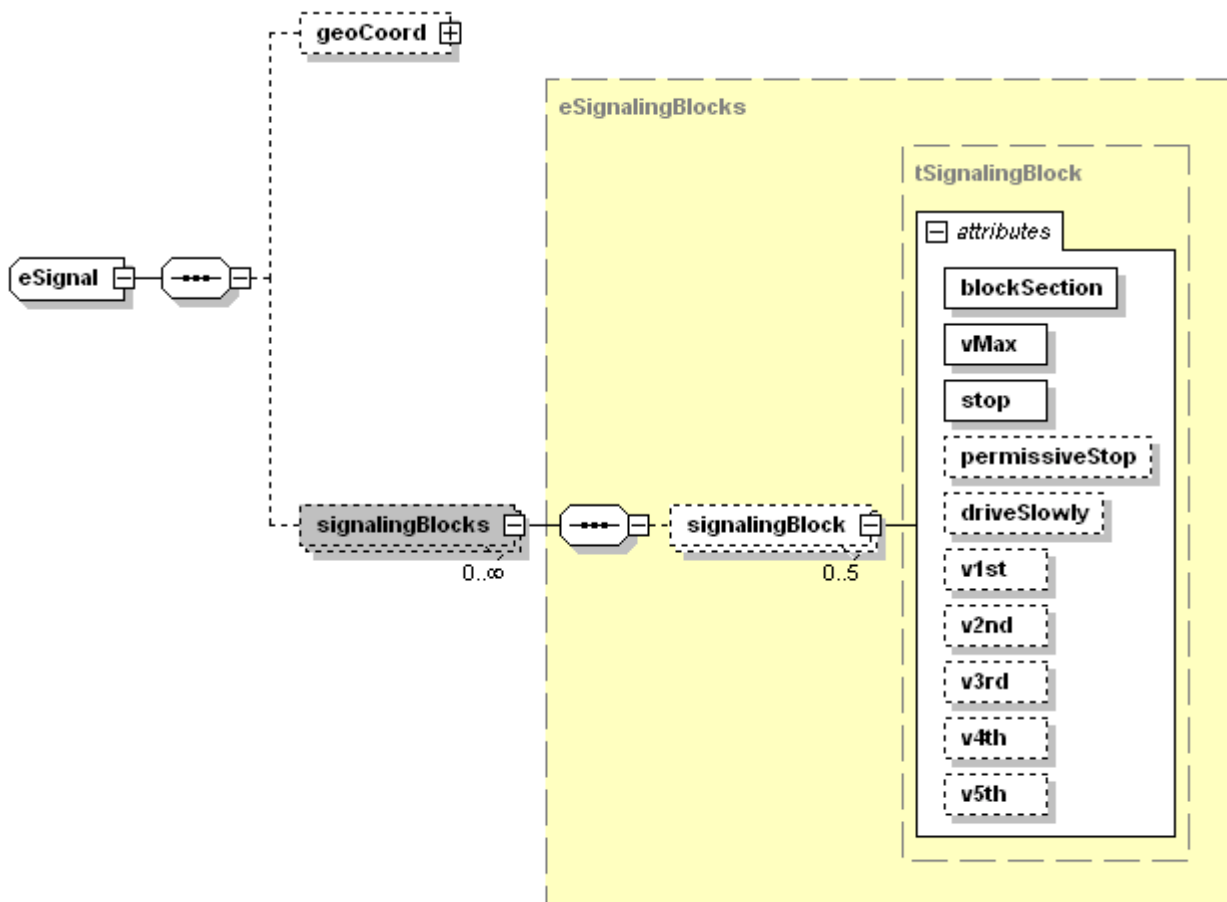
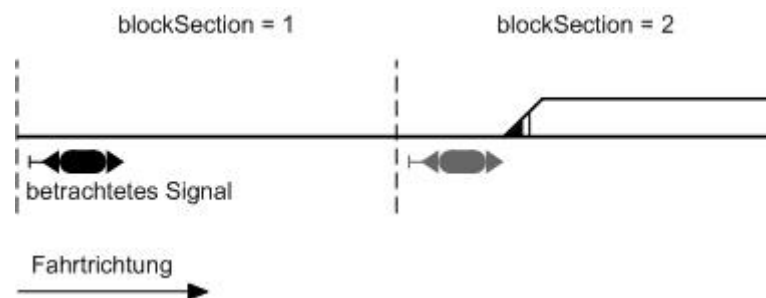


Abbildung 15 Erweiterung um <signalingBlocks>

In den neuen Unterelementen können die Anzeigemöglichkeiten für jeden beeinflussten Blockabschnitt abgebildet werden. Tabelle 4 zeigt das zuvor erläuterte Beispiel mit der vorgeschlagenen Erweiterung.

Da es sich um eine Zweiabschnittssignalisierung handelt, erhält das Element <signalingBlocks> zwei Unterelemente vom Typ <signalingBlock>, vgl. Abbildung 16.



**Abbildung 16: Darstellung der Signalisierungsabschnitte bei Zweiabschnittssignalisierung**

Die Attribute des jeweiligen Unterelementes definieren, welche Informationen das Signal für den entsprechenden Abschnitt geben kann. Wenn wie in Abbildung 16 alternative Fahrwege nach dem 2. Signal existieren, werden die unterschiedlichen Geschwindigkeiten als zugehörig zum 2. Signal aufgelistet, ungeachtet ihrer Zugehörigkeit zu unterschiedlichen Fahrstraßen.

Die Optionen für den unmittelbar anschließenden Blockabschnitt (blockSection = 1) sind:

- Fahrt mit zulässiger Streckengeschwindigkeit (vMax = true),
- Halt (Stopp = true) und
- Fahrt mit 80km/h (v1st = 80 km/h).

Darin spiegeln sich praktisch die Hauptsignaleigenschaften wider.

Die vorankündigenden Informationen für den nächsten Abschnitt (blockSection = 2) werden äquivalent zu den Informationen im ersten Abschnitt abgebildet.


Die Attribute in diesem <signalingBlock> (blockSection = 2) sind als Vorsignaleigenschaften zu interpretieren, im Einzelnen:

- Fahrt erwarten (vMax = true),
- Halt erwarten (Stopp = true),
- erwarte Fahrt mit 60 km/h (v1st = 60) und
- erwarte Fahrt mit 40 km/h (v2nd = 40).

Die Geschwindigkeiten sollten dabei in absteigender Reihenfolge von der größtmöglichen (in „v1st“) hin zur kleinstmöglichen Geschwindigkeit hinterlegt werden.



**Tabelle 4: Signalelement mit neuem Unterelement für die Signalisierungsabschnitte**

<code>&lt;signal&gt;</code>		Werte		Bemerkungen
	Mehrab-	id:	64P1	
	schnittsignal	name:	Ausfahrtsignal P1	
	mit Zs3v für	type:	Combined	Kombinationssignal
	60/40 km/h	function:	Exit	Ausfahrtsignal
	und Zs3 für	sigSystem:	Ks	Ks-System der DB
80 km/h				
<code>&lt;signalingBlocks&gt;</code>		Unterelemente		
		<code>&lt;signalingBlock&gt;</code>	<code>&lt;signalingBlock&gt;</code>	
blockSection:		1	2	
vMax:		True	True	
stop:		True	True	
v1st:		80km/h	60	
v2nd:		{entfällt}	40	

Ein weiteres Beispiel (siehe Tabelle 5) zeigt die Konfiguration von `<signalingBlocks>` für ein Vorsignal, welches vor einem Einfahrtsignal steht und die folgenden zwei Fahrtbegriffe vorankündigen kann:

Ks2:	gelbes Dauerlicht	Halt erwarten
Ks1+Zs3v	grünes Blinklicht mit gelber 6	Erwarte Fahrt mit v = 60 km/h


Ein Vorsignal erlaubt generell die Einfahrt in den folgenden Abschnitt, weshalb im ersten Abschnitt (blockSection = 1) lediglich vMax = true zu setzen ist.

Das Signal kann vorankündigende Informationen für den Abschnitt hinter dem Einfahrtsignal (blockSection = 2) geben (Zs3v als Licht- oder Formsignal).

Das Zs3v Formsignal kündigt in jedem Fall die Geschwindigkeitseinschränkung auf 60 km/h für das nächste Hauptsignal an. Die Signalbegriffe sind somit:

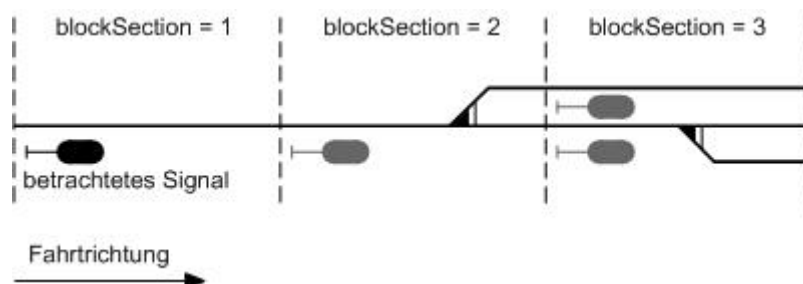
- Erwarte Fahrt mit 60 km/h (v1st = 60) und
- Halt erwarten (stop = true).

**Tabelle 5: Signalelement mit neuem Unterelement für die Signalisierungsabschnitte**

<code>&lt;signal&gt;</code>				Werte	Bemerkungen
	Vorsignal				
	mit Zs3v	id:	62Vb		
	(Formsignal)	name:	Vorsignal Vb		
	für 60 km/h	type:	distant		Vorsignal
		function:	home		für ein Einfahrsignal
	sigSystem:	Ks		Ks-System der DB	
<code>&lt;signalingBlocks&gt;</code>		Unterelemente			
		<code>&lt;signalingBlock&gt;</code>	<code>&lt;signalingBlock&gt;</code>		
	blockSection:	1	2		
	vMax:	true	false		
	stop:	false	true		
	v1st:	{entfällt}	60		

Da das Signal niemals nur „Fahrt erwarten“ ohne eine Geschwindigkeitsvorkündigung anzeigen kann, muss das Attribut vMax = false gesetzt werden.

Bei Signalsystemen, in denen mehr als zwei Abschnitte voraus signalisiert werden, müssen mehr signalingBlocks definiert werden. Am Beispiel in Abbildung 17 wird noch einmal deutlich, dass die eingestellte Fahrstraße für die Nummerierung der blocksection nicht relevant ist, ausschlaggebend ist einzig die Anzahl der passierten Signale.



**Abbildung 17: Darstellung einer Dreiabschnittssignalisierung**

Die vorgeschlagene Art der Abbildung möglicher Geschwindigkeiten wurde vom Signal und dessen Eigenschaften ausgehend entwickelt, d.h. die Eigenschaften des Signals sollen hierbei abgebildet werden. Dies kann zu Redundanzen in der Datenhaltung führen, wenn Signale immer alle Begriffe des darauffolgenden Signals vorsignalisieren können. Praktisch existieren aber auch Fälle, bei denen bestimmte

Höchstgeschwindigkeiten, die sich aus eingestellten Fahrstraßen ergeben, nicht vom Signal angezeigt werden können. Hier muss dann durch das Signal die nächst niedrigere anzeigbare Geschwindigkeit signalisiert werden, was mit der vorgeschlagenen Darstellungsart abgebildet werden kann.

Die vorgestellte Erweiterung beschränkt sich auf die Abbildung der Geschwindigkeitssignalisierung. In einer Weiterentwicklung dieses Ansatzes sollten weitere Elemente definiert werden, die zusätzliche Aspekte wie z. B. Hilfsanzeigen und Richtungssignalisierung berücksichtigen.

### 3.4.2 <trainProtectionElement>

Systeme zur Zugbeeinflussung sorgen für die Vermeidung von Gefahren, die aus der Nichtbeachtung von Signalen bei der Steuerung des Zuges entstehen könnten. Bei der Simulation des Eisenbahnbetriebes ist es daher unerlässlich, die Zustände der Anlagen des Zugsicherungssystems zu berücksichtigen. Da eine solche Möglichkeit in der vorliegenden RailML-Definition nicht existiert, wird eine Erweiterung des Elements <trainProtectionElement> vorgeschlagen. Die Überlegungen für die Erweiterung des Elementes sind dabei stark durch das Zugbeeinflussungssystem PZB geprägt. Deswegen soll auch auf Grundlage dieses Beispiels argumentiert werden. Eine Verallgemeinerung auf andere Systeme erfolgt im Anschluss.

Die PZB dient meist der Überwachung von Signalen. In diesem Anwendungsfall wird ein Gleismagnet immer von genau einem Signal angesteuert. Ein Signal hingegen kann dabei mehrere Gleismagnete ansteuern, im häufigsten Fall einen 2000 Hz-, einen 1000 Hz- und einen 500 Hz-Magnet.

Das Element <trainProtectionElement> wird deshalb, wie in Abbildung 18 dargestellt, um ein Unterelement <controlUnit> erweitert. Dieses Element soll eine Referenz auf das Signal enthalten, welches den Magneten ansteuert. Es ist noch zu prüfen, ob auch den dazugehörigen Signalen eine Liste mit den angesteuerten Elementen der Zugbeeinflussung zugeordnet werden sollte – was allerdings zu Redundanzen in der Datenhaltung führt. Die RailML-Designgrundsätze verbieten dies zwar nicht grundsätzlich, jedoch sollte Datenredundanz prinzipiell vermieden werden.

Bei Einsatz der PZB zur Geschwindigkeitsüberwachung (z. B. Absicherung von Langsamfahrstellen) ist der Aktivierungszustand der Magnete nicht von einem Signal abhängig. Es ist noch offen, inwiefern diese Anwendung der PZB überhaupt im Sinne eines <trainProtectionElement> interpretiert werden kann.

Bei anderen Zugbeeinflussungssystemen erfolgt die Ansteuerung der aktiven streckenseitigen Elemente ebenfalls durch jeweils eine Steuereinheit, hier kann analog vorgegangen werden (z. B. signalabhängige ETCS-Balisen Typ 3 in Level 1, Gleisstromkreise bei ATB-EG). Bei kontinuierlich wirkenden Systemen, z. B. dem Zugsicherungselement Linienleiter im System LZB, könnte als <controlUnit> eine Referenz zur LZB-Blockzentrale angegeben werden.

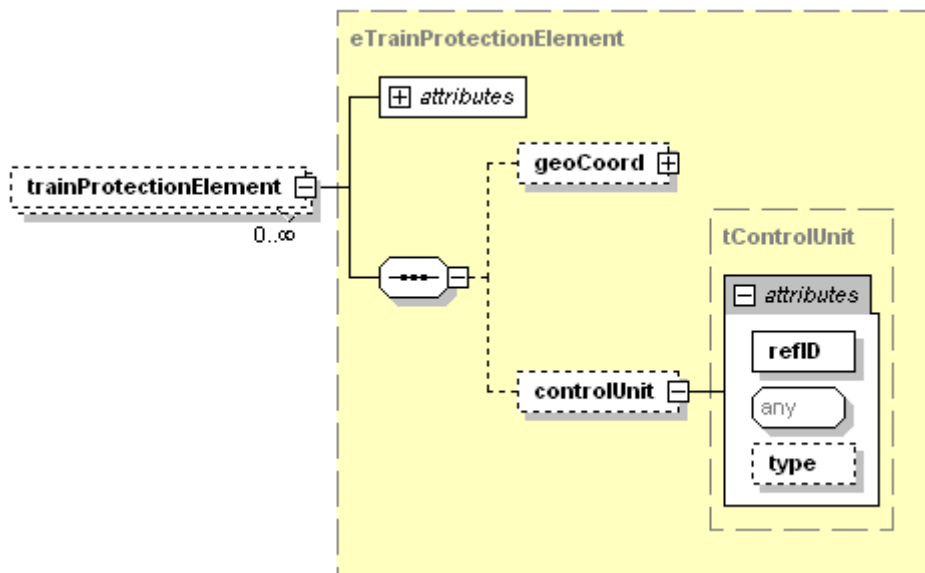


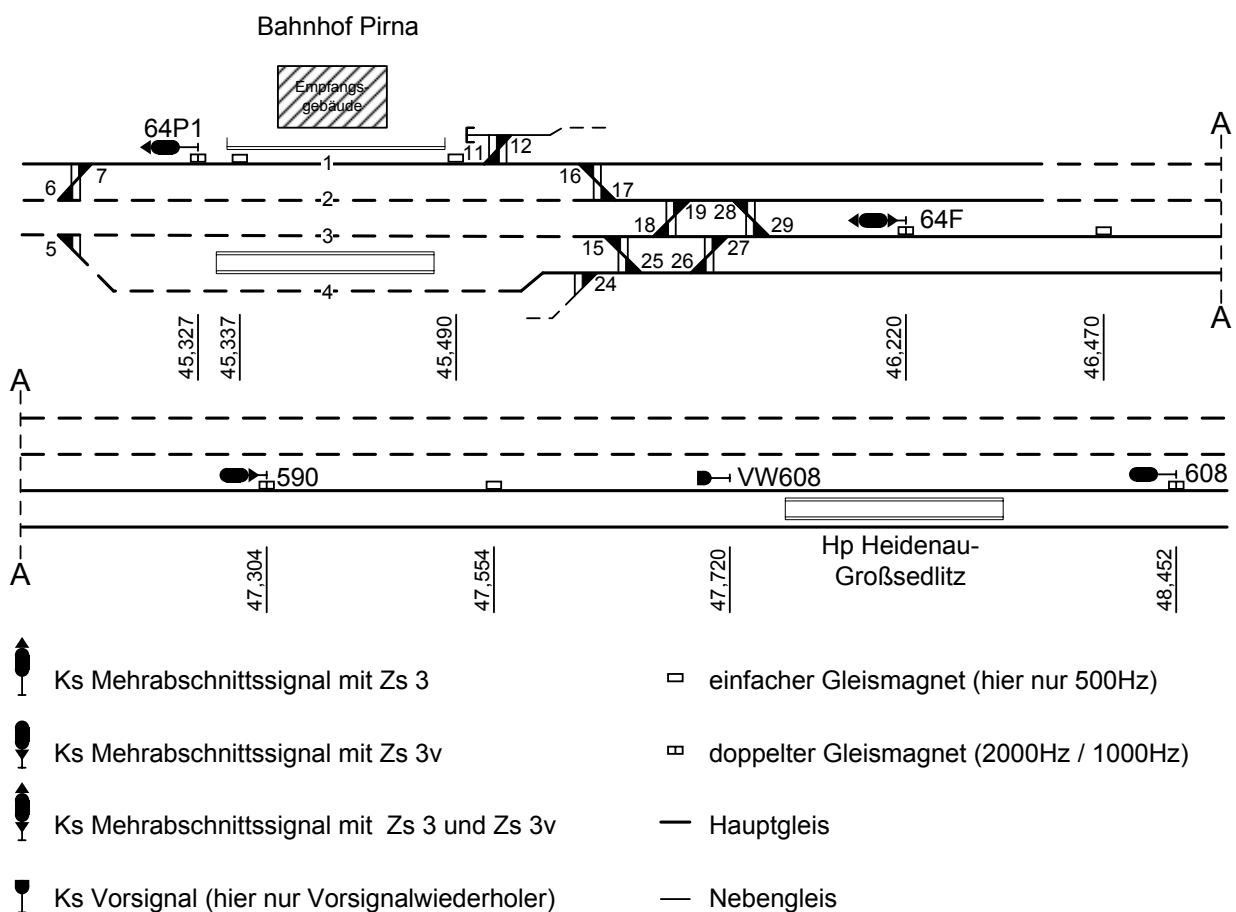
Abbildung 18 Erweiterung um <controlUnit>

## 4 Praxisbeispiel

### 4.1 Festlegungen und schematischer Plan

Anhand des folgenden Praxisbeispiels der S-Bahn Dresden soll die Modellierung der Sicherungstechnik in RailML veranschaulicht werden. Es werden nicht die gesamten sicherungstechnischen Beziehungen mit allen Fahrstraßen und Infrastrukturelementen abgebildet. Um das Beispiel übersichtlich zu halten, beschränkt es sich auf die Einfahrt der S-Bahn S1 von Heidenau-Großsedlitz kommend in Pirna an den Hausbahnsteig (Gleis 1).

Die Abbildung 19 zeigt schematisch die Darstellung der Infrastruktur vom Haltepunkt Heidenau-Großsedlitz bis zum Bahnhof Pirna. Für die Einfahrt irrelevante Elemente wurden ausgeblendet bzw. gestrichelt dargestellt. Der komplette XML-Code – für die betrachtete, eingeschränkte Infrastruktur – ist im Anhang B zu finden. In diesem Kapitel werden wesentliche Auszüge der RailML-Daten erläutert.



**Abbildung 19: Bsp. Einfahrt in Pirna**

## 4.2 Daten im Bereich <interlocking>

### 4.2.1 Einordnung des Stellwerkes

Der Bereich <interlocking> befindet sich im Teilschema <infrastructure> (im Codebeispiel ausgeblendet) und nimmt die LST-Daten auf.

Zunächst wird ein Element <mainSignalBox> hinzugefügt, welches für dieses Beispiel die ESTW-Zentrale aufnimmt. Es handelt sich um das elektronische Stellwerk Pirna (type = microelectronic, name = Stellwerk Pirna) mit der id = ESTW-Z\_DPI. Diesem wird ein Element <subSignalBox> untergeordnet, welches ein ESTW-A abbildet. Hier wurde nur ein ESTW-A definiert, welches alle relevanten sicherungstechnischen Tabellen aufnimmt. Die Attribute sind äquivalent zum übergeordneten Element.

```
<interlocking>  
  <mainSignalBox type="microelectronic" id="ESTW-Z_DPI" name="Stellwerk Pirna">  
    <subSignalBox type="microelectronic" id="ESTW-A_DPI_64" name="ESTW-A BF Pirna">
```

### 4.2.2 Fahrstraßentabelle

Eine Fahrstraße wird im Element <route> beschrieben und dem Wurzelement <routes> (Fahrstraßentabelle) zugeordnet.

Das Element erhält eine eindeutige ID (id = 64\_F/P1) und im Attribut vMax kann die maximal zulässige Geschwindigkeit für die Fahrstraße hinterlegt werden, die sich aus den Kurvenradien der Fahrweegelemente ergibt.

Es ist zu beachten, dass die hier hinterlegte Geschwindigkeit nicht zwangsläufig maßgebend für die Fahrstraße sein muss, da auch die Durchrutschwege die zulässige Geschwindigkeit beeinflussen (vgl. Abschnitt 3.3.4).

Der Fahrstraßenanfang wird im Element <routeStart> als Referenz auf ein Objekt hinterlegt, hier das Einfahrsignal „64F“. Analog wird für das Fahrstraßenziel im Element <routeDestination> das Ausfahrtsignal „64P1“ referenziert.

```
<routes>  
  <route vMax="60" id="64_F/P1">  
    <routeStart refID="64F"></routeStart>  
    <routeDestination refID="64P1"></routeDestination>
```

In `<elements>` befinden sich alle beweglichen Fahrwegelemente. Für das Beispiel sind es sechs Weichen `<switch>`. Die Attribute Weichenstellung (`switchStand`) und Referenz auf das eigentliche Objekt (`refID`) müssen gepflegt werden (Weichenstellung: für die Fahrstraße benötigt Lage).

```

<elements>
  <element><switch switchStand="right" refID="64W29"/></element>
  <element><switch switchStand="right" refID="64W28"/></element>
  <element><switch switchStand="right" refID="64W19"/></element>
  <element><switch switchStand="right" refID="64W17"/></element>
  <element><switch switchStand="right" refID="64W16"/></element>
  <element><switch switchStand="right" refID="64W11"/></element>
</elements>

```

Die Flankenschutzelemente befinden sich im Ordner `<flankProtElements>` und werden in gleicher Weise wie die beweglichen Fahrwegelemente beschrieben. Auf die Abbildung der Merkmale von Zwieschutzweichen (in diesem Bsp. die Weiche 18) wurde vorerst verzichtet.

```

<flankProtElements>
  <FlankProtElement><switch switchStand="right" refID="64W26"/></FlankProtElement>
  <FlankProtElement><switch switchStand="right" refID="64W15"/></FlankProtElement>
  <FlankProtElement><switch switchStand="left" refID="64W18"/></FlankProtElement>
  <FlankProtElement><switch switchStand="right" refID="64W12"/></FlankProtElement>
</flankProtElements>

```

Im Element `<overlaps>` muss mindestens ein Durchrutschweg referenziert sein. Es können aber auch mehrere Referenzen angegeben werden, wenn Wahldurchrutschwege abgebildet werden sollen. Die Durchrutschwege befinden sich als Objekte in der Durchrutschwegtabelle (siehe 4.2.3).

```

<overlaps>
  <refOverlap refID="D1_64P1"/>
  <refOverlap refID="D2_64P1"/>
</overlaps>
</route>

```

Das Element `<trackSections>` fehlt, da für die Gleisfreimeldeabschnitte keine Daten vorlagen.

Für das gewählte Beispiel kann die Beschreibung der Fahrstraße F/P1 damit als abgeschlossen betrachtet werden.

### 4.2.3 Durchrutschwegtabelle

Das Element `<overlaps>` nimmt die Durchrutschwegtabelle auf. Die einzelnen Durchrutschwege können mit Hilfe der Attribute des Elementes `<overlap>` beschrieben werden.

Am Beispiel des ersten Durchrutschweges werden die Werte der Attribute kurz erläutert. Das Objekt bekommt eine eindeutige ID (`id = D1_64P1`). Der Durchrutschweg liegt hinter dem Ausfahrtsignal P1 (`fromSignal = 64P1`), lässt eine Einfahrgeschwindigkeit von 80 km/h zu (`vPossible = 80`), endet am Grenzzeichen der Weiche W1 (`until = 64W1`) und seine Soll-Länge beträgt 200 m (`targetLength = 200`). Optional können zudem die Ist-Länge (`actualLength`) und die maßgebende Neigung (`slope`) angegeben werden, worauf hier allerdings verzichtet wurde.

```
<overlaps>
  <overlap fromSignal="64P1" id="D1_64P1" vPossible="80" until="64W1" targetLength="200"></overlap>
  <overlap fromSignal="64P1" id="D2_64P1" vPossible="60" until="64W7" targetLength="100"></overlap>
  <overlap fromSignal="64F" id="D1_64F" vPossible="160" until="64W29" targetLength="200"></overlap>
  <overlap fromSignal="590" id="D1_590" vPossible="160" until="xxx" targetLength="200"></overlap>
</overlaps>
```

## 4.3 Daten im Bereich <lines>

### 4.3.1 Signalliste

Die Attribute des Elementes `<signal>` haben sich nicht verändert und werden deswegen nicht diskutiert. Das neu eingeführte Element `<signalingBlocks>` wird jedem Signal untergeordnet mit Ausnahme des Vorsignalwiederholers (`id = VW608`). Es enthält immer zwei Unterelemente `<signalingBlock>`, da es sich um eine Zweiabschnittssignalisierung handelt. Die Verwendung der Attribute des Elementes `<signalingBlock>` wurde in Abschnitt 3.4.1 ausführlich erläutert. Für die Einfahrt (Fahrstraße: F/P1) sind das „Blocksignal590“ und das „Einfahrtsignal F“ von Bedeutung. Die Einfahrgeschwindigkeiten von 60 km/h wird vom „Einfahrtsignal F“ angezeigt (`blockSection = 1` mit `v2nd = 60`) und wird vom „Blocksignal590“ (`blockSection = 2` mit `v2nd = 60`) vorsignalisiert.

```
<signals>
  <signal id="64P1" name="Ausfahrtsignal P1" pos="45.327" dir="up" type="main" function="exit">
    <signalingBlocks>
      <signalingBlock vMax="true" stop="true" blockSection="1" v1st="80" />
      <signalingBlock vMax="true" stop="true" blockSection="2"/>
    </signalingBlocks>
  </signal>
```



```

<signal id="64F" name="Einfahrsignal F" pos="46.220" dir="up" type="main" function="home">
  <signalingBlocks>
    <signalingBlock vMax="true" stop="true" blockSection="1" v1st="80" v2nd="60"/>
    <signalingBlock vMax="true" stop="true" blockSection="2" v1st="80" v2nd="60"/>
  </signalingBlocks>
</signal>
<signal id="590" name="Blocksignal590" pos="47.304" dir="up" type="main" function="blocking">
  <signalingBlocks>
    <signalingBlock vMax="true" stop="true" blockSection="1"/>
    <signalingBlock vMax="true" stop="true" blockSection="2" v1st="80" v2nd="60"/>
  </signalingBlocks>
</signal>
<signal id="VW608" name="Vorsignalwiederholer608" pos="47.720" dir="up" type="repeater"/>
<signal id="608" name="Blocksignal608" pos="48.452" dir="up" type="main" function="blocking">
  <signalingBlocks>
    <signalingBlock vMax="true" stop="true" blockSection="1"/>
    <signalingBlock vMax="true" stop="true" blockSection="2"/>
  </signalingBlocks>
</signal>
</signals>

```

#### 4.3.2 Zugbeeinflussung

Die Infrastruktur im Beispiel ist mit der punktförmigen Zugbeeinflussung (system = PZB) ausgestattet. Die hier dargestellten PZB-Magnete werden alle vom „Ausfahrtsignal P1“ angesteuert. Aus diesem Grund erhält jedes Element `<trainProtectionElement>` ein Unterelement `<controlUnit>`, welches die Referenz auf dieses Signal (refID = 64P1) herstellt.

```

<trainProtectionElements>
  <trainProtectionElement id="64P1_1_01" name="Gleismagnet1000Hz" pos="45.327" dir="up"
    medium="ind" system="PZB" model="1000Hz">
    <controlUnit refID="64P1"></controlUnit>
  </trainProtectionElement>
  <trainProtectionElement id="64P1_2_02" name="Gleismagnet2000Hz" pos="45.327" dir="up"
    medium="ind" system="PZB" model="2000Hz">
    <controlUnit refID="64P1"></controlUnit>
  </trainProtectionElement>
  <trainProtectionElement id="64P1_0.5_03" name="Gleismagnet500Hz" pos="45.337" dir="up"
    medium="ind" system="PZB" model="500Hz">
    <controlUnit refID="64P1"></controlUnit>
  </trainProtectionElement>
  <trainProtectionElement id="64P1_0.5_04" name="Gleismagnet500Hz" pos="45.490" dir="up"
    medium="ind" system="PZB" model="500Hz">
    <controlUnit refID="64P1"></controlUnit>
  </trainProtectionElement>

```

## 5 Zusammenfassung und Ausblick

RailML ist mittlerweile ein anerkanntes Datenformat in der Bahninformatik. Insbesondere Simulationsanwendungen nutzen das Format. Eine Vielzahl von Sachverhalten lässt sich schon heute gut mit den RailML-Teilschemata abbilden.

Es gibt allerdings noch Sachverhalte, die nur schlecht oder gar nicht im aktuellen Schema eingebunden sind.

Die Ergänzung der Leit- und Sicherungstechnik in RailML wird gegenwärtig als dringlichste Aufgabe angesehen [4]. Mit der Entwicklung eines `<interlocking>` Schemas kann diese gelöst werden. Die Arbeit ist der Versuch - aufbauend auf dem bestehenden Infrastrukturschema und den Vorarbeiten des RailML Entwicklerkonsortiums - einen Entwicklungsvorschlag für den Bereich `<interlocking>` zu unterbreiten.

Es war nicht möglich, im Rahmen dieser Arbeit ein komplettes Schema zur LST zu erstellen. Der detaillierte Vorschlag zur Darstellung von Fahrstraßen sowie der von Signalen dargestellten Geschwindigkeiten soll jedoch einen Beitrag zur gegenwärtigen Diskussion über die Gestaltung eines LST-Schemas darstellen.

Während der Ausarbeitung kamen grundlegende Gestaltungsfragen auf, die während der Bearbeitung vorerst zurückgestellt wurden. Sie sollten jedoch in einer Diskussion zum `<interlocking>` Schema zur Sprache kommen. So ist zu prüfen, ob eine generelle Zuordnung der Stellelemente zu den Stellwerken oder Betriebsstellen nicht sinnvoller wäre, anstatt sie über `<ocsElements>` einem Gleis zuzuordnen. Beispielsweise könnten Signale direkt als Signaltabelle in einem Stellwerk geführt werden, eine Zuordnung zu den Gleisen könnte über eine `refID` und die Position erfolgen. Diese Art der Zuordnung wäre auch für andere Elemente der LST denkbar (z. B. Achszähler und Gleisstromkreise).

Die nächste Konferenz der RailML-Initiative bietet die Möglichkeit, diese Fragen zu erörtern und die Entwicklung einer neuen Version voranzutreiben, die ein `<interlocking>` Teilschema beinhaltet.

Für die Implementierung betrieblicher Sachverhalte in die Fahrsimulation am Lehrstuhl Verkehrsleitsysteme und –prozessautomatisierung sind nutzbare Datenformate entstanden. Es müssen nun Module geschaffen werden, die mit diesen Daten den Betrieb simulieren und über zu definierende Schnittstellen dem Fahrsimulator die zur Anzeige benötigten Daten liefern.

## Quellenverzeichnis

- [1] **Fenner, W, Naumann, P.:**  
Verkehrssicherungstechnik; Publicis MCD Verlag; 1998
- [2] **Fries, N.:**  
Modellierung einer Eisenbahninfrastruktur in RailML; Studienarbeit;  
TU-Dresden; 2003
- [3] **Institution of Railway Signal Engineers:**  
European Railway Signalling; A & C Black; London; 1995
- [4] **Maschek, U.:**  
1. Kolloquium „Durchgängige Datenhaltung für ESTW“;  
Signal + Draht (96), 1+2/2004
- [5] **Maschek, U.:**  
Datenmodell zur Planung von Stellwerken; Dissertation; 2001
- [6] **Naumann, P., Pacht J.:**  
Leit- und Sicherungstechnik im Bahnbetrieb; Tetzlaff Verlag GmbH & Co. KG;  
2002
- [7] **RailML.org:**  
Versionsübersicht; <http://www.RailML.org/de/development/overview.html>;  
[Zugriff:22.06.08]
- [8] **RailML.org:**  
Bildungsvorschriften für RailML-TeilSchemata;  
[http://www.RailML.org/schemas/versions/RailML\\_v1-1.zip](http://www.RailML.org/schemas/versions/RailML_v1-1.zip); [Zugriff: 20.06 08]
- [9] **RailML.org:**  
Interlocking Intended Features;  
[http://wiki.RailML.org/index.php?title=IL\\_IntendedFeatures](http://wiki.RailML.org/index.php?title=IL_IntendedFeatures); [Zugriff: 06.08.08]
- [10] **World Wide Web Consortium (W3C):**  
W3C XML Schema; <http://www.w3.org/XML/Schema>; [Zugriff: 20.06.08]

## Anhang A – Neue und veränderte Elemente in RailML

Quelldatei: RailML\_erweiterung.xsd:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2008 rel. 2 (http://www.altova.com) by Martin Lehmann -->
<xsd:schema xmlns="http://www.RailML.org/schemas/2007"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.RailML.org/schemas/2007" elementFormDefault="qualified"
version="1.2β">
  <xsd:include schemaLocation="infrastructure.xsd"/>
  <!-->
  <!-- RailML -->
  <!-- +- infrastructure -->
  <!-->
  <xsd:element name="RailML">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="infrastructure" type="eInfrastructure" minOccurs="0"/>
        <xsd:element name="infrastructureVisualizations"
type="eInfrastructureVisualizations" minOccurs="0"/>
        <xsd:element name="rollingstock" minOccurs="0"/>
        <xsd:element name="timetable" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="version" type="xsd:decimal" use="optional"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Quelldatei: RailML\_erweiterung.xsd:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2008 rel. 2 (http://www.altova.com) by Mr. Lee (none) -->
<xsd:schema xmlns="http://www.RailML.org/schemas/2007"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.RailML.org/schemas/2007"
elementFormDefault="qualified" version="1.2β">
  <xsd:include schemaLocation="infrastructureTypes.xsd"/>
  <!-->
  <!-- infrastructure -->
  <!-- +- operationControlPoints -->
  <!-- +- lines -->
  <!-- +- interlocking -->
  <!-->
  <xsd:complexType name="eInfrastructure">
    <xsd:complexContent>
      <xsd:extension base="tInfrastructure">
        <xsd:sequence>
          <xsd:element name="lines" type="eLines" minOccurs="0"/>
          <xsd:element name="operationControlPoints"
type="eOperationControlPoints" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:schema>
```

```

        <xsd:element name="interlocking" type="eInterlocking" minOccurs="0"/>
    </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-->
<!-- interlocking -->
<!-- +- signalBox -->
<!-->
<xsd:complexType name="eInterlocking">
    <xsd:sequence>
        <xsd:element name="mainSignalBox" type="eMainSignalBox"/>
    </xsd:sequence>
</xsd:complexType>
<!-->
<!-- MainSignalBox -->
<!-- +- routes -->
<!-- +- subSignalBox -->
<!-->
<xsd:complexType name="eMainSignalBox">
    <xsd:complexContent>
        <xsd:extension base="tSignalBox">
            <xsd:sequence>
                <xsd:element name="ownControlRange" type="eOwnControlRange"
minOccurs="0"/>
                <xsd:element name="subSignalBox" type="eSubSignalBox" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
            <xsd:attribute name="operationControlCenter" type="tGenericID"
use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<!-->
<!-- ownControlRange -->
<!-- +- routes -->
<!-- +- localOpArea -->
<!-->
<xsd:complexType name="eOwnControlRange">
    <xsd:sequence>
        <xsd:element name="routes" type="eRoutes"/>
        <xsd:element name="trackSections" type="eTrackSections"/>
        <xsd:element name="overlaps" type="eOverlaps"/>
        <xsd:element name="localOpAreas" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<!-->
<!-- subSignalBox -->
<!-- +- routes -->
<!-- +- localOpArea -->
<!-->

```

```

<xsd:complexType name="eSubSignalBox">
  <xsd:complexContent>
    <xsd:extension base="tSignalBox">
      <xsd:sequence>
        <xsd:element name="routes" type="eRoutes" minOccurs="0"/>
        <xsd:element name="trackSections" type="eTrackSections" minOccurs="0"/>
        <xsd:element name="overlaps" type="eOverlaps" minOccurs="0"/>
        <xsd:element name="localOpAreas" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-->
<!-- routes -->
<!-- +- route-->
<!-->
<xsd:complexType name="eRoutes">
  <xsd:sequence>

    <xsd:element name="route" type="eRoute" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<!-->
<!-- route -->
<!-- +- startSignal -->
<!-- +- destination -->
<!-- +- elements -->
<!-- +- trackSections -->
<!-- +- overlaps -->
<!-->
<xsd:complexType name="eRoute">
  <xsd:complexContent>
    <xsd:extension base="tRoute">
      <xsd:sequence>
        <xsd:element name="routeStart" type="tRouteElement"/>
        <xsd:element name="routeDestination" type="tRouteElement"/>
        <xsd:element name="elements" type="eElements" minOccurs="0"/>
        <xsd:element name="flankProtElements" type="eFlankProtElements"
minOccurs="0"/>
        <xsd:element name="trackSections" type="eRefTrackSections"
minOccurs="0"/>
        <xsd:element name="overlaps" type="eRefOverlaps" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-->
<!-- routeElements -->
<!-- +- Element -->
<!-->
<xsd:complexType name="eElements">

```

```

    <xsd:sequence>
      <xsd:element name="element" type="eElement" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
<!-->
<!-- Element -->
<!-- +- switch -->
<!-- +- signal -->
<!-- +- crossing -->
<!-- +- levelCrossing -->
<!-- +- derailer -->
<!-->
<xsd:complexType name="eElement">
  <xsd:choice>
    <xsd:element name="switch" type="tSwitchInRoute"/>
    <xsd:element name="signal" type="tSignalInRoute"/>
    <xsd:element name="crossing" type="tElementWithReference"/>
    <xsd:element name="levelCrossing" type="tElementWithReference"/>
    <xsd:element name="derailer" type="tElementWithReference"/>
    <!-- Provide an extension point for other attributes-->
    <xsd:any namespace="##other" processContents="strict" minOccurs="0"/>
  </xsd:choice>
</xsd:complexType>
<!-->
<!-- flankProtElements -->
<!-- +- flankProtElement -->
<!-->
<xsd:complexType name="eFlankProtElements">
  <xsd:sequence>
    <xsd:element name="FlankProtElement" type="eFlankProtElement"
minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<!-->
<!-- flankProtElement -->
<!-- +- switch -->
<!-- +- redSignal -->
<!-- +- derailer -->
<!-->
<xsd:complexType name="eFlankProtElement">
  <xsd:choice>
    <xsd:element name="switch" type="tSwitchInRoute"/>
    <xsd:element name="redSignal" type="tElementWithReference"/>
    <xsd:element name="derailer" type="tElementWithReference"/>
    <!-- Provide an extension point for other attributes-->
    <xsd:any namespace="##other" processContents="strict" minOccurs="0"/>
  </xsd:choice>
</xsd:complexType>
<!-->
<!-- Element -->

```

```

<!-- +- eRefTrackSections -->
<!-->
<xsd:complexType name="eRefTrackSections">
  <xsd:sequence>
    <xsd:element name="trackSection" type="tElementWithReference" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<!-->
<!-- Element -->
<!-- +- eRefOverlaps -->
<!-->
<xsd:complexType name="eRefOverlaps">
  <xsd:sequence>
    <xsd:element name="refOverlap" type="tElementWithReference" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<!-->
<!-- Element -->
<!-- +- eTrackSection -->
<!-->
<xsd:complexType name="eTrackSections">
  <xsd:sequence>
    <xsd:element name="trackSection" type="tTrackSection" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<!-->
<!-- Element -->
<!-- +- eOverlaps -->
<!-->
<xsd:complexType name="eOverlaps">
  <xsd:sequence>
    <xsd:element name="overlap" type="eOverlap" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<!-->
<!-- overlap -->
<!-->
<xsd:complexType name="eOverlap">
  <xsd:complexContent>
    <xsd:extension base="tOverlap">
      <xsd:sequence>
        <xsd:element name="elements" type="eElements" minOccurs="0"/>
        <xsd:element name="flankProtElements" type="eFlankProtElements"
minOccurs="0"/>
        <xsd:element name="trackSections" type="eRefTrackSections"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```



```

    </xsd:extension>20
  </xsd:complexContent>
</xsd:complexType>

```

<!--Ausgeblendete Elemente wurden nicht verändert -->

```
</xsd:schema>
```

Quelldatei: RailML\_erweiterung.xsd:

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.RailML.org/schemas/2007"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.RailML.org/schemas/2007"
  elementFormDefault="qualified" version="1.2β">
  <xsd:include schemaLocation="railwayUnits.xsd"/>
  <xsd:include schemaLocation="genericRailML.xsd"/>

```

<!--Ausgeblendete Elemente wurden nicht verändert -->

```

<!--*****-->
<!-- SignalBox -->
<!--*****-->
<xsd:simpleType name="tSignalBoxType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="mechanic"/>
    <xsd:enumeration value="electro-mechanic"/>
    <xsd:enumeration value="relais"/>
    <xsd:enumeration value="microelectronic"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="tSignalBox">
  <xsd:complexContent>
    <xsd:extension base="tElementWithIDAndName">
      <xsd:attribute name="type" type="tSignalBoxType" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!--*****-->
<!-- Route -->
<!--*****-->
<xsd:complexType name="tRoute">
  <xsd:attribute name="id" type="tGenericID" use="required"/>
  <xsd:attribute name="vMax" type="tSpeedKmPerHour" use="required"/>
  <xsd:attribute name="setupTime" type="xsd:double" use="optional"/>
  <xsd:attribute name="releaseTime" type="xsd:double" use="optional"/>
  <xsd:attribute name="Length" type="xsd:double" use="optional"/>
  <xsd:attribute name="type" type="tRouteFunction" use="optional"/>
</xsd:complexType>
<!--*****-->
<!-- RouteElement -->
<!--*****-->
<xsd:simpleType name="tRouteElementType">

```

```

    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="switch"/>
      <xsd:enumeration value="bufferStop"/>
      <xsd:enumeration value="signal"/>
      <xsd:enumeration value="sign"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="tRouteFunction">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="normal"/>
      <xsd:enumeration value="blocking"/>
      <xsd:enumeration value="shunting"/>
      <xsd:enumeration value="other"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:complexType name="tRouteElement">
    <xsd:complexContent>
      <xsd:extension base="tElementWithReference">
        <xsd:attribute name="type" type="tRouteElementType" use="optional"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="tSwitchInRoute">
    <xsd:complexContent>
      <xsd:extension base="tElementWithReference">
        <xsd:attribute name="switchStand" type="tCourse" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="tSignalInRoute">
    <xsd:complexContent>
      <xsd:extension base="tElementWithReference">
        <xsd:attribute name="SignalAspect" type="xsd:string" use="required"/>
        <xsd:attribute name="type" type="tSignalType" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="tTrackSection">
    <xsd:attribute name="id" type="tGenericID" use="required"/>
    <xsd:attribute name="kind" type="xsd:string" use="optional"/>
  </xsd:complexType>
  <xsd:complexType name="tOverlap">
    <xsd:attribute name="id" type="tGenericID" use="required"/>
    <xsd:attribute name="fromSignal" type="tGenericID" use="required"/>
    <xsd:attribute name="until" type="tGenericID" use="required"/>
    <xsd:attribute name="vPossible" type="tSpeedKmPerHour" use="required"/>
    <xsd:attribute name="targetLength" type="tLengthM" use="required"/>
    <xsd:attribute name="actualLength" type="tLengthM" use="optional"/>
    <xsd:attribute name="slope" type="tGradientPromille" use="optional"/>
  </xsd:complexType>
</xsd:schema>

```

## Anhang B – RailML-Code für das Beispiel in Kapitel 4

```

<?xml version="1.0" encoding="UTF-8"?>
<RailML xmlns:cust="www.RailML.org/schema/customer"
xmlns="http://www.RailML.org/schemas/2007"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.RailML.org/schemas/2007
RailML_erweiterung.xsd">
<infrastructure version="1.2β ">
  <lines>
    <line name="Hauptbahn DHDG-DPI" id="DHDG-DPI">
      <infraAttrGroupID>1</infraAttrGroupID>
      <tracks>
        <track name="Gleis1" id="64T1" type="mainTrack">
          <trackTopology>
            <trackBegin>
              <simpleConnection id="64W7" pos="45.200" name="Weiche2">
</simpleConnection>
              </trackBegin>
              <trackEnd>
                <simpleConnection id="64W16" pos="45.600" name="Weiche16">
</simpleConnection>
              </trackEnd>
              <connections>
                <switch id="64W11" pos="45.550" name="Weiche11"/>
              </connections>
            </trackTopology>
            <ocsElements>
              <signals>
                <signal id="64P1" name="Ausfahrtsignal P1" pos="45.327" dir="up" type="main"
function="exit">
                  <signalingBlocks>
                    <signalingBlock vMax="true" stop="true" blockSection="1" v1st="80" />
                    <signalingBlock vMax="true" stop="true" blockSection="2"/>
                  </signalingBlocks>
                </signal>
              </signals>
              <trainProtectionElements>
                <trainProtectionElement id="64P1_1_01" name="Gleismagnet1000Hz"
pos="45.327" dir="up" medium="ind" system="PZB" model="1000Hz">
                  <controlUnit refID="64 P1"></controlUnit>
                </trainProtectionElement>
                <trainProtectionElement id="64P1_2_02" name="Gleismagnet2000Hz"
pos="45.327" dir="up" medium="ind" system="PZB" model="2000Hz">
                  <controlUnit refID="64 P1"></controlUnit>
                </trainProtectionElement>
                <trainProtectionElement id="64P1_0.5_03" name="Gleismagnet500Hz"
pos="45.337" dir="up" medium="ind" system="PZB" model="500Hz">
                  <controlUnit refID="64P1"></controlUnit>
                </trainProtectionElement>
              </trainProtectionElements>
            </ocsElements>
          </track>
        </tracks>
      </line>
    </lines>
  </infrastructure>

```

```

        </trainProtectionElement>
        <trainProtectionElement id="64P1_0.5_04" name="Gleismagnet500Hz"
pos="45.490" dir="up" medium="ind" system="PZB" model="500Hz">
            <controlUnit refID="64P1"></controlUnit>
        </trainProtectionElement>
    </trainProtectionElements>
</ocsElements>
</track>
<track name="Gleis2" id="64T2" type="mainTrack">
    <trackTopology>
        <trackBegin>
            <simpleConnection id="64W6" pos="45.150" name="Weiche6">
</simpleConnection>
        </trackBegin>
        <trackEnd>
            <simpleConnection id="64W28" pos="45.780" name="Weiche28">
</simpleConnection>
        </trackEnd>
        <connections>
            <switch id="64W17" pos="45.650" name="Weiche17"/>
            <switch id="64W19" pos="45.700" name="Weiche19"/>
        </connections>
    </trackTopology>
</track>
<track name="Gleis3" id="64T3" type="mainTrack">
    <trackTopology>
        <trackBegin>
            <simpleConnection id="64W5" pos="45.190" name="Weiche5">
</simpleConnection>
        </trackBegin>
        <trackEnd>
            <simpleConnection id="64W29" pos="45.820" name="Weiche29">
</simpleConnection>
        </trackEnd>
        <connections>
            <switch id="64W15" pos="45.550" name="Weiche15"/>
            <switch id="64W18" pos="45.650" name="Weiche18"/>
            <switch id="64W27" pos="45.780" name="Weiche27"/>
        </connections>
    </trackTopology>
</track>
<track name="Gleis4" id="64T4" type="mainTrack">
    <trackTopology>
        <trackBegin>
            <simpleConnection id="64W5" pos="45.190" name="Weiche5">
</simpleConnection>
        </trackBegin>
        <trackEnd>
            <simpleConnection id="64W26" pos="45.700" name="Weiche26">
</simpleConnection>
        </trackEnd>
    </trackTopology>
</track>

```

```

    <connections>
      <switch id="64W24" pos="45.600" name="Weiche24"/>
      <switch id="64W25" pos="45.630" name="Weiche25"/>
    </connections>
  </trackTopology>
</track>
<track name="SteckengleisS1r" id="S1r" type="mainTrack">
  <trackTopology>
    <trackBegin>
      <simpleConnection id="64W29" pos="45.820" name="Weiche29">
</simpleConnection>
    </trackBegin>
    <trackEnd>
      <bufferStop id="SkizzenGrenzen1" pos="48.300" name="SkizzenGrenzen"/>
    </trackEnd>
  </trackTopology>
  <ocsElements>
    <signals>
      <signal id="64F" name="Einfahrtsignal F" pos="46.220" dir="up" type="main"
function="home">
        <signalingBlocks>
          <signalingBlock vMax="true" stop="true" blockSection="1" v1st="80"
v2nd="60"/>
          <signalingBlock vMax="true" stop="true" blockSection="2" v1st="80"
v2nd="60"/>
        </signalingBlocks>
      </signal>
      <signal id="590" name="Blocksignal590" pos="47.304" dir="up"
type="main" function="blocking">
        <signalingBlocks>
          <signalingBlock vMax="true" stop="true" blockSection="1"/>
          <signalingBlock vMax="true" stop="true" blockSection="2" v1st="80"
v2nd="60"/>
        </signalingBlocks>
      </signal>
      <signal id="VW608" name="Vorsignalwiederholer608" pos="47.720"
dir="up" type="repeater"/>
      <signal id="608" name="Blocksignal608" pos="48.452" dir="up"
type="main" function="blocking">
        <signalingBlocks>
          <signalingBlock vMax="true" stop="true" blockSection="1"/>
          <signalingBlock vMax="true" stop="true" blockSection="2"/>
        </signalingBlocks>
      </signal>
    </signals>
    <trainProtectionElements>
      <trainProtectionElement id="64F_1_01" name="Gleismagnet1000Hz"
pos="46.220" dir="up" medium="ind" system="PZB" model="1000Hz">
        <controlUnit refID="64F"></controlUnit>
      </trainProtectionElement>

```

```

    <trainProtectionElement id="64F_2_02" name="Gleismagnet2000Hz"
pos="46.220" dir="up" medium="ind" system="PZB" model="2000Hz">
    <controlUnit refID="64F"></controlUnit>
    </trainProtectionElement>
    <trainProtectionElement id="64F_0.5_03" name="Gleismagnet500Hz"
pos="46.470" dir="up" medium="ind" system="PZB" model="500Hz">
    <controlUnit refID="64F"></controlUnit>
    </trainProtectionElement>
    <trainProtectionElement id="590_1_01" name="Gleismagnet1000Hz"
pos="47.304" dir="up" medium="ind" system="PZB" model="1000Hz">
    <controlUnit refID="590"></controlUnit>
    </trainProtectionElement>
    <trainProtectionElement id="590_2_02" name="Gleismagnet2000Hz"
pos="47.304" dir="up" medium="ind" system="PZB" model="2000Hz">
    <controlUnit refID="590"></controlUnit>
    </trainProtectionElement>
    <trainProtectionElement id="590_0.5_03" name="Gleismagnet500Hz"
pos="47.554" dir="up" medium="ind" system="PZB" model="500Hz">
    <controlUnit refID="590"></controlUnit>
    </trainProtectionElement>
    <trainProtectionElement id="608_1_01" name="Gleismagnet1000Hz"
pos="48.452" dir="up" medium="ind" system="PZB" model="1000Hz">
    <controlUnit refID="608"></controlUnit>
    </trainProtectionElement>
    <trainProtectionElement id="608_2_02" name="Gleismagnet2000Hz"
pos="48.452" dir="up" medium="ind" system="PZB" model="2000Hz">
    <controlUnit refID="608"></controlUnit>
    </trainProtectionElement>
  </trainProtectionElements>
</ocsElements>
</track>
</tracks>
</line></lines>
<operationControlPoints>
  <ocp id="DHDG" name="Heidenau-Grosssedlitz"/>
  <ocp id="DPI" name="BF Pirna"/>
</operationControlPoints>
<interlocking>
  <mainSignalBox type="microelectronic" id="ESTW-Z_DPI" name="Stellwerk Pirna">
    <subSignalBox type="microelectronic" id="ESTW-A_DPI_64" name="ESTW-A BF
Pirna">
    <routes>
      <route vMax="60" id="64_F/P1">
        <routeStart refID="64F"></routeStart>
        <routeDestination refID="64P1"></routeDestination>
        <elements>
          <element><switch switchStand="right" refID="64W29"/></element>
          <element><switch switchStand="right" refID="64W28"/></element>
          <element><switch switchStand="right" refID="64W19"/></element>
          <element><switch switchStand="right" refID="64W17"/></element>
          <element><switch switchStand="right" refID="64W16"/></element>
        </elements>
      </route>
    </routes>
  </subSignalBox>
</mainSignalBox>

```

```

        <element><switch switchStand="right" refID="64W11"/></element>
    </elements>
    <flankProtElements>
        <FlankProtElement><switch switchStand="right" refID="64W26"/>
    </FlankProtElement>
        <FlankProtElement><switch switchStand="right" refID="64W15"/>
    </FlankProtElement>
        <!-- Achtung!: Zwieschutzweiche, fehlt noch im Vorschlag-->
        <FlankProtElement><switch switchStand="left" refID="64W18"/>
    </FlankProtElement>
        <FlankProtElement><switch switchStand="right" refID="64W12"/>
    </FlankProtElement>
    </flankProtElements>
    <trackSections><!-- Im Beispiel weggelassen!--></trackSections>
    <overlaps>
        <refOverlap refID="D1_64P1"/>
        <refOverlap refID="D2_64P1"/>
    </overlaps>
</route>
<route vMax="160" id="B1590">
    <routeStart refID="590"></routeStart>
    <routeDestination refID="64F"></routeDestination>
    <overlaps><refOverlap refID="D1_64F"/></overlaps>
</route>
<route vMax="160" id="B1608">
    <routeStart refID="608"></routeStart>
    <routeDestination refID="590"></routeDestination>
    <elements>
        <element>
            <signal type="repeater" refID="VW608" SignalAspect="undef."/>
        </element>
    </elements>
    <overlaps><refOverlap refID="D1_590"/></overlaps>
</route>
</routes>
<trackSections><!-- Im Beispiel weggelassen!--></trackSections>
<overlaps>
    <overlap fromSignal="64P1" id="D1_64P1" vPossible="80" until="64W1"
targetLength="200"></overlap>
    <overlap fromSignal="64P1" id="D2_64P1" vPossible="60" until="64W7"
targetLength="100"></overlap>
    <overlap fromSignal="64F" id="D1_64F" vPossible="160" until="64W29"
targetLength="200"></overlap>
    <overlap fromSignal="590" id="D1_590" vPossible="160" until="km47,504"
targetLength="200"></overlap>
</overlaps>
</subSignalBox>
</mainSignalBox>
</interlocking>
</infrastructure>
</RailML>

```