VDV Die Verkehrsunternehmen

VDV-Schrift 431-2

06/2015

Echtzeit Kommunikations- und Auskunftsplattform EKAP

Teil 2: EKAP-Schnittstellenbeschreibung V1.1

Gesamtbearbeitung
Ausschuss für Kundenservice, -information und -dialog (K3),
Ausschuss für Telematik und Informationssysteme (ATI)

Gefördert durch:

Bundesministerium für Wirtschaft und Energie

aufgrund eines Beschlusses des Deutschen Bundestages

Das dieser VDV-Schrift zugrundeliegende Vorhaben IP-KOM-ÖV wurde mit Mitteln des Bundesministeriums für Wirtschaft und Energie unter dem Förderkennzeichen 19P10003 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Echtzeit Kommunikations- und Auskunftsplattform EKAP

Teil 2: EKAP-Schnittstellenbeschreibung V1.1

Gesamtbearbeitung

Kernteam Information KTI des K3, Unterausschuss „intermodal transport control system“ (UA itcs)

Autorenverzeichnis

M. Sc. Frank Englert, TU Darmstadt, Darmstadt (bis V.1.0)

Dipl.-Inf. Günther Gruber, Mentz Datenverarbeitung GmbH, München (bis V.1.0)

B. Sc. Stephan Großberndt, side by site GmbH & Co. KG, Köln (ab V. 1.1)

Dipl.-Ing. Stephan Hörold, TU Ilmenau, Ilmenau

Dipl.-Inf. Christine Keller, TU Dresden, Dresden (bis V.1.0)

Dipl.-Medieninf. Romina Kühn, TU Dresden, Dresden (bis V.1.0)

Dipl.-Math. Werner Kohl, Mentz Datenverarbeitung GmbH, München

Dipl.-Medienwiss. Cindy Mayas, TU Ilmenau, Ilmenau (bis V.1.0)

Dipl.-Ing. ETH Walter Meier-Leu, Weisskopf Engineering AG, Schaffhausen

Dipl.-Ing. Berthold Radermacher, VDV, Köln

Dipl.-Inform. Anselmo Stelzer, TU Darmstadt, Darmstadt (bis V.1.0)

Dipl.-Inf. Katja Tietze, TU Dresden, Dresden (bis V.1.0)

Dipl.-Ing. (FH) Andreas Wehrmann, VDV, Köln (bis V.1.0)

Dipl.-Ing. Dirk Weißer, Init, Karlsruhe

Dipl.-Math. Peter von Grumbkow, HaCon Ing.-Ges. mbH, Hannover

© Verband Deutscher Verkehrsunternehmen e. V. Köln 2014 | Alle Rechte, einschließlich des Nachdrucks von Auszügen, der fotomechanischen oder datenverarbeitungstechnischen Wiedergabe und der Übersetzung, vorbehalten.

Inhaltsverzeichnis

Vorwort **16**

1. Einleitung **18**

2. Anwendungsbereich **18**

3. Notation der XML-Elemente und -Strukturen **18**

3.1. Darstellung von XML-Elementen im Text 18

3.2. Darstellung von Beziehungen 19

3.3. Tabellennotation von XML-Strukturen 19

3.3.1. Gruppierung 20

3.3.2. Elementname 21

3.3.3. Multiplizität & Choice (Min:Max) 21

3.3.4. Datentyp 21

3.3.5. Erläuterung 21

4. Nachrichtenübermittlung **22**

4.1. Einsatz der SIRI-Verfahren 22

4.2. HTTP und REST 23

4.3. Rollen von Server und Client 24

5. Identifikation von Objekten über Systemgrenzen hinweg **25**

5.1. Haltestellen und Haltepunkte 25

5.2. Orte und Gemeinden 26

5.3. Adressen und POIs 26

5.4. Verkehrsunternehmen und Verkehrsverbünde 27

5.5. Linien und Linienrichtungen 27

5.6. Fahrten 28

5.7. Fahrzeuge 29

5.8. Fahrzeugtypen 29

5.9. Verkehrstage 29

5.10. Eigentümer 29

5.11. Haltestellen- und Fahrzeugeinrichtungen 30

5.12. Teilnehmende Systeme / IT-Systeme 30

5.13. Ereignismeldungen 30

5.14. Tarifverantwortliche 30

5.15. Tarifzonen 30

5.16. Fahrscheine und Vielfahrerkarten 30

6. Dienste und XML-Schemata 32
6.1. Bereitgestellte Dienste 32
6.2. Dienstübergreifend genutzte XML-Schemata 34
6.3. Importierte Schemata 37
6.4. Fehlerzustände beim Betrieb von TRIAS-Diensten 37
6.4.1. Fehlercodes aus SIRI 38
6.4.2. Allgemeine TRIAS-Fehlerzustände 39
7. Gemeinsam genutzte XML-Strukturen 40
7.1. Trias, das Wurzelelement 40
7.1.1. ServiceRequestStructure 40
7.1.2. SubscriptionRequestStructure 40
7.1.3. RequestPayloadStructure 41
7.1.4. ServiceDeliveryStructure 42
7.1.5. DeliveryPayloadStructure 42
7.2. Trias_Utility 43
7.2.1. Einfache Typen 43
7.2.2. InternationalTextStructure 44
7.2.3. GeoPositionStructure 44
7.2.4. WebLinkStructure 44
7.3. Trias_ModesSupport 44
7.3.1. Einfache Typen 44
7.3.2. IndividualTransportOptionsStructure 46
7.3.3. PtSubmodeChoiceGroup 47
7.3.4. ModeStructure 47
7.3.5. PtModeFilterStructure 48
7.4. Trias_Common 48
7.4.1. Einfache Typen 48
7.4.2. ErrorMessageStructure 48
7.4.3. PrivateCodeStructure 49
7.4.4. OperatorFilterStructure 49
7.4.5. LineDirectionStructure 49
7.4.6. LineDirectionFilterStructure 49
7.4.7. SharingServiceStructure 50
7.4.8. OperatingDaysStructure 50
7.4.9. WeekdayTimePeriodStructure 50

7.4.10. GeneralAttributeStructure 51
7.5. Trias_LocationSupport 51
7.5.1. Einfache Typen 51
7.5.2. StopPointStructure 52
7.5.3. StopPlaceStructure 52
7.5.4. LocalityStructure 53
7.5.5. PointOfInterestStructure 53
7.5.6. PointOfInterestCategoryStructure 53
7.5.7. OsmTagStructure 54
7.5.8. PointOfInterestFilterStructure 54
7.5.9. AddressStructure 54
7.5.10. LocationStructure 55
7.5.11. LocationRefStructure 55
7.6. Trias_JourneySupport 56
7.6.1. ServiceViaPointStructure 56
7.6.2. ViaStructure 56
7.6.3. DatedServiceGroup 57
7.6.4. DatedJourneyStructure 57
7.6.5. TripLocationStructure 58
7.6.6. CallAtStopStructure 58
7.6.7. DatedCallAtLocationStructure 59
7.6.8. ContinuousServiceStructure 60
7.6.9. VehiclePositionStructure 60
7.6.10. ProgressBetweenStopsStructure 60
7.6.11. LocationContextStructure 61
7.6.12. AbstractResponseContextStructure 61
7.6.13. LegAttributeStructure 61
7.7. Trias_FacilitySupport 62
7.7.1. siri:CommonFacilityGroup 62
7.7.2. siri:StopFacilityGroup 65
7.7.3. siri:ServiceFacilityGroup 65
7.7.4. siri:AllFacilitiesGroup 66
7.8. Trias_SituationSupport 66
7.8.1. SituationsStructure 66
7.8.2. SituationFullRefStructure 67

7.9. Trias_RequestSupport 68
7.9.1. Einfache Typen 68
7.9.2. AbstractTriasServiceRequestStructure 68
7.9.3. AbstractTriasSubscriptionRequestStructure 69
7.9.4. AbstractTriasResponseStructure 70
7.10. Trias_FaresSupport 71
7.10.1. Einfache Typen 71
7.10.2. FareZoneStructure 71
7.10.3. FareZoneListInAreaStructure 71
7.10.4. BookingInfoStructure 72
7.10.5. TicketStructure 72
7.10.6. TripFaresResultStructure 73
7.10.7. FaresPassengerStructure 73
7.10.8. FaresParamStructure 74
8. Ortsinformationsdienst 75
8.1. Beschreibung 75
8.2. Einfache Datentypen 75
8.3. Anfragestrukturen 76
8.3.1. LocationInformationRequestStructure 76
8.3.2. InitialLocationInputStructure 76
8.3.3. GeoRestrictionsStructure 76
8.3.4. GeoCircleStructure 77
8.3.5. GeoRectangleStructure 77
8.3.6. GeoAreaStructure 77
8.3.7. LocationParamStructure 77
8.4. Antwortstrukturen 79
8.4.1. LocationInformationResponseStructure 79
8.4.2. LocationResultStructure 79
9. Dienst Verbindungsauskunft 80
9.1. Beschreibung 80
9.2. Anfragestrukturen 80
9.2.1. TripRequestStructure 80
9.2.2. TripParamStructure 80
9.2.3. NumberOfResultsGroup 82
9.2.4. NotViaStructure 82

9.2.5. NoChangeAtStructure 82
9.3. Antwortstrukturen 83
9.3.1. TripResponseStructure 83
9.3.2. TripResponseContextStructure 83
9.3.3. TripResultStructure 84
9.3.4. TripStructure 85
9.3.5. TripLegStructure 85
9.3.6. TimedLegStructure 86
9.3.7. InterchangeLegStructure 87
9.3.8. ContinuousLegStructure 88
9.3.9. LegBoardStructure 89
9.3.10. LegAlightStructure 90
9.3.11. LegIntermediateStructure 91
9.3.12. LegTrackStructure 91
9.3.13. TrackSectionStructure 92
9.3.14. NavigationPathStructure 92
9.3.15. NavigationSectionStructure 92
9.3.16. AccessPathStructure 93
10. Dienst Abfahrtstafeln **94**
10.1. Beschreibung 94
10.2. Anfragestrukturen 94
10.2.1. StopEventRequestStructure 94
10.2.2. StopEventParamStructure 94
10.3. Antwortstrukturen 95
10.3.1. StopEventResponseStructure 95
10.3.2. StopEventResponseContextStructure 95
10.3.3. StopEventResultStructure 95
10.3.4. StopEventStructure 96
10.3.5. CallAtNearStopStructure 96
11. Dienst Logische Ortung **97**
11.1. Beschreibung 97
11.2. Anfragestrukturen 97
11.2.1. PositioningRequestStructure 97
11.2.2. PositioningParamStructure 97
11.2.3. TimedPositionStructure 97

11.2.4. TimedStopStructure 98
11.3. Antwortstrukturen 98
11.3.1. PositioningResponseStructure 98
11.3.2. PositioningResultStructure 99
11.3.3. PositioningStructure 99
11.3.4. RankedPositionStructure 99
12. Dienst Fahrtinformation (EKAP) 100
12.1. Beschreibung 100
12.2. Anfragestrukturen 100
12.2.1. TripInfoRequestStructure 100
12.2.2. TripInfoParamStructure 100
12.3. Antwortstrukturen 101
12.3.1. TripInfoResponseStructure 101
12.3.2. TripInfoResponseContextStructure 101
12.3.3. TripInfoResultStructure 101
13. Anschlussdienste 102
13.1. Beschreibung 102
13.1.1. Dienst Anschlussvoranmeldung 103
13.1.2. Dienst Anschlussstatus 103
13.1.3. Dienst Info bei Anschlussverlust 104
13.1.4. Dienst Anschlussrückmeldung 104
13.2. Einfache Typen 104
13.3. Komplexe Strukturen 105
13.3.1. DatedConnectionStructure 105
13.3.2. FeederDistributorStructure 105
13.3.3. GeneralizedConnectionStructure 105
13.3.4. ConnectionStatusStructure 106
13.3.5. RecommendationStructure 106
13.4. Anfragestrukturen Anschlussmeldung 106
13.4.1. ConnectionDemandRequestStructure 106
13.4.2. ConnectionDemandDeleteRequestStructure 107
13.5. Antwortstrukturen Anschlussmeldung 107
13.5.1. ConnectionDemandResponseStructure 107
13.5.2. ConnectionDemandDeleteResponseStructure 107
13.6. Anfragestrukturen Anschlussstatus 108

13.6.1. ConnectionStatusRequestStructure 108
13.6.2. ConnectionStatusNotificationStructure 108
13.7. Antwortstrukturen Anschlussstatus 109
13.7.1. ConnectionStatusResponseStructure 109
13.8. Anfragestrukturen Anschlussrückmeldung 109
13.8.1. ConnectionReportRequestStructure 109
13.9. Antwortstrukturen Anschlussrückmeldung 110
13.9.1. ConnectionReportResponseStructure 110
14. Dienst Fahrpreis- und Tarifberechnung **110**
14.1. Beschreibung 110
14.2. Anfragestrukturen 110
14.2.1. FaresRequestStructure 110
14.2.2. StopFaresRequestStructure 111
14.2.3. StaticFaresRequestStructure 111
14.2.4. TripFaresRequestStructure 111
14.2.5. MultiTripFaresRequestStructure 111
14.3. Antwortstrukturen 112
14.3.1. FaresResponseStructure 112
14.3.2. FaresResultStructure 112
14.3.3. StopFaresResultStructure 112
14.3.4. StaticFaresResultStructure 113
14.3.5. TripTicketReferenceStructure 113
14.3.6. MultiTripFaresResultStructure 113
15. Dienst Buchungsinformationen **115**
15.1. Beschreibung 115
15.2. Anfragestrukturen 115
15.2.1. BookingInfoRequestStructure 115
15.3. Antwortstrukturen 115
15.3.1. BookingInfoResponseStructure 115
15.3.2. BookingInfoResultStructure 116
16. Dienst IV-Routing **117**
16.1. Einfache Typen 117
16.2. Anfragestrukturen 117
16.2.1. IndividualRouteRequestStructure 117
16.2.2. IndividualTripParamStructure 117

16.2.3. IndividualRouteContextStructure 119
16.3. Antwortstrukturen 119
16.3.1. IndividualRouteResponseStructure 119
16.3.2. RouteResultStructure 120
16.3.3. RouteStructure 121
17. Dienst Kartendienst **122**
17.1. Einfache Datentypen 122
17.2. Anfragestrukturen 122
17.2.1. MapServiceRequestStructure 122
17.2.2. ImageCoordinatesRequestStructure 123
17.2.3. GeoCoordinatesRequestStructure 123
17.2.4. MapAspectStructure 123
17.2.5. MapSizeStructure 124
17.2.6. MapCoordinateStructure 124
17.3. Antwortstrukturen 124
17.3.1. MapServiceResponseStructure 124
17.3.2. ImageCoordinatesResponseStructure 125
17.3.3. GeoCoordinatesResponseStructure 125
17.3.4. MapResultStructure 126
17.3.5. ImagePointResultStructure 126
17.3.6. GeoCoordinateResultStructure 127
18. Dienst Schadensmeldung / Zustand von Einrichtungen **128**
18.1. Beschreibung 128
18.2. Einfache Typen 128
18.3. Komplexe Strukturen 128
18.3.1. FacilityStructure 128
18.3.2. VehicleFacilityStructure 128
18.3.3. InfrastructureFacilityStructure 129
18.3.4. VehicleFacilityRefGroup 129
18.3.5. InfrastructureFacilityRefGroup 129
18.3.6. FacilityDataFilterGroup 130
18.3.7. FacilityStatusStructure 130
18.4. Anfrage Schadensmeldung 130
18.4.1. FacilityStatusReportStructure 130
18.5. Antwort Schadensmeldung 131

18.5.1. FacilityStatusReportResponseStructure 131
18.6. Anfrage Zustand von Einrichtungen 131
18.6.1. FacilityRequestStructure 131
18.7. Antwort Zustand von Einrichtungen 131
18.7.1. FacilityResponseStructure 132
18.7.2. FacilityResultStructure 132
19. Dienst Benachrichtigungsdienst **133**
19.1. Beschreibung 133
19.2. Komplexe Strukturen 134
19.2.1. TripMonitoringParamStructure 134
19.3. Anfragestrukturen 135
19.3.1. TripMonitoringSubscriptionRequestStructure 135
19.4. Antwortstrukturen 135
19.4.1. TripMonitoringDeliveryStructure 135
20. Dienst Personalisierungsdienst **137**
20.1. Beschreibung 137
20.2. Interaktionen 137
20.2.1. Lebenszyklus eines Wertes 137
20.2.2. Werteliste ermitteln 138
20.2.3. Werte speichern und abrufen 139
20.3. Einfache Typen 141
20.4. Anfragestrukturen 141
20.4.1. PersonalisationRequestStructure 141
20.4.2. PersonalisationSaveValueRequestStructure 141
20.4.3. PersonalisationRetrieveValueRequestStructure 141
20.4.4. PersonalisationDeleteValueRequestStructure 142
20.4.5. PersonalisationEnumerateValuesRequestStructure 142
20.5. Antwortstrukturen 142
20.5.1. PersonalisationResponseStructure 142
20.5.2. PersonalisationSaveValueResponseStructure 142
20.5.3. PersonalisationRetrieveValueResponseStructure 143
20.5.4. PersonalisationDeleteValueResponseStructure 143
20.5.5. PersonalisationEnumerateValuesResponseStructure 143
21. Dienst Fahrzeuginformationen **144**
21.1. Beschreibung 144

21.2. Anfragestrukturen 144
21.2.1. VehicleDataRequestStructure 144
21.3. Antwortstrukturen 144
21.3.1. VehicleDataResponseStructure 144
21.3.2. VehicleStatusStructure 144
21.3.3. VehicleActivityStructure 145
21.3.4. StopSequenceStructure 145
21.3.5. StopInformationStructure 145
21.3.6. DisplayContentStructure 146
21.3.7. LineInformationStructure 146
21.3.8. DestinationStructure 146
21.3.9. ViaPointStructure 146
21.3.10. AnnouncementStructure 147
21.3.11. ConnectionStructure 147
21.3.12. VehicleTypeStructure 147
22. Dienst Haltewunsch an Fahrzeug **148**
22.1. Beschreibung 148
22.2. Anfragestrukturen 148
22.2.1. StopRequestRequestStructure 148
22.3. Antwortstrukturen 148
22.3.1. StopRequestResponseStructure 148
23. Dienst Diensteregister **149**
23.1. Beschreibung 149
23.2. Einfache Typen 149
23.3. Anfragestrukturen 149
23.3.1. ServiceRegisterRequestStructure 149
23.3.2. ServiceRegisterParamStructure 150
23.3.3. ServiceRegisterRegisterRequestStructure 150
23.3.4. ServiceRegisterUpdateRequestStructure 150
23.3.5. ServiceRegisterLookupRequestStructure 151
23.3.6. ServiceRegisterUnregisterRequestStructure 151
23.4. Antwortstrukturen 151
23.4.1. ServiceRegisterResponseStructure 151
23.4.2. ServiceRegisterResultStructure 152
23.4.3. ServiceRegisterRegisterResponseStructure 152

23.4.4. ServiceRegisterUpdateResponseStructure 152
23.4.5. ServiceRegisterLookupResponseStructure 152
23.4.6. ServiceRegisterUnregisterResponseStructure 152
23.4.7. TriasServiceStructure 153
24. Dienst Authentifizierung **154**
25. Normative Referenzen **156**
26. Begriffe **156**
27. Versionshistorie **157**
27.1. Version 1.1 (Schrift und Schema) 157
27.1.1. Funktionale Erweiterungen 157
27.1.2. Technische Ergänzungen/Korrekturen 157
27.1.3. Dokumentarische Korrekturen 158
28. Abkürzungen **159**
29. Abbildungsverzeichnis **160**
30. Tabellenverzeichnis **160**
31. Literaturverzeichnis **169**
Impressum **170**

Vorwort

Auf Initiative des VDV und gefördert durch das BMWi begann im September 2010 das Forschungs- und Standardisierungsprojekt

Internet Protokoll basierte Kommunikationsdienste im öffentlichen Verkehr (IP-KOM-ÖV).

Das Projekt wird von 14 Partnern aus Industrie, Universitäten und Verkehrsunternehmen getragen. Es dient der Erarbeitung moderner Kommunikationskonzepte für die umfassende und kontinuierliche Fahrgastinformation.

Eine umfassende Fahrgastinformation stellt heutzutage ein entscheidendes Wettbewerbsmerkmal im öffentlichen Personenverkehr dar, nicht nur im Vergleich mit anderen Verkehrsunternehmen sondern auch im Vergleich zum Individualverkehr.

Bereits heute ist es üblich, dass Verkehrsunternehmen ihre Fahrgäste nicht nur über die geplanten Fahrten informieren, sondern auch Echtzeitinformationen z. B. zu Verspätungen, Störungen oder Fahrtzieländerungen bereitstellen. Diese Informationen werden zum einen über öffentliche Anzeiger bzw. Ansagen in Fahrzeugen oder an Haltestellen allen dort befindlichen Personen zur Verfügung gestellt. Zum anderen lassen sich solche Informationen mit speziellen Applikationen oder über Web-Angebote individuell abfragen.

Bislang ist es aber nicht möglich, Fahrgäste im öffentlichen Verkehr direkt mit Informationen zu ihrer persönlich relevanten Fahrt zu versorgen, den Fahrgast also auch im Störungsfall mit Hilfe des öffentlichen Verkehrs auf dem schnellsten Weg zu seinem Ziel zu führen.

Die weit verbreiteten Smartphones und Tablets bieten hierfür vielfältige Möglichkeiten und ermöglichen eine hohe Akzeptanz der Benutzer. Die Informationsübertragung erfolgt dabei IP-basiert und sollte bevorzugt zwischen einem zentralen Informations-Server und dem Kundenendgerät erfolgen. Für den Fall, dass der zentrale Datenserver nicht erreichbar ist, sollte auch eine Kommunikation zwischen Kundenendgerät und Fahrzeug möglich sein.

Das Forschungs- und Standardisierungsprojekt IP-KOM-ÖV arbeitet deshalb an drei Schwerpunkten (vgl. Abbildung 1).

Erster Schwerpunkt (grün in Abbildung 1) ist die Spezifikation eines performanten IP-basierten Kommunikationsprotokolls im Fahrzeug (IBIS-IP). Dabei geht es zum einen darum, den gewachsenen Bedürfnissen der Fahrgastinformation gerecht zu werden und zum anderen um die Definition einer IP-basierten Schnittstelle zur Übertragung der Informationen vom Fahrzeug zum mobilen Kundenendgerät. Hierzu wird der in den achtziger Jahren entwickelte IBIS-Wagenbus aus der VDV-Schrift 300 auf eine moderne Ethernet-Informationsarchitektur umgesetzt.

Zweiter Schwerpunkt (rot in Abbildung 1) ist die individuelle Fahrgastinformation unter Verwendung mobiler Geräte des Fahrgasts (Smartphones, Tablet-PC u. ä.) Hierzu wurden im ersten Schritt die Bedürfnisse von Fahrgästen zu individuellen Informationen ermittelt. Im zweiten Schritt werden einheitliche Schnittstellen zwischen der Echtzeit-Kommunikations- und Auskunftsplattform (EKAP) und den mobilen Kundenendgeräten bzw. zwischen der EKAP und den Hintergrundsystemen entwickelt. Hierbei werden ausschließlich die Datenmodellierungen und Architekturen erforscht und spezifiziert.

Aufbauend auf diesen Datenmodellierungen werden semantische Modelle erarbeitet, die helfen, die Fahrgastinformationsdaten für Kommunikationsdienste auf Basis von innovativen Technologien des Semantic Web zur Verfügung zu stellen. Die Entwicklung einer Applikation für mobile Endgeräte ist ausdrücklich nicht vorgesehen.

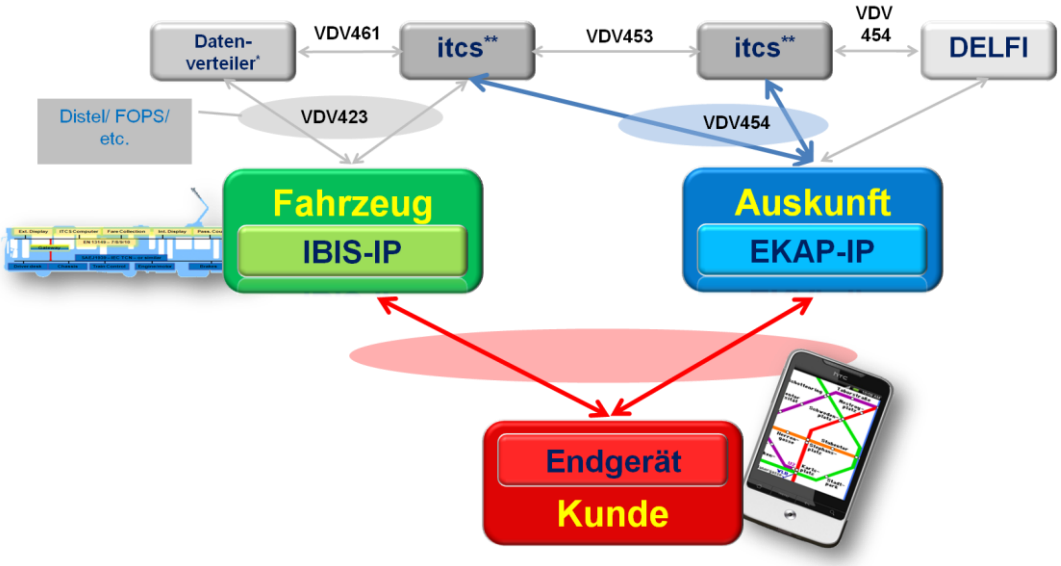

Abbildung 1: Umfeld und Schwerpunkte im Projekt IP-KOM-ÖV

Dritter Schwerpunkt (blau in Abbildung 1) ist die Definition und Schaffung einer Echtzeit-Kommunikations- und Auskunftsplattform (EKAP). Die EKAP bündelt Informationen von itcs- und anderen Auskunfts- und Informationssystemen und stellt die Vielzahl an Informationen über geeignete Schnittstellen den Applikationen auf den Kundenendgeräten zur Verfügung. Diese Plattform ermöglicht es, Kunden dynamisch mit individuellen Störungsmeldungen versorgen zu können.

Neben den Forschungsarbeiten ist die Standardisierung der Ergebnisse ein wesentliches Ziel des Projektes, um eine nachhaltige Nutzung zu gewährleisten.

Darüber hinaus wird die Praxistauglichkeit dieses neuen Standards in Labor- und Feldtests verifiziert.

1. Einleitung

In diesem Dokument werden die Dienste, die in VDV-Schrift 430 und VDV-Schrift 431-1 beschrieben sind, als XML-Schnittstellen definiert. Dadurch entstehen Schnittstellenstandards, die es Software-Entwicklern und Unternehmen erlauben, Anwendungen zu realisieren, wo mobile Apps der Fahrgäste, Fahrzeuge, Portalsysteme und echtzeitfähige Auskunftssysteme (EKAPs) miteinander kommunizieren.

Bei der Ausarbeitung dieser Schnittstellendefinitionen wurde Wert darauf gelegt, Kompatibilität zu anderen Standards auf dem Sektor des öffentlichen Verkehrs herzustellen. Hier sind vor allem TransModel als Begriffsglossar, IFOPT für die Modellierung von Haltestellen, SIRI für den Austausch von Echtzeitdaten und für sein ausgefeiltes Nachrichtenaustauschverfahren, sowie JourneyWeb und DELFI als Schnittstellen zum Abrufen von Fahrplaninformationen und Verbindungsauskünften zu nennen.

2. Anwendungsbereich

Die in diesem Dokument definierten Schnittstellen spezifizieren Dienste, die zum einen zwischen mobilen Apps und Fahrzeugen des ÖV Verwendung finden, zum anderen zwischen Portalsystemen und Auskunftssystemen (EKAPs) als Hintergrundsystem. Sie sind für den Zweck der Interaktion mit dem Fahrgast gedacht. In erster Linie soll der Fahrgast informiert werden. Es gibt aber auch Dienste, bei denen der Fahrgast von sich aus aktiv wird, so z. B. beim Haltewunsch oder der Anschlussvoranmeldung.

3. Notation der XML-Elemente und -Strukturen

Die in diesem Dokument vorgestellen TRIAS[1]-Schnittstellen werden mit Hilfe von XML-Schema definiert. Die Objekte, die über die Schnittstelle ausgetauscht werden, liegen folglich als XML-Elemente vor. Die Beschreibung der XML-Elemente wird in diesem Dokument in einer Tabellenform vorgenommen, die aus SIRI (CEN, TS 15531 Part 1) stammt. Sie ist sehr kompakt und übersichtlich und bietet eine Vielzahl an strukturellen Informationen, die ansonsten nur in der XML-Schema-Definition sichtbar wird. Dieses Kapitel erläutert die Notation der Tabellenform, die ab Kapitel 7 intensiv genutzt wird.

Alle Namen von Elementen, Datentypen und Attributen sind in Englisch gehalten, um eine etwaige Normierung auf europäischer Ebene vorzubereiten und den Austausch mit europäischen Partnern zu erleichtern.

3.1. Darstellung von XML-Elementen im Text

In diesem Dokument soll eine konsistente Notation der XML-Elemente helfen, technisch wichtige Information beim Lesen bereit zu stellen.

- XML-Elemente werden in Groß-Klein-Schreibweise (Upper Camel Case) fett und kursiv geschrieben, z. B.: ***VehicleJourneyRef***. Die Elementnamen sind – wo immer möglich und sinnvoll – an Begriffe aus TransModel angelehnt. Fehlt in TransModel

[1] Travellers Realtime Information and Advisory Standard

ein geeigneter Begriff für ein Konzept oder Objekt, so wurde versucht, den entsprechenden Begriff aus JourneyWeb oder das passende Konzept aus DELFI zu übernehmen.
- Datenypen werden kursiv dargestellt, z. B.: *xsd:boolean.*
- Code-Beispiele werden in kleinerer Schrift wiedergegeben.

3.2. Darstellung von Beziehungen

Beziehungen zwischen Objekten können mittels

- impliziter Mechanismen,
- internen Referenzen oder
- externen Referenzen

ausgedrückt werden. Ein impliziter Mechanismus ist z. B. das Enthaltensein eines Elements in einem anderen. Damit wird eine unmittelbare Kindbeziehung ausgedrückt. Eine interne Referenz ist ein Objektschlüssel, der innerhalb der Schnittstelle definiert wird (z. B. ein Identifikator einer Meldung). Eine externe Referenz ist ein Objektschlüssel, der außerhalb der Schnittstelle festgelegt wird (z. B. eine Haltestellennummer). Externe Referenzen bestehen manchmal auch aus zusammengesetzten Schlüsseln (siehe die ausführliche Darstellung in Kapitel 5).

Es ist wichtig, den Unterschied zwischen einem Identifikator (Objektschlüssel) und einer Referenz auf das Objekt festzuhalten. In TRIAS gelten folgende Regeln:

- Ein Identifikator ist ein Kindelement des definierenden Elements, das einen eindeutigen Code (Primärschlüssel) für das definierende Element angibt. Diese Identifikatoren enden auf ein signalisierendes Hauptwort wie „Code“ oder „Identifier“ (manchmal auch „Number“ in SIRI), z. B. erhält eine Fahrplanfahrt (Journey) den Schlüssel ***JourneyCode***.
- Wird ein Objekt von einem anderen Objekt aus referenziert, endet das referenzierende Element (Fremdschlüssel) auf “Ref”. Zum Bespiel lautet die Referenz auf eine Fahrplanfahrt (etwas aus einer Abfahrtstafel heraus): JourneyRef.
- Die Instanz eines Objekts und die Referenz darauf verwenden einen gemeinsamen zugrunde liegenden Datentyp. Zum Bespiel sind JourneyCode und JourneyRef beide vom Typ JourneyCodeType.

Tabellennotation von XML-Strukturen

In diesem Dokument werden XML-Strukturen in einer Tabellennotation dargestellt (vgl. Tabelle 1). Für jedes wichtige TRIAS-Anfrage/Antwort-Element findet sich eine eigene Tabelle. Weitere Tabellen werden für alle wesentlichen Kindelemente, aus denen die komplexen Strukturen aufgebaut sind, angegeben. Um Platz zu sparen, werden die Spaltenüberschriften nur im Beispiel in Tabelle 1 angezeigt und bei allen folgenden Tabellen nicht wiederholt. In den Tabellen wird ein konsistenter Satz an Regeln zur Beschreibung der XML-Elemente und der daran geknüpften Bedingungen verwendet.

Grup-pierung	Elementname	Min : Max	Datentyp	Erläuterung
ContinuousServiceStructure			*+Structure*	Eine Fahrgastbewegung mit Hilfe eines kontinuierlichen, nicht fahrplangebundenen Verkehrsmittels.
	a ***ContinuousMode***	**-1:1**	*walk | demandResponsive | replacement-Service*	Modalität für kontinuierliche Verkehre
	b ***IndividualMode***		*walk | cycle | taxi | self-drive-car | others-drive-car | motorcycle | truck*	Verkehrsmittelmodalität für Individualverkehr
Dated-Service	***OperatingDay***	**1:1**	*→Operating-Day*	Betriebstag der Fahrt. Vgl. 7.4.1.
	VehicleRef	0:1	*→Vehicle*	Fahrzeug-ID. Vgl. 7.4.1.
Service-Journey	***JourneyRef***	**1:1**	*→Journey*	Fahrt-ID. Vgl. 7.4.1.
LineIdentity	***LineRef***	**1:1**	*→Line*	Linien-ID. Vgl. 7.4.1.
	DirectionRef	**1:1**	*→Direction*	Richtungs-ID. Vgl. 7.4.1.
Service	***Mode***	**1:1**	*+Mode*	Verkehrsmitteltyp. Vgl. 7.3.4.
	PublishedLineName	**1:1**	*International-Text*	Liniennummer oder -name, wie in der Öffentlichkeit bekannt.
	OperatorRef	0:1	*→Operator*	Operator-ID. Vgl. 7.4.1.
	RouteDescription	0:1	*International-Text*	Beschreibung des Fahrwegs.
	Via	0:*	*+ServiceViaPoint*	Wichtige Halte auf dem Fahrweg. Vgl. 7.6.1.
	Attribute	0:*	*+GeneralAttribute*	Hinweise und Attribute (mit Klassifikationen) zur Fahrt. Vgl. 7.4.10.
ServiceOrigin	*OriginStopPointRef*	0:1	*→StopPoint*	ID des ersten Haltepunkts der Fahrt; Starthaltestelle. Vgl. 7.5.1.
	OriginText	0:1	*International-Text*	Name des ersten Haltepunkts der Fahrt, der Starthaltestelle.
Service-Destination	*DestinationStopPointRef*	0:1	*→StopPoint*	ID des letzten Haltepunkts der Fahrt; Endhaltestelle. Vgl. 7.5.1.
	DestinationText	0:1	*International-Text*	Name des letzten Haltepunkts der Fahrt, der Endhaltestelle.
	SituationFullRef	0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im Kontext der Meldung (ResponseContext) zu finden sein oder auf anderem Wege bekannt gemacht werden.

Tabelle 1: Beispiel (aus einem späteren Abschnitt) für die tabellarische Notation einer XML-Struktur

3.2.1. Gruppierung

In der ersten Spalte befindet sich gelegentlich ein Bezeichner, der die Elemente in sinnvolle Gruppierungen einteilt, z. B. *Service* oder *ServiceOrigin*. Dies dient rein zu Dokumentationszwecken und entspricht in den meisten Fällen den Namen einer XML-Gruppe, die im XML-Schema verwendet wurde. Die Verwendung von Gruppierungen hat nur den Zweck, die Elemente zu organisieren und damit für mehr Klarheit und bessere Wiederverwendbarkeit zu sorgen.

3.2.2. Elementname

Elementnamen werden kursiv in der zweiten Spalte wiedergegeben, z. B. *OperatingDay*. Handelt es sich um ein verpflichtendes Element, so wird es **fett** gedruckt. Optionale Elemente werden nicht fett gedruckt. Der Name der Struktur selbst ist links oben in der Tabelle angegeben.

Elemente, die geerbt (XML: “derived by extension”) oder anonym verwendet werden, tragen im Namensfeld drei Doppelpunkte “:::” zur Kennzeichnung.

3.2.3. Multiplizität & Choice (Min:Max)

Die Bedingungen, ob ein Element verpflichtend oder optional ist oder ob es einfach oder mehrfach innerhalb des übergeordneten Elements auftreten kann, werden in der dritten Spalte Min:Max angegeben. Dabei werden die üblichen UML-Konventionen „min:max“ angewendet, so steht z. B. „0:1“ für ein optionales, einfaches Element, „**1:1**“ zeigt ein verpflichtendes, einfaches Element an, „0:*“ steht für ein optionales, mehrfaches Element usw. Verpflichtende Elemente werden **fett** gedruckt.

In manchen Fällen muss ein Element aus seiner Menge ausgewählt werden (XML-Choice). Dies wird durch ein vorangestelltes Minuszeichen symbolisiert, z. B. „**-1:1**“. In diesem Fall steht vor dem Elementnamen noch ein Kleinbuchstabe, der die Auflistung der Wahlöglichkeiten anzeigt. Bei optionalen Auswahlmöglichkeiten (Choices) steht im Min-Wert eine Null: „-0:1“.

3.2.4. Datentyp

Die Datentypen werden in der vierten Spalte kursiv angegeben, z. B. *InternationalText*. Falls der Namensraum (namespace) vom TRIAS-Namensraum abweicht, wird er mitangegeben, z. B. „*xs:dateTime*“ oder „*siri:PtSituationElement*“.

- Ein komplexer Datentyp, der selbst Strukturen als Kindelemente enthält, wird in der Spalte Datentyp mit „+*Structure*“ gekennzeichnet.
- Wo Elemente als Referenzen (Fremdschlüssel) auf andere Objekte verwendet werden, wird als Datentyp der Typ des referenzierten Objekts mit vorangestelltem Pfeil verwendet. Zum Beispiel „→*StopPoint*“ als Typ einer Referenz (*StopPointRefStructure*) auf ein Objekt vom Typ „StopPointType“.
- Aufzählungstypen (Enumerated types) werden an den meisten Stellen unmittelbar mit den verwendbaren Werten dargestellt, z. B. „*walk* | *cycle*“. Nur in einigen Fällen mit sehr umfangreichen Aufzählungen, die an mehreren Stellen wiederverwendet werden, wird ein Typ deklariert und referenziert.
- Um Platz zu sparen werden bei der Angabe der Datentypen Abkürzungen verwendet, z. B. wird auf die Endungen “Structure” und “Type” durchgehend verzichtet. Statt beispielsweise „*InternationalTextStructure*“ wird also immer „*InternationalText*“ als Datentyp angegeben.

3.2.5. Erläuterung

Alle Elemente erhalten in der letzten Spalte eine Erläuterung ihres Verwendungszwecks. An vielen Stellen wird auf weitere passagen im Text hingewiesen, so z. B. bei komplexen Kindelementen an die Stelle, wo ihre Tabellenbeschreibung zu finden ist. An einigen Stellen ist die Erläuterung zu umfangreich und würde die Tabellenform sprengen. Dann finden sich diese Anmerkungen im Text unterhalb der Tabelle.

4. Nachrichtenübermittlung

In diesem Kapitel wird erläutert, wie TRIAS-Nachrichten ausgetauscht werden. Es kommen zwei grundlegende Verfahren zum Einsatz

- Anfrage mit synchroner Antwort (Request-Response-Verfahren),
- Abonnements mit asynchronen Nachrichten (Publish-Subscribe-Verfahren).

Diese Verfahren sind bereits etabliert und im Einsatz, z. B. in den SIRI-Schnittstellen.

4.1. Einsatz der SIRI-Verfahren

In SIRI wurden die eingangs aufgezählten Nachrichtenübermittlungsverfahren bereits definiert und beschrieben, vgl. (CEN, TS 15531, Part 2). Daher werden diese Verfahren hier aufgegriffen. Das hat zum einen den Vorteil, dass bereits getestete Verfahren verwendet werden können, zum anderen kann bei der Implementierung der TRIAS-Dienste evtl. auf eine bereits vorhandene SIRI-Implementierung zurückgegriffen werden, was Kosten und Zeit sparen kann.

Das grundlegende Verfahren ist die Anfrage mit synchroner Antwort. Ein Client stellt eine Anfrage an einen Server, der unmittelbar antwortet. In der SIRI-Terminologie ist der Anfrager der *Data Consumer*, der antwortende Server wird mit *Data Producer* bezeichnet (vgl. Abbildung 2).

Abbildung 2: Anfrage mit synchroner Antwort (Abbildung entnommen aus SIRI, (CEN, TS 15531, Part 2)).

Anfragen mit synchroner Antwort werden bei fast allen TRIAS-Diensten verwendet (eine Ausnahme ist nur der Benachrichtigungsdienst). Die Rolle des Anfragers übernimmt z. B. das Portalsystem, das Anfragen an die EKAP richtet. Aber auch die mobile App stellt Anfragen an das Fahrzeug oder EKAP-Komponenten stellen untereinander Anfragen.

Etwas komplizierter ist der Abonnement-Mechanismus. Ein Datenkonsument interessiert sich für neue Nachrichten, weiß aber nicht, wann diese auftreten werden. Statt regelmäßig nachzufragen und so eine Grundlast zu erzeugen (und zu riskieren, dass er von der neuen Nachricht erst erfährt, wenn er das nächste Mal nachfragt), kann er ein Abonnement einrichten.

Abbildung 3 zeigt die grundlegenden Zusammenhänge. Der Datenkonsument hat zwei Rollen zu erfüllen, die des Abonnenten (Subscriber) und die des Empfängers von Nachrichten (Notification Consumer). Der Datenkonsument bittet den Server um die Einrichtung eines Abonnements (Subscription Request). Dabei teilt er dem Server mit, bei welcher Art Ereignisse er infomiert werden möchte. Der Server richtet das Abonnement ein, indem er es beim Abo-Verwalter (Subscription Manager) registriert. Danach geschieht erst etwas, wenn ein Ereignis eintritt, das dem Konsumenten zu melden ist. In diesem Fall schickt der Server als Benachrichtigungsersteller (Notification Producer) dem Datenkonsumenten die Nachricht mit dem neuen Ereignis (Delivery). Dies wiederholt sich so lange, bis das Abonnement ausläuft oder vom Datenkonsumenten beendet wird.

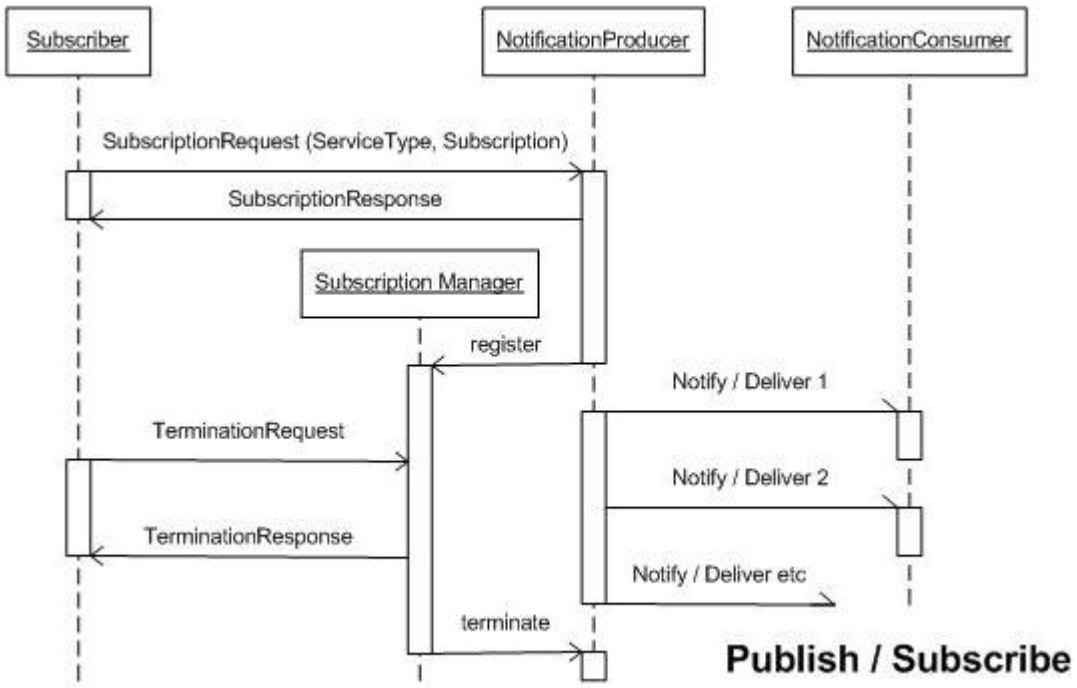

Abbildung 3: Abonnement-Verfahren mit asynchronen Benachrichtigungen (Abbildung entnommen aus SIRI, (CEN, TS 15531, Part 2)).

Sowohl Client als auch Server haben zwei Rollen zu erfüllen, nämlich der Client die Rolle des Subscribers und des Notification Consumers, der Server die Rolle des Notification Producers und des Subscription Managers. In den meisten Implementierungen wird dies aber nicht unterschieden und jeweils eine einzige Softwarekomponente erfüllt beide Rollen.

Das Abo-Verfahren wird komplettiert durch weitere Anfragen. Die Statusanfrage erlaubt es, den Status des Schnittstellenpartners abzufragen und dessen Verfügbarkeit zu testen. Die Heartbeat-Anfrage, die ein Server regelmäßig sendet, ermöglicht es dem Datenkonsumenten umgekehrt zu erkennen, wann ein Server verfügbar ist und Signale (Ping oder Heartbeat) aussendet. Details dazu finden sich in SIRI (CEN, TS 15531, Part 2), Kapitel 5.

In den TRIAS-Diensten kommt der Abo-Mechanismus beim Benachrichtigungsdienst vor, wo ein Datenkonsument (Portalsystem oder mobile App) sich über Störungen oder andere Ereignisse und Vorkommnisse informieren lassen will.

4.2. HTTP und REST

Die Umsetzung der SIRI-Nachrichtenverfahren geschieht in TRIAS mit Hilfe von HTTP/1.1 (Hypertext Transfer Protocol[2]) als Transportprotokoll und XML (Extensible Markup Language[3]) für die Nachrichteninhalte.

Eine HTTP-Anfrage wird vom Server unmittelbar unter Nutzung des schon geöffneten IP-Ports beantwortet. Zum Beispiel sendet ein Client eine Anfrage nach einer Verbindungsauskunft als HTTP-Anfrage mit dem XML-Element *Trias* und *TripRequest* als einem der Kin-

[2] http://tools.ietf.org/html/rfc2616

3 http://www.w3.org/XML/

delemente im POST-Block. Der Server antwortet synchron in der HTTP-Antwort mit dem XML-Element *Trias* und *TripResponse* als einem der Kindelemente.

Falls mehrere Anfragen in schneller Folge abgesendet werden, kann der HTTP-Mechanismus „Keep-Alive“ zum Einsatz kommen, bei dem der bereits geöffnete Port eine Zeit lang leben bleibt und wiederbenutzt werden kann, um häufiges Öffnen und Schließen des Ports zu sparen.

Für größere Nachrichten empfiehlt sich der Einsatz eines Komprimierungsverfahrens. Solche Methoden sind ebenfalls für HTTP spezifiziert.

4.3. Rollen von Server und Client

Bei der Nutzung des synchronen Anfrage-Antwort-Verfahrens ist der Datenkonsument (der Anfrager) ein HTTP-Client, der Datenproduzent (der antwortende Server) ein HTTP-Server.

Lediglich beim Benachrichtigungsdienst, wenn das Abonnement-Verfahren zum Einsatz kommt, ist die Lage komplizierter. Hier müssen sowohl Datenkonsument als auch Datenproduzent die Rollen von Client und Server im HTTP-Sinne beide ausfüllen. Denn in dem Moment, wo der Datenproduzent (Notification Producer) eine neue Nachricht an den Datenkonsumenten senden will, wird er zum Client im HTTP-Sinn und der Datenkonsument zum Server im HTTP-Sinn.

5. Identifikation von Objekten über Systemgrenzen hinweg

Damit verschiedene Systeme dasselbe Objekt referenzieren können, ist eine Objekt-ID notwendig, die allen Systemen bekannt ist. Im Rahmen der TRIAS-Schnittstellen sind Haltestellen, Linien und Verkehrsunternehmen Beispiele für solche Objekttypen, zu denen Informationen über die Schnittstellendienste ausgetauscht werden. Daher braucht man für sie (und weitere Objekttypen) Referenzierungssysteme, die allgemein bekannt sind und verwendet werden können.

Das bedeutet nicht notwendigerweise, dass ein Softwaresystem diese Objektschlüssel auch selbst im Betrieb verwenden muss. Es genügt, wenn es die allgemeinen Objektreferenzen verstehen und auf die intern verwendeten Identifikatoren abbilden kann.

Für die in diesem Kapitel vorgestellten Schemata zur Objektreferenzierung wird eine an IFOPT angelehnte Syntax verwendet. Sie benutzt den Doppelpunkt zur Abgrenzung von Namensräumen. Aus diesem Grund ist ein Doppelpunkt ein syntaktisches Trennzeichen und darf in Identifikatoren nicht verwendet werden.

In den folgenden Abschnitten wird für verschiedene Objekttypen vorgestellt, welche Referenzierungssysteme verwendet werden sollen.

5.1. Haltestellen und Haltepunkte

Für die Referenzierung von Haltestellen und Haltepunkten gibt es von CEN die europäische Norm IFOPT (CEN, EN 28701:2012, 2012). Dort ist in Kapitel 6.8.1 eine Syntax für den Aufbau eines Referenzierungsschlüssels vorgesehen. Einige Systeme in Deutschland unterstützen diese Syntax bereits. Eine bundesweite Einführung wird im BMVBS-Projekt DELFIplus vorbereitet. In den TRIAS-Schnittstellen sollen die Ergebnisse aus diesem Projekt zur Anwendung kommen.

Aufbau eines IFOPT-Objektschlüssels
Länderkürzel:Region:Haltestellennummer:Bereich:Haltepunkt

Das folgende Beispiel zeigt den (hierarchischen) Aufbau der Schlüssel für eine Haltestelle, einen Haltestellenbereich und einen Haltepunkt

Praxis-Beispiel:

Haltestelle Karlsplatz (Stachus) in München:

Haltestellenobjekt	Eindeutige ID
Haltestelle Karlsplatz (Stachus) in München	de:9162:1
Haltestellenbereich U-Bahn U4/5	de:9162:1:2
Haltepunkt U4/5 Richtung Odeonsplatz	de:9162:1:2:URiOd

Client-Systeme, die selbst keine eigene Datenversorgung haben, können die Objektreferenzen für Haltestellen und Haltepunkte mit Hilfe des TRIAS-Schnittstellendiensts Ortsauflösung (vgl. 8.1) von der EKAP beziehen.

5.2. Orte und Gemeinden

Zur eindeutigen Referenzierung von Gemeinden existiert in Deutschland der Amtliche Gemeindeschlüssel (AGS[4]), früher auch Gemeindekennziffer (GKZ) genannt. Für die Orte innerhalb einer Gemeinde ist die Situation je nach Bundesland unterschiedlich. In Bayern z. B. gibt es je Gemeinde eine Liste von Orten mit amtlich festgelegten Orts-IDs. Wo diese Identifikatoren fehlen, müssen eigene Festlegungen getroffen werden, damit systemübergreifend ein gleiches Verständnis von Orten vorliegt, so dies notwendig ist. Dabei kann z. B. die Ortsliste aus dem Bestand der DELFI-Meta-Daten verwendet werden.

Für den Betrieb von TRIAS-Schnittstellen empfiehlt sich die Verwendung eines Ortsschlüssels, der sich vom Aufbau her an die IFOPT-Norm für Haltestellen anlehnt:

Aufbau eines Ortschlüssels
Länderkürzel:AGS:Ort

Beispiel	
Ilmenau	de:16070029:1

Client-Systeme, die selbst keine eigene Datenversorgung haben, können die Objektreferenzen für Gemeinden und Orte mit Hilfe des TRIAS-Schnittstellendiensts Ortsauflösung von der EKAP beziehen.

5.3. Adressen und POIs

Für den Betrieb von TRIAS-Schnittstellen ist es nicht notwendig, dass Adressen und wichtige Punkte (Points Of Interest, POI) systemübergreifend referenziert werden können. Es genügt, deren Lage durch Koordinatenpositionen mitzuteilen.

Die Kategorisierung von POI basiert auf dem Tagging-Schema von OpenStreetMap[5]. Ein POI kann mehrere Schlüssel-Wert-Paare zugeordnet bekommen (z.B. für eine Fahrrad-Ladestation amenity=charging_station und bike=yes).

---

[4] Siehe auch: Statistisches Bundesamt, https://www.destatis.de/DE/ZahlenFakten/LaenderRegionen/Regionales/Gemeindeverzeichnis/Gemeindeverzeichnis.html

[5] http://wiki.openstreetmap.org/wiki/DE:Map_Features

5.4. Verkehrsunternehmen und Verkehrsverbünde

Zur eindeutigen Referenzierung von Verkehrsunternehmen und Verkehrsverbünden wird ein Organisationscode verwendet. Damit diese Codes über mehrere Systeme hinweg eindeutig bleiben, empfiehlt sich der Aufbau einer übergreifenden (im besten Falle nationalen) Datenbank von Verkehrsunternehmen.

Aufbau eines Organisationscodes
Länderkürzel:Organisationscode

Beispiele	
Verkehrs- und Tarifverbund Stuttgart	de:vvs
Stuttgarter Straßenbahn AG	de:ssb
Fernverkehr Deutsche Bahn	de:dbag
DB Regio Baden-Württemberg	de:dbregiobw

5.5. Linien und Linienrichtungen

Zur eindeutigen Referenzierung von Linien wird der Linienschlüssel des verantwortlichen Datenlieferanten verwendet. Als verantwortlicher Datenlieferant kommt das beauftragte Verkehrsunternehmen (Konzessionär) oder der zuständige Verkehrsverbund in Frage.[6] Damit diese Codes über mehrere Datenlieferanten hinweg eindeutig bleiben, wird der Organisationscode (vgl. 5.4) als Namensraum vorangestellt.

Aufbau eines Linienschlüssels
Länderkürzel:Organisationscode:Linienschlüssel

Beispiel	
Stadtbahn-Linie U1 in Stuttgart	de:vvs:20001

Zur eindeutigen Referenzierung von Linienrichtungen wird der Richtungscode des verantwortlichen Datenlieferanten verwendet. Der Richtungscode ist vom Datenlieferanten frei wählbar und wird für den Fahrgast erst durch begleitende Texte verständlich. Der Richtungscode wird nur im Kontext einer Linie verwendet, so dass das Voranstellen des Linienschlüssels als Namensraum nicht notwendig ist.

---

[6] Damit in Datensammelsystemen Dopplungen von Datenlieferungen zu einer Linie von mehreren Datenlieferanten vermieden werden können, empfiehlt sich der Aufbau einer übergreifenden (im besten Falle nationalen) Datenbank von Linien.

Aufbau eines Richtungscodes	
Richtungscode	
Beispiele	
Hin	H
Rück	R
Hin	1
Rück	2
Stadteinwärts	E
Stadtauswärts	A

5.6. Fahrten

Zur eindeutigen Referenzierung von Fahrten (engl. *Vehicle journey* oder kurz: *Journey*) wird der Fahrtenschlüssel des verantwortlichen Datenlieferanten verwendet. Der Fahrtenschlüssel ist vom Datenlieferanten frei wählbar, solange er im Namensraum einer Linie eindeutig ist.

Aufbau eines Fahrtschlüssels	
Länderkürzel:Organisationscode:Linienschlüssel:Fahrtenschlüssel	
Beispiel	
Fahrt 1512 der Linie U1 in Stuttgart	de:vvs:20001:1512

Falls eine Organisation (Verkehrsunternehmen) ihre Fahrten nicht in Linien organisiert (z. B. Bahnfernverkehr), kann der Linienschlüssel leer bleiben.

Beispiel	
ICE 612 der DB AG	de:dbag::612

5.7. Fahrzeuge

Zur eindeutigen Referenzierung von Fahrzeugen (engl. *Vehicle*) wird der Fahrzeugcode des verantwortlichen Datenlieferanten verwendet.

In Echtzeitschnittstellen (VDV 454, SIRI ET) teilen Leitstellen zu einer Fahrplanfahrt den Fahrzeugcode mit, so dass eine EKAP für einen bestimmten Betriebstag wissen kann, welche Fahrt von welchem Fahrzeug durchgeführt wird. Für jeden Betriebstag muss daher die Zuordnung von Fahrzeugcode zu Fahrtenschlüssel eineindeutig sein.

Aufbau eines Fahrzeugcodes
Länderkürzel:Organisationscode:Fahrzeugcode

Beispiel	
Fahrzeug 5812 der SSB AG	de:ssb:5812

5.8. Fahrzeugtypen

Der Fahrzeugtyp (engl. *Vehicle type*) wird nur vom Fahrzeug an die mobile App mitgeteilt (in Form von Code und menschenlesbaren Text). Der Fahrzeugtypcode wird nicht in Folgeaufrufen verwendet und wird daher im Rahmen von TRIAS nicht weiter betrachtet.

5.9. Verkehrstage

Eine Fahrplanfahrt wird erst in Verbindung mit einem Verkehrstag (engl. *Operating Day*) zu einer spezifischen Fahrt. Ein Verkehrstag kann auch Uhrzeiten nach Mitternacht einschließen und daher von einem Kalendertag abweichen. Ob eine solche Abweichung existiert und wie groß sie ist, ist für die Fahrgastinformation nicht relevant. Den Fahrgästen gegenüber werden nur Uhrzeiten und Datumsangaben nach dem Kalendertagsprinzip bekannt gegeben.

Ein Verkehrstag ist in TRIAS die Referenz auf den Betriebstag-Code der Fahrplandaten.

Diese Betriebstag-Codes sollten in TRIAS nach der Norm ISO 8601 dargestellt werden.

Beispiel	
29. März 2013	2013-03-29

5.10. Eigentümer

Mit dem Begriff Eigentümer (engl. *Owner*) sind hier die Betreiber von Haltestelleneinrichtungen und Fahrgastinformationsgeräten gemeint. In der Regel sind das Verkehrsunternehmen, aber auch z. B. Kinobetreiber können einen Monitor für die Anzeige von aktuellen Haltestellenabfahrten aufstellen und betreiben. Die Referenzierung von Eigentümern erfolgt auf genau dieselbe Weise wie die von Verkehrsunternehmen und - verbünden (vgl. 5.4).

5.11. Haltestellen- und Fahrzeugeinrichtungen

Haltestellen- und Fahrzeugeinrichtungen (wie z. B. Aufzüge oder Fahrscheinautomaten) werden durch Codes referenziert, die vom Eigentümer (vgl. 5.10) vergeben werden. Im Kontext eines Eigentümers ist der Code einer Einrichtung also global eindeutig.

5.12. Teilnehmende Systeme / IT-Systeme

Die TRIAS-Dienste werden von IT-Systemen angeboten und in Anspruch genommen. Sie sind die teilnehmenden Systeme (engl. *Participants*) an einem umfassenden Systemverbund zur Steuerung des Betriebs des ÖV und zur Fahrgastinformation. Damit diese Systeme unterscheidbar und ansprechbar sind, benötigen sie Kennungen (in VDV 453/454 als *Leitstellenkennung* bekannt).

Aufbau einer Systemkennung
Länderkürzel:Organisationscode:Systemkennung

Beispiel	
Öffentliche EKAP des VVS	de:vvs:publicEKAP

5.13. Ereignismeldungen

Ereignis- und Störungsmeldungen (engl. *Situations*) werden mit Hilfe der in SIRI SX definierten Strukturen übertragen. Dort ist auch die Vergabe von IDs für die Ereignismeldungen geregelt. Die Meldungs-IDs werden im Kontext des teilnehmenden Systems (vgl. 5.12) übertragen und sind somit global eindeutig.

5.14. Tarifverantwortliche

Eine Organisation, die verantwortlich ist für die Festlegung von Tarifstrukturen und die Entwicklung von Fahrscheinprodukten, wird als Tarifverantwortlicher (engl. *Fares authority*) bezeichnet. Für Verbundtarife sind dies meist die Verkehrs- und Tarifverbünde, für Haustarife die Verkehrsunternehmen selbst. Die Referenzierung von Tarifverantwortlichen erfolgt auf genau dieselbe Weise wie die von Verkehrsunternehmen und - verbünden (vgl. 5.4).

5.15. Tarifzonen

Die Codierung von Tarifzonen (engl. *Fare zones*) liegt in der Obhut der jeweiligen Tarifverantwortlichen (vgl. 5.14). Tarifzonen werden im Kontext der jeweiligen Tarifverantwortlichen angegeben und werden so global eindeutig.

5.16. Fahrscheine und Vielfahrerkarten

Die Codierung von Fahrscheinen (engl. *Ticket*) liegt in der Obhut der jeweiligen Tarifverantwortlichen (vgl. 5.14). IDs von Fahrscheinen werden im Namensraum der jeweiligen Tarifverantwortlichen angegeben und sind so global eindeutig.

Aufbau eines Codes für einen Fahrschein
Länderkürzel:Organisationscode:FahrscheinCode

Beispiel	
Einzel-Ticket für Erwachsene im VVS-Gebiet für 2 Zonen	de:vvs:EinzelErw2Z

Die Codierung von Vielfahrerkarten (engl. *TravellerCard*), z. B. BahnCard50 der Deutschen Bahn AG, liegt in der Obhut der jeweiligen Tarifverantwortlichen (vgl. 5.14). Der Code einer Vielfahrerkarte muss im Namensraum des Tarifverantwortlichens angegeben werden.

Aufbau eines Codes für eine Vielfahrerkarte
Länderkürzel:Organisationscode:TravellerCardCode

Beispiel	
BahnCard50 der DB AG	de:dbag:BC50

6. Dienste und XML-Schemata

In diesem Dokument werden Schnittstellendefinitionen für Dienste zwischen Softwarekomponenten dargestellt. Für eine ausführliche Erläuterung der Aufgabenstellung dieser Dienste und der möglichen Systemarchitekturen sei hier auf die grundlegenden VDV-Schriften (VDV-Schrift 431-1) und (VDV-Schrift 430) verwiesen.

Die TRIAS-Schnittstellendienste sind als XML-Schemata definiert. Eine Übersicht über die Dienste und ihre Implementierung als XML-Schema bietet der erste Abschnitt dieses Kapitels. Einige Strukturdefinitionen sind in mehreren Diensten nützlich und werden daher in eigenen Schemadateien als gemeinsame Basis hierarchisch definiert, so dass eine Wiederverwendbarkeit ermöglicht wird. Das dabei verfolgte Konzept orientiert sich stark an den Grundsätzen der Objektorientierung. Die gemeinsam genutzten Strukturdefinitionen sind im zweiten Abschnitt beschrieben. Der dritte Abschnitt stellt die XML-Schemata vor, die aus SIRI importiert werden. Eine Klassifikation der Fehlerzustände findet sich im vierten Abschnitt.

6.1. Bereitgestellte Dienste

Die TRIAS-Schnittstellenfamilie umfasst derzeit folgende Dienste:

Dienst	Bezeichnung des Anfrageelements	Schema-Datei	Kapitel
Ortsinformation	LocationInformationRequest	Trias_Locations.xsd	8
Verbindungsauskunft	TripRequest	Trias_Trips.xsd	9
Abfahrtstafeln	StopEventRequest	Trias_StopEvents.xsd	10
Logische Ortung	PositioningRequest	Trias_Positioning.xsd	11
Fahrtinformation (EKAP)	TripInfoRequest	Trias_TripInfo.xsd	12
Anschlussmeldung	ConnectionDemandRequest	Trias_Connections.xsd	13.1.1
Anschlussstatus	ConnectionStatusRequest	Trias_Connections.xsd	13.1.2
Info bei Anschlussverlust	ConnectionStatus-Response	Trias_Connections.xsd	13.1.3
Anschluss-rückmeldung	ConnectionReportRequest	Trias_Connections.xsd	13.1.4
Fahrpreis- und Tarifberechnung	FaresRequest	Trias_Fares.xsd	14
Buchungsinformation	BookingInfoRequest	Trias_Booking.xsd	15

Dienst	Bezeichnung des Anfrageelements	Schema-Datei	Kapitel
IV-Routing	IndividualRouteRequest	Trias_-IndividualTrips.xsd	16
Kartendienst	MapServiceRequest ImageCoordinatesRe-quest GeoCoordinatesRequ-est	Trias_Maps.xsd	17
Schadensmeldung	FacilityStatusReport	Trias_Facilities.xsd	18.4
Zustand von Einrich-tungen	FacilityRequest	Trias_Facilities.xsd	18.6
Benachrichtigungs-dienst	SubscriptionRequest	Trias.xsd	7.1.2 und 19
Personalisierungs-dienst	PersonalisationRequest	Trias_-Personalisation.xsd	20
Fahrzeug-informationen	VehicleDataRequest	Trias_-VehicleInterface.xsd	21
Haltewunsch	StopRequestRequest	Trias_-VehicleInterface.xsd	22
Diensteregister	ServiceRegisterRequest	Trias_-ServiceRegister.xsd	23
Authentifizierung	AbstractTriasService-Request (vererbt auf alle TRIAS-Nachrichten)	Trias_-RequestSupport.xsd	7.9 und 24

Tabelle 2: Liste der TRIAS-Dienste und ihrer Anfrageelemente.

6.2. Dienstübergreifend genutzte XML-Schemata

Um Strukturen, die in mehr als einem Dienst verwendet werden, nicht mehrfach und damit redundant definieren zu müssen, werden gemeinsam benutzte Basis-XML-Schemata eingeführt, die sich hierarchisch inkludieren. Die Inklusionsreihenfolge und der Zuschnitt der Schemadateien sind dabei so gewählt, dass inhaltlich verwandte Elemente in einer Datei zusammenstehen und dass jedes Schema möglichst nur so viel inkludiert, wie für die eigenen Aufgaben notwendig ist.

Die gemeinsam genutzten Basis-Schemadateien werden ausführlich in Kapitel 7 erläutert. Die hierarchische Abhängigkeit zwischen den einzelnen Schemata zeigen die folgenden Abbildungen. Die blau eingefärbten Schemadateien werden von SIRI importiert.

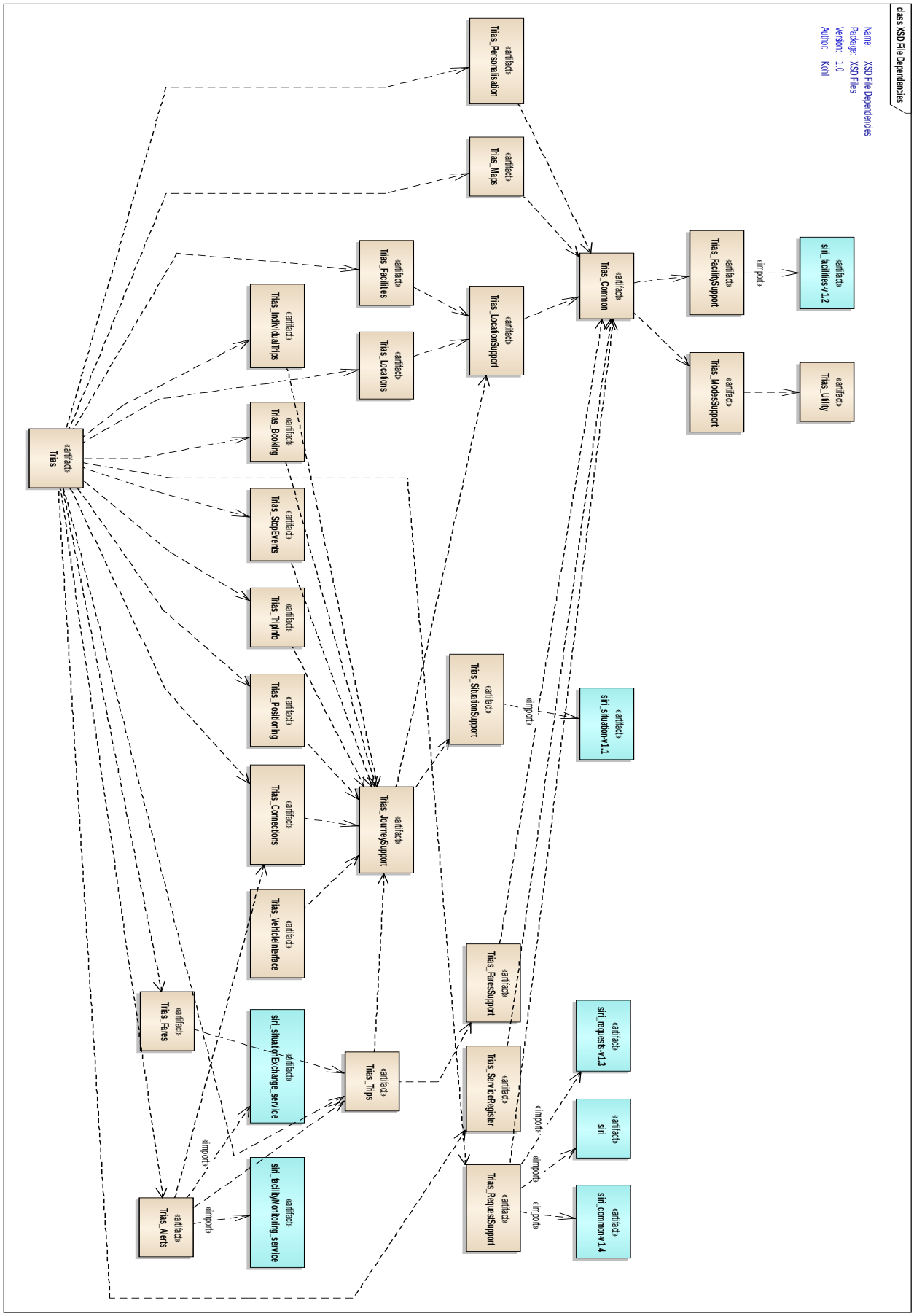

Abbildung 4: Gesamtübersicht über die Hierarchie der Schemadateien.

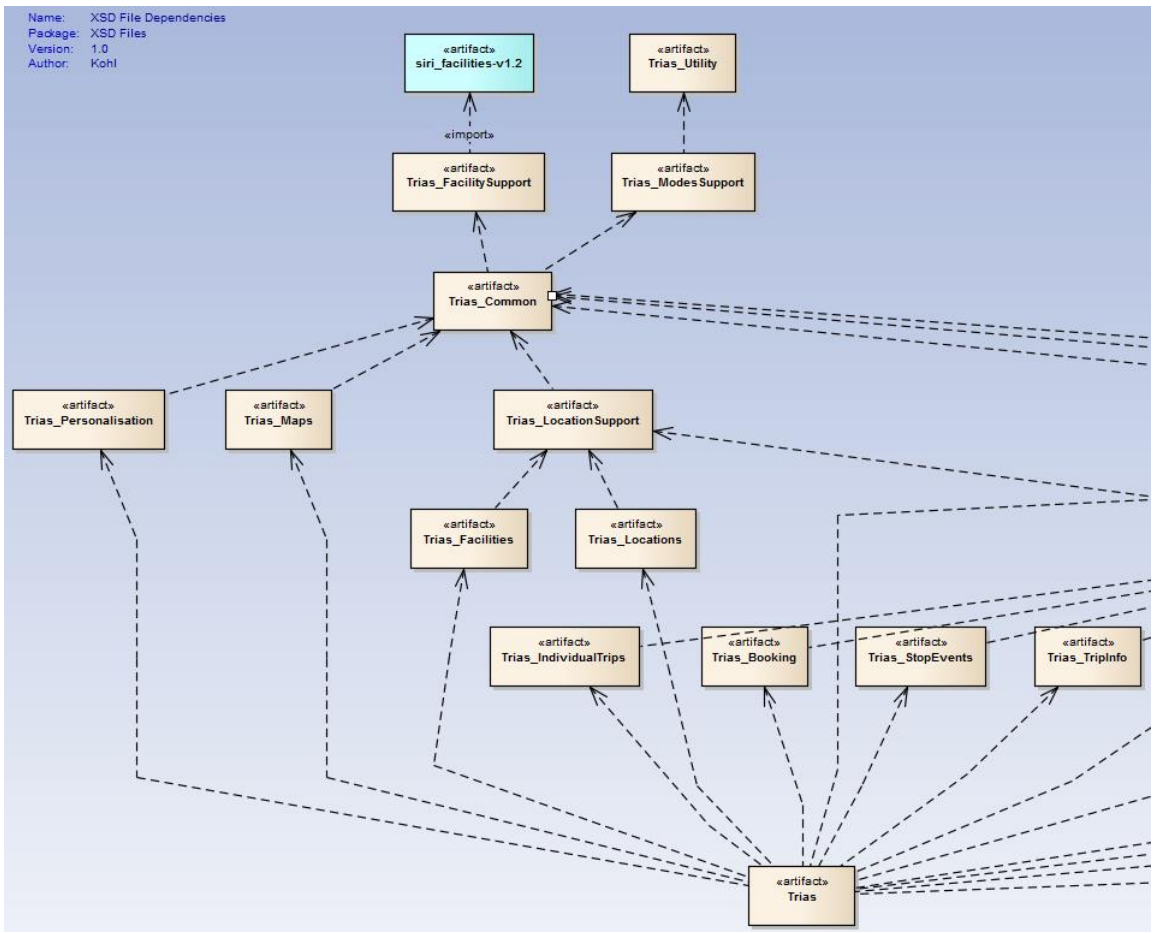

Abbildung 5: Linker Teil der Schema-Hierarchie.

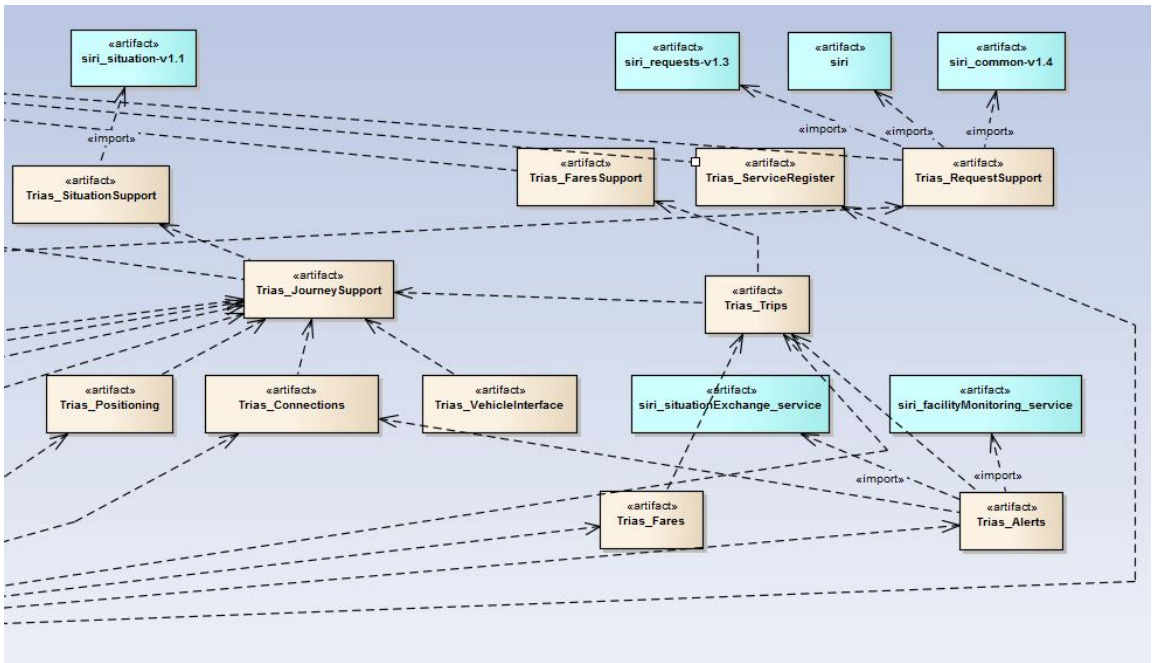

Abbildung 6: Rechter Teil der Schema-Hierarchie.

6.3. Importierte Schemata

Aus den SIRI-Schnittstellen werden einige Schemadateien nach TRIAS importiert. Diese Dateien sind in den Abbildungen des vorigen Abschnitts blau eingefärbt dargestellt. Durch diesen Import von SIRI-Definitionen wird erreicht, dass die SIRI-Verfahren für den Austausch von Nachrichten auch für die TRIAS-Meldungen anwendbar sind. Außerdem können bestimmte Strukturdefinitionen aus SIRI wiederverwendet werden, was die Konsistenz zwischen diesen Schnittstellenstandards sicherstellt. Dies betrifft unter anderem die Definition von Verkehrsmittelarten (modes), Störungsereignissen (situations) und Haltestelleneinrichtungen bzw. Fahrzeugausstattungen (facilities).

SIRI wird in der derzeit aktuellen Version 1.4 importiert.

6.4. Fehlerzustände beim Betrieb von TRIAS-Diensten

Die Fehlerzustände beim Betrieb von TRIAS-Diensten werden durch Fehlercodes signalisiert, die in der Struktur *ErrorMessage* übermittelt werden können. *ErrorMessage* kann an den meisten Stellen mehrfach auftreten und daher auch eine mehrfache, vielschichtige Fehlersituation beschreiben. In ErrorMessage können Fehlercodes auftreten, die

- aus den SIRI-Diensten geerbt werden,
- allgemeine, dienstübergreifende TRIAS-Fehlersituationen beschreiben oder
- dienstspezifische Fehlersituationen anzeigen.

Die TRIAS-Fehlercodes sind durch ein Präfix gekennzeichnet, das den jeweiligen Dienst angibt (z. B. ***STOPEVENT***_) oder anzeigt, dass es sich um einen allgemeinen Fehlerzustand handelt (***TRIASGENERIC***_).

6.4.1. Fehlercodes aus SIRI

In SIRI (CEN, TS 15531, Part 2), Kapitel 5.7, werden eine Reihe von Fehlercodes definiert, die für das Nachrichtenübermittlungsverfahren eine wichtige Rolle spielen. Diese Codes sind in die Gruppen Erfolg (Success), Systemfehler (Systemic Error) und Anwendungsfehler (Application Error) eingeteilt (vgl. Tabelle 3).

Group	Condition	Description (Beschreibung)
Success	OK (true)	Request successful. (Anfrage erfolgreich bearbeitet.)
Systemic Error	***RequestTimeout***	Server not responding. (Server antwortet nicht.)
	InvalidRequest	The server does not "understand" the request. The client should not repeat the request. (Der Server “versteht” die Anfrage nicht. Der Client braucht die Anfrage nicht zu wiederholen.)
	Unauthorized	User name and password are required for the request, or credentials not satisfied. (Benutzername und Passwort sind für die Anfrage erforderlich, oder die Berechtigungen reichen nicht aus.)
	Forbidden	The server "understands" the request, but cannot carry it out. (Der Server “versteht” die Anfrage, kann sie aber nicht ausführen.)
	NotFound	The requested URL was not found. (Die angefragte URL konnte nicht gefunden warden.)
Application Error	***VersionNotSupported***	Service is not available. (Die angefragte Version des Dienstes ist nicht verfügbar.)
	CapabilityNotSupported	Service does not support the requested capability. (Die angeforderte Funktionalität wird vom Dienst nicht unterstützt.)
	ServiceNotAvailable	Functional service is not available to use (but it is still capable of giving this response). (Der funktionale Dienst kann keine Anfragen abarbeiten (obwohl er in der Lage ist, eine Antwort zu geben).)
	AccessNotAllowed	Requestor is not authorised to the service or data requested. (Der Anfrager ist für den Zugriff auf den Dienst oder die Daten nicht autorisiert.)
	NoInfoForTopic	Valid request was made but service does not hold any data for the requested topic expression. (Die Anfrage ist gültig, der Dienst kann aber über den angrefragten Fachinhalt keine Auskunft geben.)
	UnknownSubscriber	Subscriber not found. (Der Abonnent wurde nicht gefunden.)
	UnknownSubscription	Subscription not found. (Das Abonnement wurde nicht gefunden.)
	AllowedResource-UsageExceeded	Valid request was made, but request would exceed the permitted resource usage of the client. (Die Anfrage ist gültig, sie überschreitet aber das dem Client zugestandene Ressourcen-Limit.)
	OtherError	Other Error Type. (Sonstiger Fehler.)

Tabelle 3: Liste der Fehlercodes, wie sie in SIRI für das Nachrichtenübermittlungsverfahren definiert werden.

6.4.2. Allgemeine TRIAS-Fehlerzustände

In *ErrorMessage* können folgende allgemeine Fehlerzustände auftreten:

AUTH_FAILURE	Dieser Fehler tritt auf, wenn eine Anfrage mit ungültiger oder nicht prüfbarer Signatur empfangen wurde.
AUTH_MISSING	Dieser Fehlercode tritt auf, wenn der Server zwingend eine Authentifizierung benötigt, aber eine Nachricht ohne Signatur empfangen wurde.
AUTH_USER_UNKNOWN	Dieser Fehlercode wird zurückgegeben, wenn die Authentifikation fehlschlägt, weil der Benutzer unbekannt ist.
TRIASGENERIC_ERROR	Bei der Verarbeitung der Anfrage ist ein Fehler aufgetreten, der nicht durch einen speziellen Fehlercode abgedeckt wird, Einzelheiten werden im Text der Fehlermeldung genannt.
TRIASGENERIC_SERVICENOTSUPPORTED	In der Anfrage wurde ein Dienst spezifiziert, der vom Server nicht unterstützt wird (z.B. Dienst ConnectionDemand).
TRIASGENERIC_REQUESTNOTSUPPORTED	Es wurde eine Anfrage spezifiziert, die vom Server nicht unterstützt wird (z.B. Anfrage FacilityStatusReport).
TRIASGENERIC_FEATURENOTSUPPORTED	In der Anfrage wurde ein Feature spezifiziert, das vom Server nicht unterstützt wird (z.B. Parameter NotVia in TripRequest)
TRIASGENE-RIC_LANGUAGENOTSUPPORTED	In der Anfrage wurde eine Sprache für die Anzeige der Ergebnistexte spezifiziert, die vom Server nicht unterstützt wird (zumindest im Kontext der vorliegenden Anfrage).
TRIASGENE-RIC_EXCEPTIONFROMREQUESTEDLANGU AGE	In der Anfrage wurde eine Sprache für die Anzeige der Ergebnistexte spezifiziert, die vom Server nicht bei allen Textelementen der Antwort unterstützt wird.
TRIASGENE-RIC_DATAVERSIONNOTAVAILABLE	Die in der Anfrage angeforderte Datenversion konnte vom Server nicht berücksichtigt werden.

Tabelle 4: Generische TRIAS-Fehlermeldungen, die in allen Nachrichten auftreten können.

7. Gemeinsam genutzte XML-Strukturen

In diesem Kapitel werden die XML-Strukturen erläutert, die als Basisobjekte in den dienstübergreifend genutzten XML-Schemadateien definiert werden. Die Gliederung ergibt sich anhand der einzelnen Schemadateien.

7.1. Trias, das Wurzelelement

TRIAS steht für *Travellers' Realtime Information and Advisory Standard*.

In der XML-Schema-Definition Trias.xsd wird das allgemeine Wurzelelement *TRIAS* definiert, das als gemeinsame Basis für alle Nachrichten aller TRIAS-Dienste dient.

An komplexen Strukturen finden sich in Trias.xsd folgende:

7.1.1. ServiceRequestStructure

ServiceRequestStructure			*+Structure*	**Grundstruktur für jede TRIAS-Anfrage (ohne Abonnement).**
ServiceRequestContext	***:::***	**1:1**	*AbstractTriasServiceRequest*	Gemeinsamer Anfragekontext (vgl. 7.9.2).
	RequestPayload	**1:1**	*RequestPayload*	Dienstspezifischer Anfrageinhalt (vgl. 7.1.3).

Tabelle 5: Beschreibung der Struktur ***ServiceRequestStructure***.

7.1.2. SubscriptionRequestStructure

SubscriptionRequestStructure				*+Structure*	**Grundstruktur für jede TRIAS-Abonnement-Anfrage.**
SubscriptionRequestContext		***:::***	**1:1**	*AbstractTriasSubscriptionRequest*	Gemeinsamer Anfragekontext (vgl. 7.9.3).
AlertSettingsGroup		*AlertTimeWindow*	0:*	*WeekdayTimePeriod*	Zeitfenster, in denen Benachrichtigungen zugestellt werden dürfen (vgl. 7.4.9).
		MaximumAlertFrequency	0:1	*xs:duration*	Maximale Frequenz für Benachrichtigungen aus gleichem Grund.
		MaximumTimeBeforeEvent	0:1	*xs:duration*	Früheste Zeit für eine Benachrichtigung gemessen vom Beginn des Ereignisses. Nur im Zusammenhang mit Ereignissen, deren Beginn im Voraus bekannt ist.
SubscriptionRequest	***a***	***SituationExchangeSubscriptionRequest***	**-1:1**	*SituationExchangeSubscriptionRequest*	Inhalt der Abonnement-Anfrage für allgemeine Ereignis- und Störungsbenachrichtigung (vgl. Kapitel 19).
	b	***FacilityMonitoringSubscriptionRequest***		*FacilityMonitoringSubscriptionRequest*	Inhalt der Abonnement-Anfrage für Zustandsmeldungen zur Infrastruktur von Haltestellen und Fahrzeugen (vgl. Kapitel 19).
	c	***TripMonitoringSubscriptionRequest***		*TripMonitoringSubscriptionRequest*	Inhalt der Abonnement-Anfrage für Benachrichtigungen zu einer bestimmten Reiseverbindung (vgl. 19.3.1).

Tabelle 6: Beschreibung der Struktur ***SubscriptionRequestStructure***.

7.1.3. RequestPayloadStructure

RequestPayloadStructure				*+Structure*	**Element zur Auswahl des gewünschten TRIAS-Dienstes.**
	a	***BookingInfoRequest***		*BookingInfoRequest*	Anfrage nach Buchungsinformationen (vgl. 15.2.1).
	b	***ConnectionDemandRequest***		*ConnectionDemandRequest*	Anfrage zur Anschlussvoranmeldung (vgl. 13.4.1).
	c	***ConnectionDemandDeleteRequest***		*ConnectionDemandDeleteRequest*	Löschung einer Anschlussvoranmeldung (vgl. 13.4.2).
	d	***ConnectionReportRequest***		*ConnectionReportRequest*	Meldung des Fahrgasts, ob ein Anschluss geklappt hat (vgl. 13.8.1).
	e	***ConnectionStatusRequest***		*ConnectionStatusRequest*	Anfrage zum Anschlussstatus (vgl. 13.6.1).
	f	***FacilityRequest***		*FacilityRequest*	Anfrage zu Fahrzeug- und Infrastruktureinrichtungen (vgl. 18.6.1).
	g	***FacilityStatusReport***		*FacilityStatusReport*	Übermittlung des Zustands von Fahrzeug- und Infrastruktureinrichtungen in einem aktiven Abonnement (vgl. 18.4.1).
	h	***FaresRequest***		*FaresRequest*	Anfrage zum Tarifberechnungsdienst (vgl. 14.2.1).
	i	***GeoCoordinatesRequest***		*GeoCoordinatesRequest*	Anfrage nach Geo-Koordinaten (vgl. 17.2.3).
	j	***ImageCoordinatesRequest***		*ImageCoordinatesRequest*	Anfrage nach Bildkoordinaten (vgl. 17.2.2).
	k	***IndividualRouteRequest***		*IndividualRouteRequest*	Anfrage nach einer IV-Route (vgl. 16.2.1).
	l	***LocationInformationRequest***		*LocationInformationRequest*	Anfrage zum Ortsinformationsdienst (vgl. 8.3.1).
	m	***MapServiceRequest***		*MapServiceRequest*	Anfrage zum Kartendienst (vgl. 17.2.1).
	n	***PersonalisationRequest***		*PersonalisationRequest*	Anfrage zum Personalisierungsdienst (vgl. 20.4.1).
	o	***PositioningRequest***		*PositioningRequest*	Anfrage zum Positionierungsdienst (vgl. 11.2.1).
	p	***ServiceRegisterRequest***		*ServiceRegisterRequest*	Anfrage zum Dienstregisterdienst (vgl. 23.3.1).
	q	***StopEventRequest***		*StopEventRequest*	Anfrage nach Abfahrtstafeln (vgl.10.2.1).
	r	***StopRequestRequest***		*StopRequestRequest*	Bekanntgabe eines Haltewunsches im Fahrzeug (vgl. 22.2.1).
	s	***TripInfoRequest***		*TripInfoRequest*	Anfrage nach Fahrtinformationen (vgl. 12.2.1).
	t	***TripRequest***		*TripRequest*	Anfrage nach einer intermodalen Verbindungsberechnung (vgl. 9.2.1).
	u	***VehicleDataRequest***		*VehicleDataRequest*	Anfrage nach Fahrzeuginformationen (vgl. 21.2.1).

Tabelle 7: Beschreibung der Struktur ***RequestPayloadStructure***.

7.1.4. ServiceDeliveryStructure

ServiceDeliveryStructure			*+Structure*	**Grundstruktur für jede TRIAS-spezifische Antwort.**
	:::	**1:1**	*AbstractTriasResponse*	Gemeinsamer Antwortkontext (vgl. 7.9.4).
	DeliveryPayload	**1:1**	*DeliveryPayload*	Dienstspezifischer Antwortinhalt (vgl. 7.1.5).

Tabelle 8: Beschreibung der Struktur ***ServiceDeliveryStructure***.

7.1.5. DeliveryPayloadStructure

DeliveryPayloadStructure			*+Structure*	**Element zur Auswahl der passenden TRIAS-Antwort.**
	a	***BookingInfoResponse***	*BookingInfoResponse*	Antwort mit Buchungsinformationen (vgl. 15.3.1).
	b	***ConnectionDemandResponse***	*ConnectionDemandResponse*	Antwort zur Anschlussvoranmeldung (vgl. 13.5.1).
	c	***ConnectionDemandDeleteResponse***	*ConnectionDemandDeleteResponse*	Bestätigung zur Löschung einer Anschlussvoranmeldung (vgl. 13.5.2).
	d	***ConnectionReportResponse***	*ConnectionReportResponse*	Bestätigung zur Anschlusserreichungsmeldung (vgl. 13.9.1).
	e	***ConnectionStatusNotification***	*ConnectionStatusNotification*	Übermittlung des Anschlussstatus in einem aktiven Abonnement (vgl. 13.6.2).
	f	***ConnectionStatusResponse***	*ConnectionStatusResponse*	Antwort zu Anschlussstatusanfrage (vgl. 13.7.1).
	g	***FacilityMonitoringDelivery***	*siri:FacilityMonitoringDelivery*	Aktualisierungsnachricht im Rahmen eines Abonnemts zu Ausstattungsmerkmalen und Einrichtungen (vgl. Kapitel 19).
	h	***FacilityResponse***	*FacilityResponse*	Antwort zu Fahrzeug- und Infrastruktureinrichtungen (vgl. 18.7.1).
	i	***FacilityStatusReportResponse***	*FacilityStatusReportResponse*	Bestätigung zur Übermittlung des Zustands von Fahrzeug- und Infrastruktureinrichtungen (Schadensmeldung, vgl. 18.5.1).
	j	***FaresResponse***	*FaresResponse*	Antwort zur Tarifberechnungsanfrage (vgl. 14.3.1).
	k	***GeoCoordinatesResponse***	*GeoCoordinatesResponse*	Antwort zur Anfrage nach Geo-Koordinaten (vgl. 17.3.3).
	l	***ImageCoordinatesResponse***	*ImageCoordinatesResponse*	Antwort zur Anfrage nach Bildkoordinaten (vgl. 17.3.2).
	m	***IndividualRouteResponse***	*IndividualRouteResponse*	Antwort mit berechneten IV-Routen (vgl. 16.3.1).
	n	***LocationInformationResponse***	*LocationInformationResponse*	Antwort des Ortsinformationsdiensts (vgl. 8.4.1).
	o	***MapServiceResponse***	*MapServiceResponse*	Antwort des Kartendiensts (vgl. 17.3.1).
	p	***PersonalisationResponse***	*PersonalisationResponse*	Antwort des Personalisierungsdiensts (vgl. 20.5.1).
	q	***PositioningResponse***	*PositioningResponse*	Antwort des Positionierungsdiensts (vgl. 11.3.1).

	r	***ServiceRegisterResponse***		*ServiceRegis-terResponse*	Antwort des Dienstregisterdiensts (vgl. 23.4.1).
	s	***SituationExchangeDe-livery***		*si-ri:SituationExchangeDelivery*	Aktualisierungsnachricht im Rahmen eines Abonnemts zu Störungsinformationen (vgl. Kapitel 19).
	t	***StopEventResponse***		*StopE-ventResponse*	Antwort mit Haltestellen-Abfahrtstafeln (vgl.10.2.1).
	u	***StopRequestResponse***		*StopRe-questRespon-se*	Bestätigung zum Haltewunsch im Fahrzeug (vgl. 22.3.1).
	v	***TripInfoResponse***		*TripInfo-Response*	Antwort mit Fahrtinformationen (vgl. 12.3.1).
	w	***TripMonitoringDelivery***		*TripMonito-ringDelivery*	Aktualisierungsnachricht im Rahmen eines Abonnemts zu Verbindungszuständen (vgl. 19.4.1).
	x	***TripResponse***		*TripResponse*	Antwort zur intermodalen Verbindungsberechnung (vgl. 9.3.1).
	y	***VehicleDataResponse***		*VehicleData-Response*	Antwort mit Fahrzeuginformationen (vgl. 21.3.1).

Tabelle 9: Beschreibung der Struktur ***DeliveryPayloadStructure***.

7.2. Trias_Utility

In der XML-Schema-Definition Trias_Utility.xsd werden eine Reihe von Typen und Strukturen definiert, die als Basistypen in anderen Definitionen wieder verwendet werden. Die Definitionen in Trias_Utility haben keinen unmittelbaren Bezug zur Fachthematik des ÖV.

7.2.1. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

PercentType	*xs:nonNegativeInteger*	Prozentangabe als Ganzzahlwert. Maximalwert ist 100.
OpenPercentType	*xs:nonNegativeInteger*	Prozentangabe als Ganzzahlwert, nach oben unbeschränkt.
BitStringType	*xs:string*	Zeichenkette, die nur aus Nullen und Einsen bestehen kann.
DistanceType	*xs:nonNegativeInteger*	Typ zur Angabe von Distanzen (in Metern).
LengthType	*xs:nonNegativeInteger*	Typ zur Angabe von Längen (in Metern).
SpeedType	*xs:nonNegativeInteger*	Typ zur Angabe einer Geschwindigkeit (in Meter pro Sekunde).
PriorityType	*xs:nonNegativeInteger, [1,5]*	Prioritätswerte von 1 (höchste Priorität) bis 5 (niedrigste Priorität).
LongitudeType	*xs:decimal*	Geographische Länge.
LatitudeType	*xs:decimal*	Geographische Breite.
AltitudeType	*xs:decimal*	Höhe über dem Meeresspiegel in Meter.
AbsoluteBearingType	*xs:nonNegativeInteger*	Kompassrichtung in Grad. Nord = 0 Grad, im Uhrzeigersinn aufsteigende Werte.
PhoneNumberType	*xs:normalizedString*	Typ zur Angabe einer Telefonnummer.

Tabelle 10: Liste der einfachen Typdefinitionen in Trias_Utility.xsd.

7.2.2. InternationalTextStructure

An komplexen Strukturen findet sich in Trias_Utility Folgende:

InternationalTextStructure			*+Structure*	**Ein Text mit einer Text-ID und Angabe der Sprache, in der er verfasst ist.**
	Text	**1:1**	*xs:normalized String*	Text.
	TextId	0:1	*xs:NMTOKEN*	ID des Texts.
	Language	0:1	*xs:language*	Sprache, in der der Text verfasst ist.

Tabelle 11: Beschreibung der Struktur ***InternationalTextStructure***.

Elemente vom Typ *InternationalText* werden verwendet, um Texte in unterschiedlichen Sprachen angeben zu können.

7.2.3. GeoPositionStructure

GeoPositionStructure			*+Structure*	**Geographische Position in WGS84.**
	Longitude	**1:1**	*Longitude*	Geographische Länge bzgl. des Greenwich-Meridians. Wertebereich von -180 Grad (West) bis +180 Grad (Ost).
	Latitude	**1:1**	*Latitude*	Geographische Breite bzgl. des Äquators. Wertebereich von -90 Grad (Süden) bis +90 Grad (Norden).
	Altitude	0:1	*Altitude*	Höhe über dem Meeresspiegel in Meter.

Tabelle 12: Beschreibung der Struktur ***GeoPositionStructure***.

7.2.4. WebLinkStructure

WebLinkStructure			*+Structure*	**URL mit Beschriftungstext zu einer Ressource im Web**
	Label	**1:1**	*International-Text*	Beschriftungstext des Links (s. 7.2.2).
	Url	**1:1**	*xs:anyURI*	URL zur Web-Ressource.

Tabelle 13: Beschreibung der Struktur ***WebLinkStructure***.

Um weitergehende Informationen zu einem Objekt abrufen zu können, werden in die TRIAS-Antworten an manchen Stellen Elemente des Typs *WebLinkStructure* eingefügt. Ein Client kann die darin enthaltenen URLs nutzen, um weitere Aktionen auszuführen oder Informationen abzurufen. Da die Clients von unterschiedlichem Typ (z.B. App oder Web-Browser) sein können, sind die Anbieter der Web-Ressourcen aufgerufen, eine möglichst breite Unterstützung unterschiedlicher Clienttypen anzubieten.

7.3. Trias_ModesSupport

In der XML-Schema-Definition Trias_ModesSupport.xsd werden eine Reihe von Basistypen und -strukturen definiert, die zur Klassifizierung von Verkehrsmitteln dienen. Diese Definitionen lehnen sich sehr stark an die TPEG-Codierung an, die auch in SIRI verwendet wird.

7.3.1. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

IndividualModesEnumeration	*walk* | *cycle* | *taxi* | *self-drive-car* | *others-drive-car* | *motorcycle* | *truck*	Klassifizierung der Individualverkehrsarten.
ContinuousModesEnumeration	*walk* | *demandResponsive* | *replacementService*	Klassifizierung von kontinuierlichen Verkehrsarten, die zu jeder beliebigen Zeit (ohne Fahrplan) stattfinden können.
InterchangeModesEnumeration	*walk* | *parkAndRide* | *bikeAndRide* | *carHire* | *bikeHire* | *protectedConnection* | *guaranteedConnection* | *remainInVehicle* | *changeWithinVehicle* | *checkIn* | *checkOut*	Klassifizierung von Umsteigevorgängen
PtModesEnumeration	*all* | *unknown* | *air* | *bus* | *trolleyBus* | *tram* | *coach* | *rail* | *intercityRail* | *urbanRail* | *metro* | *water* | *cableway* | *funicular* | *taxi*	Klassifizierung der ÖV-Verkehrsmittel (nach TPEG pti_table 01).
RailSubmodeEnumeration	*unknown* | *undefined* | *local* | *highSpeedRail* | *suburbanRailway* | *regionalRail* | *interregionalRail* | *longDistance* | *international* | *sleeperRailService* | *nightRail* | *carTransportRailService* | *touristRailway* | *railShuttle* | *replacementRailService* | *specialTrain* | *crossCountryRail* | *rackAndPinionRailway*	Unter-Klassifizierung der Züge (nach TPEG pti_table 02).
CoachSubmodeEnumeration	*unknown* | *undefined* | *internationalCoach* | *nationalCoach* | *shuttleCoach* | *regionalCoach* | *specialCoach* | *sightseeingCoach* | *touristCoach* | *commuterCoach*	Unter-Klassifizierung der Überlandbusse (nach TPEG pti_table 03).
MetroSubmodeEnumeration	*unknown* | *undefined* | *metro* | *tube* | *urbanRailway*	Unter-Klassifizierung der Untergrundbahnen (nach TPEG pti_table 04).
BusSubmodeEnumeration	*unknown* | *undefined* | *localBus* | *regionalBus* | *expressBus* | *nightBus* | *postBus* | *specialNeedsBus* | *mobilityBus* | *mobilityBusForRegisteredDisabled* | *sightseeingBus* | *shuttleBus* | *schoolBus* | *schoolAndPublicServiceBus* | *railReplacementBus* | *demandAndResponseBus* | *airportLinkBus*	Unter-Klassifizierung der Busse (nach TPEG pti_table 05).
TramSubmodeEnumeration	*unknown* | *undefined* | *cityTram* | *localTram* | *regionalTram* | *sightseeingTram* | *shuttleTram*	Unter-Klassifizierung der Straßenbahnen (nach TPEG pti_table 06).
WaterSubmodeEnumeration	*unknown* | *undefined* | *internationalCarFerry* | *nationalCarFerry* | *regionalCarFerry* | *localCarFerry* | *internationalPassengerFerry* | *nationalPassengerFerry* | *regionalPassengerFerry* | *localPassengerFerry* | *postBoat* | *trainFerry* | *roadFerryLink* | *airportBoatLink* | *highSpeedVehicleService* | *highSpeedPassengerService* | *sightseeingService* | *schoolBoat* | *cableFerry* | *riverBus* | *scheduledFerry* | *shuttleFerryService*	Unter-Klassifizierung der Wasserverkehrsmittel (nach TPEG pti_table 07).

AirSubmodeEnumeration	*unknown* | *undefined* | *internationalFlight* | *domesticFlight* | *intercontinentalFlight* | *domesticScheduledFlight* | *shuttleFlight* | *intercontinentalCharterFlight* | *internationalCharterFlight* | *roundTripCharterFlight* | *sightseeingFlight* | *helicopterService* | *domesticCharterFlight* | *SchengenAreaFlight* | *airshipService* | *shortHaulInternationalFlight* | *canalBarge*	Unter-Klassifizierung der Luftverkehrsmittel (nach TPEG pti_table 08).
TelecabinSubmodeEnumeration	*unknown* | *undefined* | *telecabin* | *cableCar* | *lift* | *chairLift* | *dragLift* | *telecabinLink*	Unter-Klassifizierung der Lift- und Aufzugsarten (nach TPEG pti_table 09).
FunicularSubmodeEnumeration	*unknown* | *funicular* | *allFunicularServices* | *undefinedFunicular*	Unter-Klassifizierung der Seilbahnen (nach TPEG pti_table 10).
TaxiSubmodeEnumeration	*unknown* | *undefined* | *communalTaxi* | *waterTaxi* | *railTaxi* | *bikeTaxi* | *blackCab* | *miniCab* | *allTaxiServices*	Unter-Klassifizierung der Taxiarten (nach TPEG pti_table 11).

Tabelle 14: Liste der einfachen Typdefinitionen in Trias_ModesSupport.xsd.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_ModesSupport definiert sind.

7.3.2. IndividualTransportOptionsStructure

IndividualTransportOptionsStructure			*+Structure*	**Arten von IV und deren Nutzungsgrenzen, wie sie der Benutzer vorgegeben hat.**
	Mode	**1:1**	*IndividualModesEnumeration*	Angabe des IV-Typs. Zugelassen sind hier Werte für Fußweg, Fahrrad, Taxi, selbst gefahrenes Auto, durch andere gefahrenes Auto, Motorrad und LKW. Der Modus „selbst gefahrenes Auto“ benötigt beim Umstieg in ein anderes Verkehrsmittel einen längerfristigen Parkplatz und ist daher ein verallgemeinertes Synonym für Park&Ride. Der Modus „durch andere gefahrenes Auto“ benötigt dagegen nur einen Platz zum Aussteigen lassen.
	MaxDistance	0:1	*Distance*	Maximale Distanz, bis zu der die Nutzung dieses IV-Typs zugelassen ist.
	MaxDuration	0:1	*xs:duration*	Maximale Zeitdauer, bis zu der die Nutzung dieses IV-Typs zugelassen ist.
	MinDistance	0:1	*Distance*	Minimale Distanz, ab der die Nutzung dieses IV-Typs zugelassen ist.
	MinDuration	0:1	*xs:duration*	Minimale Zeitdauer, ab der die Nutzung dieses IV-Typs zugelassen ist.
	Speed	0:1	*OpenPercent*	Relative Geschwindigkeit in Prozent. Wert 100 stellt Standardgeschwindigkeit dar. Werte kleiner 100 verringern die Geschwindigkeit, Werte größer 100 vergrößern die Geschwindigkeit anteilig.

Tabelle 15: Beschreibung der Struktur ***IndividualTransportOptionsStructure***.

7.3.3. PtSubmodeChoiceGroup

PtSubmodeChoiceGroup				*+Group*	**Gruppe zur Auswahl der Verkehrsmitteluntertypen**.
	a	*AirSubmode*	-0:1	*AirSubmodeEnumeration*	Untertypen der Luftverkehrsmittel.
	b	*BusSubmode*		*BusSubmodeEnumeration*	Untertypen der Busse.
	c	*CoachSubmode*		*CoachSubmodeEnumeration*	Untertypen der Überlandbusse.
	d	*FunicularSubmode*		*FunicularSubmodeEnumeration*	Untertypen der Seilbahnen.
	e	*MetroSubmode*		*MetroSubmodeEnumeration*	Untertypen der Untergrundbahnen.
	f	*RailSubmode*		*RailSubmodeEnumeration*	Untertypen der Züge.
	g	*TelecabinSubmode*		*TelecabinSubmodeEnumeration*	Untertypen der Lift- und Aufzugsarten.
	h	*TramSubmode*		*TramSubmodeEnumeration*	Untertypen der Straßenbahnen.
	i	*WaterSubmode*		*WaterSubmodeEnumeration*	Untertypen der Wasserverkehrsmittel.

Tabelle 16: Beschreibung der Gruppe ***PtSubmodeChoiceGroup***.

7.3.4. ModeStructure

ModeStructure			*+Structure*	**Verkehrsmittel mit Klassifizierung und Namen.**
Mode	***PtMode***	**1:1**	*PtModesEnumeration*	Angabe der ÖV-Verkehrsmittelart.
PtSubmodeChoice	*:::*	-0:1	*PtSubmodeChoice*	Verkehrsmitteluntertypen (vgl. 7.3.3)
	Name	0:1	*InternationalText*	Verkehrsmittelname.
	ShortName	0:1	*InternationalText*	Kurzname oder Abkürzung.
	Description	0:1	*InternationalText*	Beschreibender Text.

Tabelle 17: Beschreibung der Struktur ***ModeStructure***.

7.3.5. PtModeFilterStructure

PtModeFilterStructure			*+Structure*	**Struktur zum Filtern nach Verkehrsmitteltypen**
	Exclude	0:1	*xs:boolean*	Indikator, ob die in der Liste angegebenen Verkehrsmittel ausgeschlossen (Wert *true*) oder als einzige verwendet werden sollen (Wert *false*). Voreinstellung ist *true*.
	PtMode	0:*	*PtModesEnumeration*	ÖV-Verkehrsmitteltypen.
PtSubmodeChoice	*:::*	0:*	*PtSubmodeChoice*	ÖV-Untertypen.

Tabelle 18: Beschreibung der Struktur ***PtModeFilterStructure***.

7.4. Trias_Common

7.4.1. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

ParticipantCodeType	*xs:normalizedString*	ID eines Kommunikationspartners.
OperatorCodeType	*xs:NMTOKEN*	ID eines Verkehrsunternehmens.
LineCodeType	*xs:NMTOKEN*	ID einer Linie.
DirectionCodeType	*xs:NMTOKEN*	ID einer Linienrichtung.
JourneyCodeType	*xs:NMTOKEN*	ID einer Fahrplanfahrt.
VehicleCodeType	*xs:NMTOKEN*	ID eines Fahrzeugs.
FacilityCodeType	*xs:NMTOKEN*	ID einer Einrichtung/Ausstattung.
OwnerCodeType	*xs:NMTOKEN*	ID einer verantwortlichen Organisation (Eigentümer).
OperatingDayCodeType	*xs:NMTOKEN*	ID eines Verkehrstags.

Tabelle 19: Liste der einfachen Typdefinitionen in Trias_Common.xsd.

Damit die Codes von Linien, Verkehrsunternehmen etc. systemübergreifend verstanden werden können, sind bestimmte Vereinbarungen zu treffen. Diese sind in Kapitel 5 beschrieben.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_Common definiert sind.

7.4.2. ErrorMessageStructure

ErrorMessageStructure			*+Structure*	**Struktur zur Meldung von Fehlerzuständen.**
	Code	**1:1**	*xs:normalized String*	Code des Fehlerzustands.
	Text	0:1	*+International Text*	Beschreibung des Fehlerzustands.

Tabelle 20: Beschreibung der Struktur ***ErrorMessageStructure***.

7.4.3. PrivateCodeStructure

PrivateCodeStructure			*+Structure*	**Objekt-ID innerhalb eines proprietären (privaten) Schlüsselsystems (Fremdschlüssel).**
	System	**1:1**	*xs:NMTOKEN*	Bezeichnung des Schlüsselsystems.
	Value	**1:1**	*xs:NMTOKEN*	Code/Objekt-ID.

Tabelle 21: Beschreibung der Struktur ***PrivateCodeStructure***.

7.4.4. OperatorFilterStructure

OperatorFilterStructure			*+Structure*	**Struktur zum Filtern nach Verkehrsunternehmen.**
	Exclude	0:1	*xs:boolean*	Indikator, ob die in der Liste angegebenen Verkehrsunternehmen ausgeschlossen (Wert *true*) oder als einzige verwendet werden sollen (Wert *false*). Voreinstellung ist *true*.
	OperatorRef	0:*	*→Operator*	Referenz auf Verkehrsunternehmen. Vgl. 7.4.1.

Tabelle 22: Beschreibung der Struktur ***OperatorFilterStructure***

7.4.5. LineDirectionStructure

LineDirectionStructure			*+Structure*	**Linien-ID, evtl. verfeinert auf eine Richtung**
	LineRef	**1:1**	*→LineCode*	Referenz auf die Linie. Vgl. 7.4.1.
	DirectionRef	0:1	*→Direction-Code*	Referenz auf die Linienrichtung. Vgl. 7.4.1.

Tabelle 23: Beschreibung der Struktur ***LineDirectionStructure***.

7.4.6. LineDirectionFilterStructure

LineDirectionFilterStructure			*+Structure*	**Filterstruktur zum Ein/Ausschließen von Linien(richtungen)**
	Line	**1:***	*+LineDirection*	Referenz auf die Linie (vgl. 7.4.5).
	Exclude	0:1	*xs:boolean*	Indikator, ob die Linien(richtungen) dieser Liste in die Suche aufgenommen oder von ihr ausgeschlossen werden sollen. Default ist Ausschluss (Exclude).

Tabelle 24: Beschreibung der Struktur ***LineDirectionFilterStructure***.

7.4.7. SharingServiceStructure

SharingServiceStructure			*+Structure*	**Struktur zur Beschreibung eines Mobilitätsdienstes mit Ausleihfahrzeugen**
	OperatorRef	**1:1**	*→Operator*	Operator-ID. Vgl. 7.4.1.
	Name	0:1	*xs:string*	Name des Mobilitätsdienstes.
	SharingModel	0:1	*singleStationBased* | *multipleStationBased* | *nonStationBased*	Typ des Ausleih- und Rückgabeverfahrens.
SharingServiceUsage	*TimeBufferBefore*	0:1	*xs:duration*	Typische Zeit, die ein Benutzer einplanen muss, um sich am System anzumelden und das Fahrzeug in Fahrbereitschaft zu versetzen.
	TimeBufferAfter	0:1	*xs:duration*	Typische Zeit, die ein Benutzer einplanen muss, um das Fahrzeug ordnungsgemäß abzustellen, zu verschließen und sich am System abzumelden.
	InfoURL	0:1	*+WebLink*	Link zu Web-Seite mit weiterführenden Informationen (vgl. 7.2.4).

Tabelle 25: Beschreibung der Struktur ***SharingServiceStructure***.

7.4.8. OperatingDaysStructure

OperatingDaysStructure			*+Structure*	**Struktur für die Definition von Verkehrstagen mittels Bit-Kette.**
	From	**1:1**	*xs:date*	Startdatum des Zeitraums.
	To	**1:1**	*xs:date*	Enddatum des Zeitraums.
	Pattern	**1:1**	*BitString*	Bitmuster für die Verkehrstage im Zeitraum von Startdatum (*From*) bis Enddatum (*To*). Die Länge des Bitmusters in *Pattern* entspricht der Anzahl der Tage von *From* bis *To*. Eine „1“ bedeutet, dass das in Frage kommende Ereignis an dem Tag stattfindet, der der Position in der Bitkette entspricht.

Tabelle 26: Beschreibung der Struktur ***OperatingDaysStructure***.

7.4.9. WeekdayTimePeriodStructure

WeekdayTimePeriodStructure			*+Structure*	**Struktur für die Definition von Zeitintervallen an einem Wochentag.**
	Weekday	0:1	*Sunday* | *Monday* | *Tuesday* | *Wednesday* | *Thursday* | *Friday* | *Saturday* | *PublicHoliday*	Wochentagstyp.
	StartTime	**1:1**	*xs:time*	Startzeit des Zeitintervalls.
	Duration	**1:1**	*xs:duration*	Dauer des Zeitintervalls.

Tabelle 27: Beschreibung der Struktur ***WeekdayTimePeriodStructure***.

7.4.10. GeneralAttributeStructure

GeneralAttributeStructure			*+Structure*	**Struktur für die Definition von Attributen/Hinweisen**.
	Text	**1:1**	*+International Text*	Attributtext für die Fahrgastinformation.
	Code	**1:1**	*xs:NMTOKEN*	Interner Attribute-Code. Kann verwendet werden, um mehrfaches Auftreten desselben Attributs zu erkennen.
AllFacilities	*:::*	0:1	*+AllFacilitiesGroup*	Klassifizierung nach TPEG. Vgl. 7.7.4.
	Mandatory	0:1	*xs:boolean*	Legt fest, ob das Attribut in jedem Fall angezeigt werden muss. Voreinstellung ist *false*.
	Importance	0:1	*Percent*	Wichtigkeit für die Priorisierung von Attributen gegeneinander.
	InfoURL	0:1	*xs:anyURI*	URL zu weiteren Informationen über dieses Attribut. Falls vorhanden, soll der gesamte Text als Link zu dieser URL gekennzeichnet werden.

Tabelle 28: Beschreibung der Struktur ***GeneralAttributeStructure***.

7.5. Trias_LocationSupport

In der XML-Schema-Definition Trias_LocationSupport.xsd werden eine Reihe von Basistypen und -strukturen definiert, die als Ortsbezüge (Haltestellen, Haltepunkte, Ortschaften und POIs) und Beschreibungen von Haltestellen und Haltepunkten in anderen Definitionen wieder verwendet werden können.

7.5.1. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

StopPointCodeType	*xs:normalizedString*	Code für einen Haltepunkt.
StopPlaceCodeType	*xs: normalizedString*	Code für eine Haltestelle.
LocalityCodeType	*xs: normalizedString*	Code für eine Ortschaft.
PointOfInterestCodeType	*xs: normalizedString*	Code für einen POI.
AddressCodeType	*xs: normalizedString*	Code für eine Adresse.

Tabelle 29: Liste der einfachen Typdefinitionen in Trias_LocationSupport.xsd.

Damit die Codes von Haltestellen, Haltepunkten etc. systemübergreifend verstanden werden können, sind bestimmte Vereinbarungen zu treffen. Diese sind in Kapitel 5 beschrieben.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_LocationSupport definiert sind.

7.5.2. StopPointStructure

StopPointStructure			*+Structure*	**Modellierung eines Haltepunkts.**
Stop-Point	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+International Text*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
	PrivateCode	0:*	*+PrivateCode*	Privater Code für diesen Haltepunkt in einem anderen Schlüsselsystem. Vgl. 7.4.3.
	ParentRef	0:1	*→StopPlace*	Referenz auf die Haltestelle, zu der dieser Haltepunkt gehört. Vgl. 7.5.1.
	LocalityRef	0:1	*→Locality*	Referenz auf die Ortschaft, zu der dieser Haltepunkt gehört. Vgl. 7.5.1.
Sto-pAttribu-tes	*WheelchairAccessible*	0:1	*xs:boolean*	Rollstuhltauglichkeit dieses Haltepunkts.
	Lighting	0:1	*xs:boolean*	Angabe zur Beleuchtung dieses Haltepunkts.
	Covered	0:1	*xs:boolean*	Angabe, ob dieser Haltepunkt Witterungsschutz bietet (vor Regen, Schnee, Sturm etc.).

Tabelle 30: Beschreibung der Struktur ***StopPointStructure***.

7.5.3. StopPlaceStructure

StopPlaceStructure			*+Structure*	**Modellierung einer Haltestelle.**
StopP-lace	***StopPlaceRef***	**1:1**	*→StopPlace*	Referenz auf einen Code für eine Haltestelle. Vgl. 7.5.1.
	StopPlaceName	**1:1**	*+International Text*	Name der Haltestelle für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „Messe/Exhibition Center“.
	PrivateCode	0:*	*+PrivateCode*	Privater Code für diese Haltestelle in einem anderen Schlüsselsystem.
	LocalityRef	0:1	*→Locality*	Referenz auf die Ortschaft, zu der diese Haltestelle gehört. Vgl. 7.5.1.
Sto-pAttribu-tes	*WheelchairAccessible*	0:1	*xs:boolean*	Rollstuhltauglichkeit dieser Haltestelle insgesamt.
	Lighting	0:1	*xs:boolean*	Angabe zur Beleuchtung dieser Haltestelle.
	Covered	0:1	*xs:boolean*	Angabe, ob diese Haltestelle Witterungsschutz bietet (vor Regen, Schnee, Sturm etc.).

Tabelle 31: Beschreibung der Struktur ***StopPlaceStructure***.

7.5.4. LocalityStructure

LocalityStructure			*+Structure*	**Modellierung einer Ortschaft/Stadt.**
	LocalityCode	**1:1**	*→Locality*	Identifikator der Ortschaft/Stadt. Vgl. 7.5.1.
	LocalityName	**1:1**	*+International Text*	Name der Ortschaft für Fahrgastinformation.
	PrivateCode	0:*	*+PrivateCode*	Privater Code für diesen Haltepunkt in einem anderen Schlüsselsystem.
	ParentRef	0:1	*→Locality*	Referenz auf eine übergeordnete Ortschaft, zu der diese Ortschaft gehört, z. B. Beziehung Stadtteil zu Stadt. Vgl. 7.5.1.
Area	*Points*	3:*	*+GeoPosition*	Polygonzug, der das Gebiet der Ortschaft beschreibt.

Tabelle 32: Beschreibung der Struktur ***LocalityStructure***.

7.5.5. PointOfInterestStructure

PointOfInterestStructure			*+Structure*	**Modellierung eines wichtigen Punkts (POI).**
	PointOfInterestCode	**1:1**	*→PointOfInterest*	Identifikator des wichtigen Punkts.
	PointOfInterestName	**1:1**	*+International Text*	Name des wichtigen Punkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „Messe/Exhibition Center“.
	PointOfInterestCategory	0:*	*+PointOfInterestCategory*	Kategorien, die diesem POI zugeordnet sind. Vgl. 7.5.6. Falls mehrere aufgeführt sind, sind sie nach absteigender Relevanz sortiert.
	PrivateCode	0:*	*+PrivateCode*	Privater Code für diesen Haltepunkt in einem anderen Schlüsselsystem.
	LocalityRef	0:1	*→Locality*	Referenz auf die zugeordnete Ortschaft, zu der dieser wichtige Punkt gehört. Vgl. 7.5.1.

Tabelle 33: Beschreibung der Struktur ***PointOfInterestStructure***.

7.5.6. PointOfInterestCategoryStructure

PointOfInterestCategoryStructure			*+Structure*	**Modellierung einer POI-Kategorie-Liste.**
	OsmTag	1:*	*+OsmTag*	Liste der POI-Kategorien, definiert durch Schlüssel-Wert-Paare wie in OpenStreetMap.[7] Vgl. 7.5.7

Tabelle 34: Beschreibung der Struktur ***PointOfInterestCategoryStructure***.

[7] http://wiki.openstreetmap.org/wiki/DE:Map_Features

7.5.7. OsmTagStructure

OsmTagStructure			*+Structure*	**Modellierung einer POI-Kategorie.**
	Tag	**1:1**	*xs:NMTOKEN*	Name des OpenStreetMap-Tags (z.B. amenity, leisure, tourism, bike, ...)
	Value	**1:1**	*xs:NMTOKEN*	Wert des OpenStreetMap-Tags (z.B. yes, hostel, charging_station, ...)

Tabelle 35: Beschreibung der Struktur ***OsmTagStructure***.

7.5.8. PointOfInterestFilterStructure

PointOfInterestFilterStructure			*+Structure*	**Struktur zum Filtern nach POI-Kategorien**
	Exclude	0:1	*xs:boolean*	Legt fest, ob die nachfolgenden Kategorien bei der POI-Suche als einzige eingeschlossen (Exclude=*false*) oder ausgeschlossen werden sollen (Exclude=*true*). Default ist *false*.
	PointOfInterestCategory	**1:***	*+PointOfInterestCategory*	Bezeichner für POI-Kategorien. Vgl. 7.5.6. Wenn mehrere aufgelistet sind, werden die Kategorien bei der Suche mit einem logischen „ODER“ (im Fall von Exclude=*false*) bzw. mit einem logischen „UND“ (im Fall von Exclude=*true*) berücksichtigt.

Tabelle 36: Beschreibung der Struktur ***PointOfInterestFilterStructure***.

7.5.9. AddressStructure

AddressStructure			*+Structure*	**Modellierung einer Adresse.**
	AddressCode	**1:1**	*→Address*	Identifikator der Adresse. Vgl. 7.5.1.
	PrivateCode	0:*	*+PrivateCode*	Privater Code für diesen Haltepunkt in einem anderen Schlüsselsystem.
	AddressName	**1:1**	*+International Text*	Adressbeschriftung für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „Messe/Exhibition Center“.
Address Detail	*CountryName*	0:1	*xs:string*	Angabe zum Land.
	PostalCode	0:1	*xs:string*	Postleitzahl.
	LocalityName	0:1	*xs:string*	Name der Stadt oder Ortschaft, in der die Adresse liegt.
	LocalityRef	0:1	*→Locality*	Referenz auf die Stadt oder Ortschaft, zu der diese Adresse gehört. Vgl. 7.5.1.
	StreetName	0:1	*xs:string*	Name der Straße, in der die Adresse liegt.
	HouseNumber	0:1	*xs:string*	Hausnummer. Falls leer, kann a) in *CrossingStreet* eine Kreuzung angegeben werden oder b) die Straße ist als Ganzes gemeint.
	CrossingStreet	0:1	*xs:string*	Name der kreuzenden Straße.

Tabelle 37: Beschreibung der Struktur ***AddressStructure***.

7.5.10. LocationStructure

LocationStructure				*+Structure*	**Modell eines allgemeinen Ortspunkts (Haltepunkt, Haltestelle, Koordinatenposition, Ortschaft, POI oder Adresse).**
	a	*StopPoint*	-0:1	*+StopPoint*	Angaben zu einem Haltepunkt. Vgl.7.5.2.
	b	*StopPlace*		*+StopPlace*	Angaben zur Haltestelle. Vgl. 7.5.3.
	c	*Locality*		*+Locality*	Angaben zu einer Stadt/Ortschaft. Vgl. 7.5.4.
	d	*PointOfInterest*		*+PointOfInterest*	Angaben zu einem POI. Vgl. 7.5.5.
	e	*Address*		*+Address*	Angaben zu einer Adresse. Vgl.7.5.9.
	LocationName		**1:1**	*+International Text*	Name oder Bezeichnung des Ortspunkts.
	GeoPosition		**1:1**	*+GeoPosition*	Koordinatenposition. Vgl. 7.2.3..
	Attribute		*0:**	*+GeneralAttribute*	Attribute, die dem Ortspunkt zugeordnet sind. Vgl. 7.4.10.
	Extension		0:1	*xs:anyType*	Erweiterungen.

Tabelle 38: Beschreibung der Struktur ***LocationStructure***.

7.5.11. LocationRefStructure

LocationRefStructure				*+Structure*	**Referenz auf einen allgemeinen Ortspunkt (Haltepunkt, Haltestelle, Koordinatenposition, Ortschaft oder POI).**
	a	***StopPointRef***	**-1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	b	***StopPlaceRef***		*→StopPlace*	Referenz auf einen Code für eine Haltestelle. Vgl. 7.5.1.
	c	***GeoPosition***		*+GeoPosition*	Koordinatenposition.
	d	***LocalityRef***		*→Locality*	Referenz auf einen Code für eine Ortschaft. Vgl. 7.5.1.
	e	***PointOfInterestRef***		*→PointOfInterest*	Referenz auf einen Code für einen POI. Vgl. 7.5.1.
	f	***AddressRef***		*→Address*	Referenz auf eine Adresse. Vgl. 7.5.1.
	LocationName		**1:1**	*+International Text*	Name oder Bezeichnung des Ortspunkts.

Tabelle 39: Beschreibung der Struktur ***LocationRefStructure***.

7.6. Trias_JourneySupport

In der XML-Schema-Definition Trias_JourneySupport.xsd werden Strukturen beschrieben, die den Fahrbetrieb des ÖV beschreiben. Dazu gehören u. a. die Beschreibung einer Fahrzeugfahrt, Informationen zu Ankünften und Abfahrten an Haltestellen, sowie die Bewegung eines Fahrzeugs entlang des Fahrwegs.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_JourneySupport definiert sind.

7.6.1. ServiceViaPointStructure

ServiceViaPointStructure			*+Structure*	**Via-Punkt auf dem Fahrweg.**
Stop-Point	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+International Text*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
	DisplayPriority	0:1	*Priority*	Priorität, mit der dieser Via-Punkt angezeigt werden soll (z. B. falls Platz knapp ist und nicht alle Via-Punkte angezeigt werden können).

Tabelle 40: Beschreibung der Struktur ***ServiceViaPointStructure***.

7.6.2. ViaStructure

ViaStructure			*+Structure*	**Angaben zu einer Via-Bedingung.**
	ViaPoint	**1:1**	*+LocationRef*	Referenz auf den Via-Punkt. Vgl. 7.5.11.
	DwellTime	0:*	*xs:duration*	Vom Benutzer vorgeschriebene Mindestaufenthaltszeit am Via-Punkt.

Tabelle 41: Beschreibung der Struktur ***ViaStructure***.

7.6.3. DatedServiceGroup

DatedServiceGroup			*+Group*	**Gruppe zur Beschreibung der Fahrt einer Linie an einem bestimmten Tag.**
	OperatingDayRef	**1:1**	*→Operating-Day*	Betriebstag der Fahrt. Vgl. 7.4.1.
	VehicleRef	0:1	*→Vehicle*	Fahrzeug-ID. Vgl. 7.4.1.
Service-Journey	***JourneyRef***	**1:1**	*→Journey*	Fahrt-ID. Vgl. 7.4.1.
LineIdentity	***LineRef***	**1:1**	*→Line*	Linien-ID. Vgl. 7.4.1.
	DirectionRef	**1:1**	*→Direction*	Richtungs-ID. Vgl. 7.4.1.
Service	***Mode***	**1:1**	*+Mode*	Verkehrsmitteltyp. Vgl. 7.3.4.
	PublishedLineName	**1:1**	*+International Text*	Liniennummer oder –name, wie in der Öffentlichkeit bekannt.
	OperatorRef	0:1	*→Operator*	Operator-ID. Vgl. 7.4.1.
	RouteDescription	0:1	*+International Text*	Beschreibung des Fahrwegs.
	Via	0:*	*+ServiceViaPoint*	Wichtige Halte auf dem Fahrweg. Vgl.7.6.1.
	Attribute	0:*	*+GeneralAttribute*	Hinweise und Attribute (mit Klassifikationen) zur Fahrt. Vgl. 7.4.10.

Tabelle 42: Beschreibung der Gruppe ***DatedServiceGroup***.

7.6.4. DatedJourneyStructure

DatedJourneyStructure			*+Structure*	**Fahrplanfahrt an bestimmtem Tag.**
	:::	1:1	*+DatedServiceGroup*	Fahrt einer Linie am Stichtag (vgl. 7.6.3).
ServiceOrigin	*OriginStopPointRef*	0:1	*→StopPoint*	ID des ersten Haltepunkts der Fahrt; Starthaltestelle. Vgl. 7.5.1.
	OriginText	0:1	*+International Text*	Name des ersten Haltepunkts der Fahrt, der Starthaltestelle.
Service-Destination	*DestinationStopPointRef*	0:1	*→StopPoint*	ID des letzten Haltepunkts der Fahrt; Endhaltestelle. Vgl. 7.5.1.
	DestinationText	**1:1**	*+International Text*	Name des letzten Haltepunkts der Fahrt, der Endhaltestelle.
ServiceStatus	*Unplanned*	0:1	*xs:boolean*	Gibt an, ob es sich um eine zusätzliche, ungeplante Fahrt handelt. Voreinstellung ist *false*.
	Cancelled	0:1	*xs:boolean*	Gibt an, ob diese Fahrt zur Gänze entfällt. Voreinstellung ist *false*.
	Deviation	0:1	*xs:boolean*	Gibt an, ob diese Fahrt einen anderen Weg nimmt. Voreinstellung ist *false*.
	Occupancy	0:1	*manySeatsAvailable | fewSeatsAvailable | noSeatsAvailable | standingAvailable | full*	Auslastungszustand des Fahrzeugs.
	SituationFullRef	0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im Kontext der Antwort (Response context)) zu finden sein oder auf anderem Wege bekannt gemacht werden. Vgl.7.8.2.

Tabelle 43: Beschreibung der Struktur ***DatedJourneyStructure***.

7.6.5. TripLocationStructure

TripLocationStructure			*+Structure*	**Fahrplanfahrt als momentaner Aufenthaltsort eines Fahrgasts**
	OperatingDayRef	**1:1**	*→OperatingDay*	Betriebstag der Fahrt. Vgl. 7.4.1.
	JourneyRef	**1:1**	*→Journey*	Fahrt-ID. Vgl. 7.4.1.
LineIdentity	***LineRef***	**1:1**	*→Line*	Linien-ID. Vgl. Vgl. 7.4.1.
	DirectionRef	**1:1**	*→Direction*	Richtungs-ID. Vgl. 7.4.1.

Tabelle 44: Beschreibung der Struktur ***TripLocationStructure***.

7.6.6. CallAtStopStructure

CallAtStopStructure			*+Structure*	**Halt einer Fahrt an einem Haltepunkt oder Haltestelle.**
StopPoint	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+InternationalText*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+InternationalText*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+InternationalText*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+InternationalText*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
ServiceArrival	***TimetabledTime***	**1:1**	*xs:dateTime*	Ankunftszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Ankunftszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Ankunftszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Ankunftszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Ankunftszeit.
ServiceDeparture	***TimetabledTime***	**1:1**	*xs:dateTime*	Abfahrtszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Abfahrtszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Abfahrtszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Abfahrtszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Abfahrtszeit.
StopCallStatus	***StopSeqNumber***	**1:1**	*xs:positiveInteger*	Laufende Nummer des Halts im Fahrweg der Fahrt.
	DemandStop	0:1	*xs:boolean*	Bedarfshalt. Fahrzeug bedient diesen Halt nur nach Voranmeldung.
	UnplannedStop	0:1	*xs:boolean*	Halt, der laut Planung nicht vorgesehen war.
	NotServicedStop	0:1	*xs:boolean*	Entgegen der Planung findet kein Halt statt.
	SituationFullRef	0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im ResponseContext der Antwort zu finden sein oder auf anderem Wege bekannt gemacht werden. Vgl. 7.8.2.

Tabelle 45: Beschreibung der Struktur ***CallAtStopStructure***.

7.6.7. DatedCallAtLocationStructure

DatedCallAtLocationStructure			*+Structure*	**Bedienung eines allgemeinen Ortspunkts durch eine Fahrt an einem bestimmten Tag**
DatedJourney-Ref	***JourneyRef***	**1:1**	*→Journey*	Fahrt-ID. Vgl. 7.4.1.
	OperatingDayRef	**1:1**	*→Operating-Day*	Betriebstag der Fahrt. Vgl. 7.4.1.
	CallLocation	**1:1**	*+LocationRef*	Verallgemeinerter Abfahrtspunkt. Im Normalfall eine Haltestelle, kann aber auch eine Adresse oder Koordinate sein, wenn es sich um flexible Linien oder Flächenbedarfsverkehre handelt. Vgl. 7.5.11.
ServiceArrival	***TimetabledTime***	**1:1**	*xs:dateTime*	Ankunftszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Ankunftszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Ankunftszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Ankunftszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Ankunftszeit.
ServiceDeparture	***TimetabledTime***	**1:1**	*xs:dateTime*	Abfahrtszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Abfahrtszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Abfahrtszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Abfahrtszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Abfahrtszeit.
StopCallStatus	***StopSeqNumber***	**1:1**	*xs:positiveInteger*	Laufende Nummer des Halts im Fahrweg der Fahrt.
	DemandStop	0:1	*xs:boolean*	Bedarfshalt. Fahrzeug bedient diesen Halt nur nach Voranmeldung.
	UnplannedStop	0:1	*xs:boolean*	Halt, der laut Planung nicht vorgesehen war.
	NotServicedStop	0:1	*xs:boolean*	Entgegen der Planung findet kein Halt statt.

Tabelle 46: Beschreibung der Struktur ***DatedCallAtLocationStructure***.

7.6.8. ContinuousServiceStructure

ContinuousServiceStructure				*+Structure*	**Eine Fahrgastbewegung mit Hilfe eines kontinuierlichen, nicht fahrplangebundenen Verkehrsmittels.**
	a	***ContinuousMode***	**-1:1**	*walk | demandResponsive | replacementService*	Modalität für kontinuierliche Verkehre.
	b	***IndividualMode***		*walk | cycle | taxi | self-drive-car | others-drive-car | motorcycle | truck*	Verkehrsmittelmodalität für Individualverkehr.
	a	*:::*	-0:1	*+DatedServiceGroup*	Beschreibung einer ÖV-Fahrmöglichkeit am Stichtag (vgl. 7.6.3).
	b	*SharingService*		*+SharingService*	Beschreibung eines Mobilitätsangebots mit Ausleihfahrzeugen (vgl. 7.4.7).
ServiceOrigin	*OriginStopPointRef*		0:1	*→StopPoint*	ID des ersten Haltepunkts der Fahrt; Starthaltestelle. Vgl. 7.5.1.
	OriginText		0:1	*InternationalText*	Name des ersten Haltepunkts der Fahrt, der Starthaltestelle.
Service-Destination	*DestinationStopPointRef*		0:1	*→StopPoint*	ID des letzten Haltepunkts der Fahrt; Endhaltestelle. Vgl. 7.5.1.
	DestinationText		0:1	*InternationalText*	Name des letzten Haltepunkts der Fahrt, der Endhaltestelle.
	SituationFullRef		0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im Kontext der Meldung (ResponseContext) zu finden sein oder auf anderem Wege bekannt gemacht werden.

Tabelle 47: Beschreibung der Struktur ***ContinuousServiceStructure***.

7.6.9. VehiclePositionStructure

VehiclePositionStructure			*+Structure*	**Geographische und logische Position eines Fahrzeugs.**
	GeoPosition	0:1	*+GeoPosition*	Geographische Position (vgl. 7.2.3).
	Progress	0:1	*Not yet operated | Operation finished | At stop | Between stops*	Logische Position bezogen auf Haltestellenfolge im Fahrplan.
	Bearing	0:1	*AbsoluteBearing*	Kompassrichtung in Grad, in die sich das Fahrzeug bewegt (vgl. 7.2.1).
	ProgressBetweenStops	0:1	*+ProgressBetweenStops*	Position zwischen der zuletzt besuchten Haltestelle und der aktuellen Position (vgl. 7.6.10).

Tabelle 48: Beschreibung der Struktur ***VehiclePositionStructure***.

7.6.10. ProgressBetweenStopsStructure

ProgressBetweenStopsStructure			*+Structure*	**Position zwischen der zuletzt besuchten Haltestelle und der aktuellen Position.**
	LinkDistance	0:1	*Distance*	Gesamtdistanz in Meter zwischen der letzten und der nächsten Haltestelle.
	Percentage	0:1	*Percent*	Prozentanteil, den das Fahrzeug entlang der Streckendistanz (LinkDistance) zurückgelegt hat.

Tabelle 49: Beschreibung der Struktur ***ProgressBetweenStopsStructure***.

7.6.11. LocationContextStructure

LocationContextStructure				*+Structure*	**Angabe eines Orts und der Möglichkeiten, wie ein Benutzer ihn per IV erreichen kann.**
	a	***LocationRef***	**-1:1**	*+LocationRef*	Angabe eines räumlichen Orts (vgl. 7.5.11).
	b	***TripLocation***		*+TripLocation*	Aufenthaltsort in einem (sich bewegenden) Fahrzeug (vgl. 7.6.5).
	DepArrTime		0:1	*xs:dateTime*	Beabsichtigte Abfahrts- oder Ankunftszeit an dem in *Location* oder *TripLocation* bezeichneten Ort.
	IndividualTransportOptions		0:*	*+IndividualTransportOptions*	Angaben des Benutzers, wie er/sie den Ort mittels IV erreichen/verlassen könnte (vgl. 7.3.2).

Tabelle 50: Beschreibung der Struktur ***LocationContextStructure***.

Elemente vom Typ *LocationContextStructure* werden vor allem dazu benutzt, um den Start- (oder Ziel-) Kontext zu beschreiben, in dem ein Reisender sich befindet. Beim Dienst Verbindungsauskunft z. B. dienen Elemente dieses Typs als Angabe von Start und Ziel. Die Implementierung des Suchalgorithmus ist dabei selbst dafür zuständig, die Ortsangaben, etwa eine Koordinate, auf die internen Elemente (z. B. Knoten oder Kanten) des Such-Netzwerks abzubilden.

Die *IndividualTransportOptions* geben dabei an, mit welchen IV-Verkehrsmitteln der Benutzer bereit ist, die Haltestelle zu erreichen/verlassen. Im Standardfall ist das ein Fußweg, es kommen aber auch das Fahrrad, PKW und Taxi in Betracht. Beim Fahrrad muss man in den Optionen z. B. der Verbindungsauskunft wählen, ob man das Fahrrad in den ÖV-Fahrzeugen mitnehmen will. Dies zieht dann unter Umständen eine andere Reiseplanung nach sich. Bei Pkw wird zwischen „selber fahren“ und „gefahren werden“ unterschieden. Im ersten Fall muss in der Reiseplanung ein Weg zu einem Parkplatz berücksichtigt werden, wohingegen im zweiten Fall ein Anhalten zum Aussteigen genügt.

7.6.12. AbstractResponseContextStructure

AbstractResponseContextStructure			*+Structure*	**Basisstruktur für Antwortkontext. Hier können Objekte abgelegt werden, die mehrfach auftreten und durch Referenzen auf den Kontext ersetzt werden können.**
Locations	*Location*	0:*	*+Location*	Modellierung von Ortspunkten (vgl. 7.5.10).
Situations	*Situation*	0:*	*+siri:PtSituationElement*	SIRI-Modellierung eines Ereignisses oder einer Störung (vgl. 7.8.1).

Tabelle 51: Beschreibung der Struktur ***AbstractResponseContextStructure***.

7.6.13. LegAttributeStructure

LegAttributeStructure			*+Structure (abgeleitet von GeneralAttributeStructure, vgl. 7.4.10)*	**Definition von Attributen und Hinweisen, die nur auf Teilen einer Verbindung gültig sind.**
	FromStopSeqNumber	0:1	*xs:positiveInteger*	Fahrwegpositionsnummer des Haltepunkts, ab dem das Attribut gültig ist. Falls leer, dann gültig ab Beginn des Fahrwegs.
	ToStopSeqNumber	0:1	*xs:positiveInteger*	Fahrwegpositionsnummer des Haltepunkts, bis zu dem das Attribut gültig ist. Falls leer, dann gültig bis zum Ende des Fahrwegs.

Tabelle 52: Beschreibung der Struktur ***LegAttributeStructure***.

7.7. Trias_FacilitySupport

In der XML-Schema-Definition Trias_FacilitySupport.xsd werden Strukturdefinitionen aus dem SIRI-FM-Dienst zur Verfügung gestellt, die für die Übermittlung von Meldungen zu Infrastruktureinrichtungen und Fahrzeugausstattungen verwendet werden können. Die hier definierten Strukturen haben den Sinn, den Import des SIRI-Schemas an einer Stelle zu kapseln und eine Abstraktionsebene zu schaffen, die eine Erweiterung erlaubt, ohne dass die SIRI-Definitionen verändert werden müssten.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_FacilitySupport definiert sind.

7.7.1. siri:CommonFacilityGroup

Die Gruppe *CommonFacilityGroup* wird in SIRI in der Schemadatei siri_facilities-v1.2.xsd definiert. Sie ist hier nur aus Gründen der Vollständigkeit und leichteren Verständlichkeit wiedergegeben.

siri:CommonFacilityGroup			*+Group*	**Klassifizierung von allgemeinen Ausstattungsmerkmalen und Einrichtungen (nach TPEG pti_table 23).**
	FareClassFacility	0:*	*unknown | firstClass | secondClass | thirdClass | economyClass | businessClass*	Tarifklassen.
	TicketingFacility	0:*	*unknown | ticketMachines | ticketOffice | ticketOnDemandMachines | ticketSales | mobileTicketing | ticketCollection | centralReservations | localTickets | nationalTickets | internationalTickets*	Einrichtungen zum Fahrscheinkauf.
	NuisanceFacility	0:*	*unknown | smoking | noSmoking | mobilePhoneUseZone | mobilePhoneFreeZone*	Aufenthaltsbereiche.
	MobilityFacility	0:*	*unknown | suitableForWheelChairs | lowFloor | boardingAssistance | stepFreeAccess | tactilePatformEdges | onboardAssistance | unaccompa-*	Ausstattungsmerkmale für Mobilitätseingeschränkte.

			niedMinorAs-sistance | audioInforma-tion | visualIn-formation | displaysForVi-suallyImpaired | audio-ForHea-ringImpaired	
	PassengerInformation-Facility	0:*	*unknown | nextStopIndi-cator | sto-pAnnounce-ments | pas-sengerInfor-mationDisplay | audioInfor-mation | visua-lInformation | tactilePlatfor-mEdges | tactileInforma-tion | wal-kingGuidance | journeyPlan-ning | lost-Found | infor-mationDesk | interactiveKi-osk-Display | printedPublicN otice*	Einrichtungen zur Fahrgastinformation.
	PassengerCommsFacili-ty	0:*	*unknown | faccomms_1 | passengerWifi | telephone | audioServices | vide-oServices | businessSer-vices | internet | postoffice | letterbox*	Kommunikationseinrichtungen für Fahrgäste.
	RefreshmentFacility	0:*	*unknown | restau-rantService | snacksService | trolley | bar | foodNotAvail-able | bever-agesNotAvail-able | bistro | foodVending-Machine | bever-ageVending-Machine*	Versorgung mit Erfrischungen, Speisen und Getränken.
	AccessFacility	0:*	*unknown | lift | escalator | travelator | ramp | stairs | shuttle | nar-rowEntrance | barrier | pal-letAc-cess_lowFloor | validator*	Zugangsmerkmale zu Haltestellen oder Fahrzeugen.
	SanitaryFacility	0:*	*unknown | toilet | noToilet | shower | wheel-chairAcccessT*	Sanitäre Einrichtungen.

			oilet | baby-Change	
	LuggageFacility	0:*	*unknown | bikeCarriage | baggageStor-age | leftLug-gage | porter-age | bag-gageTrolleys*	Einrichtungen zur Gepäckbeförderung oder - aufbe-wahrung.

Tabelle 53: Beschreibung der Gruppe ***siri:CommonFacilityGroup***.

7.7.2. siri:StopFacilityGroup

Die Gruppe *StopFacilityGroup* wird in SIRI in der Schemadatei siri_facilities-v1.2.xsd definiert. Sie ist hier nur aus Gründen der Vollständigkeit und leichteren Verständlichkeit wiedergegeben.

siri:StopFacilityGroup			*+Group*	**Klassifizierung von Ausstattungsmerkmalen und Einrichtungen an Haltestellen (nach TPEG pti_table 23).**
CommonFacilityGroup	*:::*	0:*	*siri:CommonFacilityGroup*	Allgemeine Ausstattungsmerkmale. Vgl. 7.7.1.
	AssistanceFacility	0:*	*unknown | police | firstAid | sosPoint | specificAssistance | unaccompanied-MinorAssistance | boardingAssistance*	Einrichtungen für Hilfesuchende.
	HireFacility	0:*	*unknown | carHire | motorCycleHire | cycleHire | taxi | recreation-DeviceHire*	Vermietungs- und Leihangebote.

Tabelle 54: Beschreibung der Gruppe ***siri:StopFacilityGroup***.

7.7.3. siri:ServiceFacilityGroup

Die Gruppe *ServiceFacilityGroup* wird in SIRI in der Schemadatei siri_facilities-v1.2.xsd definiert. Sie ist hier nur aus Gründen der Vollständigkeit und leichteren Verständlichkeit wiedergegeben.

siri:ServiceFacilityGroup			*+Group*	**Klassifizierung von Ausstattungsmerkmalen und Einrichtungen in Fahrzeugen (nach TPEG pti_table 23).**
CommonFacilityGroup	*:::*	0:*	*siri:CommonFacilityGroup*	Allgemeine Ausstattungsmerkmale. Vgl. 7.7.1.
	AccommodationFacility	0:*	*unknown | sleeper | couchette | specialSeating | freeSeating | recliningSeats | babyCompartment | familyCarriage*	Abteilarten.

Tabelle 55: Beschreibung der Gruppe ***siri:ServiceFacilityGroup***.

7.7.4. siri:AllFacilitiesGroup

Die Gruppe *AllFacilitiesGroup* wird in SIRI in der Schemadatei siri_facilities-v1.2.xsd definiert. Sie ist hier nur aus Gründen der Vollständigkeit und leichteren Verständlichkeit wiedergegeben.

siri:AllFacilitiesGroup			*+Group*	**Umfassende Gruppe mit allen Klassifizierungen von Ausstattungsmerkmalen und Einrichtungen (nach TPEG pti_table 23).**
ServiceFacilityGroup	*:::*	0:*	*siri:ServiceFacilityGroup*	Ausstattungsmerkmale von Fahrzeugen. Vgl.7.7.3.
	AssistanceFacility	0:*	*unknown | police | firstAid | sosPoint | specificAssistance | unaccompanied-MinorAssistance | boardingAssistance*	Einrichtungen für Hilfesuchende.
	HireFacility	0:*	*unknown | carHire | motorCycleHire | cycleHire | taxi | recreation-DeviceHire*	Vermietungs- und Leihangebote.

Tabelle 56: Beschreibung der Gruppe ***siri:AllFacilitiesGroup***.

7.8. Trias_SituationSupport

In der XML-Schema-Definition Trias_SituationSupport.xsd werden Strukturdefinitionen aus dem SIRI-SX-Dienst zur Verfügung gestellt, die für die Übermittlung von Störungs- und Ereignismeldungen verwendet werden können. Die hier definierten Strukturen haben den Sinn, den Import des SIRI-Schemas an einer Stelle zu kapseln und eine Abstraktionsebene zu schaffen, die eine Erweiterung erlaubt, ohne dass die SIRI-Definitionen verändert werden müssten.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_SituationSupport definiert sind.

7.8.1. SituationsStructure

SituationsStructure			*+Structure*	**Container für die strukturierte Beschreibung einer Situation im ÖV oder im Straßenverkehr, wie z. B. einer Störung im ÖV oder auf der Straße oder eines Ereignisses mit Auswirkungen auf das Verkehrsgeschehen.**
	PtSituation	0:*	*siri:PtSituationElement*	Kapselung der SIRI-Strukturdefinition für ÖV-Ereignisse, vgl. (CEN, TS 15531 Part 5).
	RoadSituation	0:*	*siri:RoadSituationElement*	Kapselung der SIRI-Strukturdefinition für IV-Ereignisse, vgl. (CEN, TS 15531 Part 5).

Tabelle 57: Beschreibung der Struktur ***SituationsStructure***.

7.8.2. SituationFullRefStructure

SituationFullRefStructure			*+Structure (abgeleitet von siri:SituationFullRefStructure)*	**Referenz auf eine Situationsbeschreibung.**
SituationFullIdentity	*VersionCountryRef*	0:1	*ifopt:CountryRef*	Referenziert das Land, um ggf. die *ParticipantRef* eindeutig zu machen.
	ParticipantRef	**1:1**	*ParticipantRef*	Eindeutige ID des Schnittstellenpartners (vgl. 5.12). Stellt Namensraum für die ID der Situation bereit.
	SituationNumber	**1:1**	*EntryQualifier*	Eindeutige ID der Situation.
SituationUpdateIdentity	*VersionCountryRef*	0:1	*ifopt:CountryRef*	Referenziert das Land, um ggf. die *ParticipantRef* eindeutig zu machen.
	UpdateParticipantRef	0:1	*ParticipantRef*	Eindeutige ID des Schnittstellenpartners (vgl. 5.12). Stellt Namensraum für die ID der Situation bereit.
	Version	0:1	*SituationVersion*	Versionsnummer des Updates zur Situation. Kann bei der Erstmeldung entfallen.

Tabelle 58: Beschreibung der Struktur ***SituationFullRefStructure***.

7.9. Trias_RequestSupport

In der XML-Schema-Definition Trias_RequestSupport.xsd werden eine Reihe von Basistypen und -strukturen definiert, die die Aufgabe haben, die SIRI-Nachrichtenaustauschverfahren für die TRIAS-Dienste nutzbar zu machen.

7.9.1. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

DataVersionType	*xs:NMTOKEN*	Datentyp zur Angabe der Datenversion.
CalcTimeType	*xs:integer*	Datentyp für die Rechenzeit in Millisekunden.
SignatureType	*xs:string*	Datentyp für Signaturen.
CertificateIdType	*xs:NMTOKEN*	Datentyp für Zertifikat-IDs.

Tabelle 59: Liste der einfachen Typdefinitionen in Trias_RequestSupport.xsd.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_RequestSupport definiert sind.

7.9.2. AbstractTriasServiceRequestStructure

AbstractTriasServiceRequestStructure			*+Structure*	**Basisstruktur für alle direkten Anfragen (ohne Abonnement)**
siri:ContextualisedRequest	*ServiceRequestContext*	0:1	*+siri:ServiceRequestContext*	Allgemeine Nachrichteneigenschaften, die üblicherweise durch die Konfiguration bekannt sind und nicht je Anfrage ausgetauscht werden müssen. Siehe auch (CEN, TS 15531, Part 2), Kap.6.1.2.
	RequestTimestamp	**1:1**	*xs:dateTime*	Zeitstempel der Anfrage.
RequestorEndpoint	*Address*	0:1	*siri:EndpointAddress*	Adresse, an die die Antwort gesendet werden soll. Kann auch mittels RequestorRef aus der Konfiguration ermittelt werden.
	RequestorRef	**1:1**	*→siri:ParticipantCode*	ID des Anfragers.
	MessageIdentifier	0:1	*siri:MessageQualifier*	Beliebige, eindeutige ID, mit der diese Nachricht referenziert werden kann.
ServiceRequestContext	*DataVersion*	0:1	*DataVersion*	Datenversion, die vom Server bei der Bearbeitung benutzt werden soll.
	Language	0:1	*xs:language*	Bevorzugte Sprache, in der Texte in der Antwort zurückgegeben werden sollen.
MessageIntegrityProperties	*Signature*	0:1	*Signature*	Signatur der Nachricht.
	CertificateId	0:1	*CertificateId*	Zertifikat-ID für die Überprüfung der Nachricht.
ServiceRequestContext	*Extension*	0:1	*xs:anyType*	Erweiterungen.

Tabelle 60: Beschreibung der Struktur ***AbstractTriasServiceRequestStructure***.

7.9.3. AbstractTriasSubscriptionRequestStructure

AbstractTriasSubscriptionRequestStructure			*+Structure*	**Basisstruktur für alle Anfragen zur Einrichtung eines Abonnements.**
siri:AbstractSubscriptionRequest	***RequestTimestamp***	**1:1**	*xs:dateTime*	Zeitstempel der Anfrage.
RequestorEndpoint	*Address*	0:1	*siri:EndpointAddress*	Adresse, an die die Antwort gesendet werden soll. Kann auch mittels RequestorRef aus der Konfiguration ermittelt werden.
	RequestorRef	**1:1**	*→siri:ParticipantCode*	ID des Anfragers.
	MessageIdentifier	0:1	*siri:MessageQualifier*	Beliebige, eindeutige ID, mit der diese Nachricht referenziert werden kann.
SubscriberEndpoint	*ConsumerAddress*	0:1	*siri:EndpointAddress*	Adresse, an die die im Rahmen des Abonnements entstandenen Nachrichten gesendet werden sollen. Diese Angabe kann entfallen, falls ConsumerAddress identisch zu RequestorEndpoint:Address.
	SubscriptionFilterIdentifier	0:1	*xs:NMTOKEN*	ID eines vorkonfigurierten Filters, dem die Nachrichten für dieses Abonnement unterzogen werden sollen.
siri:AbstractSubscriptionRequest	*SubscriptionContext*	0:1	*siri:SubscriptionContext*	Allgemeine Abonnementeigenschaften, die üblicherweise durch die Konfiguration bekannt sind und nicht explizit angegeben werden müssen. Siehe auch (CEN, TS 15531, Part 2), Kap.7.1.1.2.
SubscriptionRequestContext	*DataVersion*	0:1	*DataVersion*	Datenversion, die vom Server bei der Bearbeitung benutzt werden soll.
	Language	0:1	*xs:language*	Bevorzugte Sprache, in der Texte in der Antwort zurückgegeben werden sollen.
MessageIntegrityProperties	*Signature*	0:1	*Signature*	Signatur der Nachricht.
	CertificateId	0:1	*CertificateId*	Zertifikat-ID für die Überprüfung der Nachricht.
SubscriptionRequestContext	*Extension*	0:1	*xs:anyType*	Erweiterungen.

Tabelle 61: Beschreibung der Struktur ***AbstractTriasSubscriptionRequestStructure***.

7.9.4. AbstractTriasResponseStructure

AbstractTriasResponseStructure			*+Structure*	**Basisstruktur für alle Antworten.**
si-ri:Producer-Response	***RequestTimestamp***	**1:1**	*xs:dateTime*	Zeitstempel der Antwort.
si-ri:Producer-ResponseEndpoint	*ProducerRef*	*1:1*	*→si-ri:ParticipantCode*	ID des antwortenden Teilnehmers.
	Address	0:1	*si-ri:EndpointAddress*	Adresse, an die eine etwaige Empfangsbestätigung für den Erhalt der Nachricht gesendet werden soll. Kann auch mittels RequestorRef aus der Konfiguration ermittelt werden.
	ResponseMessageIdentifier	0:1	*si-ri:MessageQualifier*	Beliebige, eindeutige ID, mit der diese Nachricht referenziert werden kann.
	RequestMessageRef	0:1	*→si-ri:MessageQualifier*	Referenz auf die Anfragenachricht, die diese Antwortnachricht ausgelöst hat.
ResponseStatus	*Status*	0:1	*xs:boolean*	Indikator, ob die gesamte Anfrage komplett erfolgreich bearbeitet werden konnte. Default ist *true*.
	ErrorCondition	0:1	*si-ri:ErrorCondition*	SIRI-Fehlerzustände, die die Bearbeitung der Anfrage als Ganzes betreffen. Siehe auch (CEN, TS 15531, Part 2), Kap. 5.7.
	MoreData	0:1	*xs:boolean*	Indikator, ob noch weitere Aktualisierungen vorliegen, die abgerufen werden könnten. Default ist *false*.
Service-Response-Context	*DataVersion*	0:1	*DataVersion*	Datenversion, die vom Server bei der Bearbeitung benutzt wurde.
	Language	0:1	*xs:language*	Sprache, in der Texte in der Antwort zurückgegeben wurden.
	CalcTime	0:1	*CalcTime*	Rechenzeit für die Bearbeitung der Anfrage.
MessageIntegrityProperties	*Signature*	0:1	*Signature*	Signatur der Nachricht.
	CertificateId	0:1	*CertificateId*	Zertifikat-ID für die Überprüfung der Nachricht.
Service-Response-Context	*Extension*	0:1	*xs:anyType*	Erweiterungen.

Tabelle 62: Beschreibung der Struktur ***AbstractTriasResponseStructure***.

7.10. Trias_FaresSupport

In der XML-Schema-Definition Trias_FaresSupport.xsd werden eine Reihe von Basistypen und -strukturen definiert, die für die Tarifberechnung einer Reise verwendet werden.

7.10.1. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

FaresAuthorityCodeType	*xs:NMTOKEN*	Code für ein Tarifgebiet oder einen Unternehmens-tarif, z. B. „VVS“ oder „DBAG“.
FareZoneCodeType	*xs:NMTOKEN*	Code für eine Tarifzone in einem Tarifgebiet oder einem Unternehmenstarif.
TicketCodeType	*xs:NMTOKEN*	Code für ein Ticket. Eindeutig innerhalb eines Tarif-gebiets oder eines Unternehmenstarifs.
TravellerCardCodeType	*xs:NMTOKEN*	Code für eine Vielfahrerkarte, z. B. „BahnCard50“ oder „BahnCard25First“.
TravelClassEnumeration	*all* | *first* | *second* | *third* | *business* | *economy*	Reiseklasse.
VatRateEnumeration	*no* | *full* | *half* | *mixed* | *unknown*	Aufzählung möglicher Mehrwertsteuersätze.
PassengerCategoryEnumeration	*Adult* | *Child* | *Senior* | *Youth* | *Disabled*	Kategorisierung von Fahrgästen in tariflicher Hin-sicht.

Tabelle 63: Liste der einfachen Typdefinitionen in Trias_FaresSupport.xsd.

Damit die Codes von Tarifgebieten, Tarifzonen etc. systemübergreifend verstanden werden können, sind bestimmte Vereinbarungen zu treffen. Diese sind in Kapitel 5 beschrieben.

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_FaresSupport definiert sind.

7.10.2. FareZoneStructure

FareZoneStructure			*+Structure*	**Modell einer Tarifzone mit öffentlich bekannter Bezeichnung.**
	FareZoneRef	**1:1**	*→FareZoneCode*	Code für eine Tarifzone (vgl. 7.10.1).
	FareZoneText	**1:1**	*xs:string*	Bezeichnung der Tarifzone für die Fahrgäste.

Tabelle 64: Beschreibung der Struktur ***FareZoneStructure***.

7.10.3. FareZoneListInAreaStructure

FareZoneListInAreaStructure			*+Structure*	**Liste von Tarifzonen bezogen auf ein Tarifgebiet.**
FaresAu thority	***FaresAuthorityRef***	**1:1**	*→FaresAuthori-tyCode*	Code für ein Tarifgebiet oder einen Unternehmens-tarif (vgl. 7.10.1).
	FaresAuthorityText	**1:1**	*xs:string*	Beschreibung oder Name des Tarifgebiets.
	FareZone	**1:***	*+FareZone*	Eine oder mehrere Tarifzonen (vgl.7.10.2).

Tabelle 65: Beschreibung der Struktur ***FareZoneListInAreaStructure***.

7.10.4. BookingInfoStructure

BookingInfoStructure			*+Structure*	**Beschreibung einer Buchungsmöglichkeit für das angefragte Objekt.**
	BookingAgencyName	0:1	*+InternationalText*	Name der Buchungsagentur (Vertragspartner).
	BookingUrl	0:1	*xs:anyURI*	URL für Online-Buchung.
	InfoUrl	0:1	*xs:anyURI*	URL zu Informationsseiten.
	PhoneNumber	0:1	*PhoneNumber*	Telefonnummer zum Buchen (vgl. 7.2.1).
	BookingDeadline	0:1	*xs:duration*	Minimale Vorlaufzeit für Buchungen, bevor die Fahrt beginnt.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 66: Beschreibung der Struktur ***BookingInfoStructure***.

7.10.5. TicketStructure

TicketStructure			*+Structure*	**Modellierung eines Fahrscheins und zugehöriger Informationen.**
	TicketId	**1:1**	*→TicketCode*	Eindeutige Ticket-ID (vgl. 7.10.1).
	TicketName	**1:1**	*xs:string*	Name/Bezeichnung des Tickets.
FaresAuthority	***FaresAuthorityRef***	**1:1**	*→FaresAuthorityCode*	Code für ein Tarifgebiet oder einen Unternehmenstarif (vgl. 7.10.1).
	FaresAuthorityText	**1:1**	*xs:string*	Beschreibung oder Name des Tarifgebiets.
Ticket-Price	*Price*	0:1	*xs:decimal*	Ticketpreis als Dezimalzahl.
	NetPrice	0:1	*xs:decimal*	Netto-Ticketpreis als Dezimalzahl für Abrechnungszwecke.
	Currency	0:1	*xs:NMTOKEN*	Währungscode nach ISO 4217, z. B. „EUR“ oder „GBP“.
	VatRate	0:1	*VatRateEnumeration*	Mehrwertsteuersatz (vgl. 7.10.1). Voreinstellung ist *unknown*.
TicketValidity	*TravelClass*	0:1	*TravelClassEnumeration*	Reiseklasse, für die das Ticket gültig ist (vgl. 7.10.1).
	RequiredCard	0:*	*→TravellerCardCode*	Eine oder mehrere Vielfahrerkarten, die nötig sind, um dieses Ticket erwerben oder nutzen zu dürfen (vgl. 7.10.1).
	ValidFor	0:*	*PassengerCategoryEnumeration*	Personengruppen, die dieses Ticket nutzen dürfen (vgl. 7.10.1).
	ValidityDuration	0:1	*xs:duration*	Maximale zeitliche Gültigkeit des Tickets ab Kauf oder Entwertung.
	ValidityDurationText	0:1	*+InternationalText*	Beschreibung der zeitlichen Gültigkeit.
	ValidityFareZones	0:1	*+FareZoneListInArea*	Räumliche Gültigkeit des Tickets ausgedrückt durch eine Liste der Tarifzonen, für die das Ticket gilt.
	ValidityAreaText	0:1	*+InternationalText*	Beschreibung der räumlichen Gültigkeit.
Ticket-Booking	*InfoUrl*	0:*	*+WebLink*	URL zu Informationsseiten für dieses Ticket (vgl. 7.2.4).
	SaleUrl	0:*	*+WebLink*	URL zu Online-Kaufmöglichkeiten, um dieses Ticket zu erwerben (vgl. 7.2.4).
	BookingInfo	0:*	*+BookingInfo*	Beschreibung der Buchungsmöglichkeiten (vgl. 7.10.4).
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 67: Beschreibung der Struktur ***TicketStructure***.

7.10.6. TripFaresResultStructure

TripFaresResultStructure			*+Structure*	**Fasst die Ergebnisdaten für die Tarifauskunft zu einer Verbindung (oder zu Teilen einer Verbindung) zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf diese Tarifauskunft. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
TripLeg Range	*FromTripLegIdRef*	0:1	*xs:NMTOKEN*	Referenz auf einen Teilweg der Verbindung als Beginn der Gültigkeit dieser Tarifauskunft.
	ToTripLegIdRef	0:1	*xs:NMTOKEN*	Referenz auf einen Teilweg der Verbindung als Ende der Gültigkeit dieser Tarifauskunft.
	PassedZones	0:1	*+FareZoneList InArea*	Die durchfahrenen Tarifzonen auf diesem Abschnitt der Verbindung (vgl. 7.10.3).
	Ticket	0:*	*+Ticket*	Fahrscheine, die auf diesem Abschnitt der Verbindung gültig sind (vgl. 7.10.5).
	StaticInfoURL	0:*	*+WebLink*	URL zu Informationsseiten (vgl. 7.2.4).

Tabelle 68: Beschreibung der Struktur ***TripFaresResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

FARES_OUTOFAREA	Der in der Verbindungsauskunft gefundene Weg verlässt das Tarifgebiet.
FARES_JOURNEYNOTPERMITTED	Ein in der Verbindungsauskunft benutztes Verkehrsmittel ist für den Tarif nicht zulässig.
FARES_ADDITIONALCHARGES	Zusätzliche Gebühren sind voraussichtlich zu entrichten (z. B. Mautzuschläge oder Reservierungsgebühren).
FARES_ADDITIONALTICKETS	Zusätzliche Fahrscheine sind notwendig, da nicht für alle Verkehrsmittel ein passender Fahrschein ermittelt werden konnte.
FARES_ROUTENOTFEASIBLE	Es kann kein Fahrschein ermittelt werden, weil die Route der Verbindungsauskunft nicht in Einklang mit den Tarifregeln steht (z. B. wegen Rundreisen, Stichfahrten oder Überschreitung der zulässigen Gesamtdauer).
FARES_ALREADYCOVERED	Diese Verbindung (oder der in *TripLegRange* angegebene Teil davon) kann mit der Fahrberechtigung benutzt werden, die in der Anfrage mitgegeben wurde.

Tabelle 69: Liste der Fehlerzustände in ***TripFaresResult***.

7.10.7. FaresPassengerStructure

FaresPassengerStructure				*+Structure*	**Profil eines Reisenden für die Tarifermittlung.**
	a	***Age***	**-1:1**	*xs:nonNegativeInteger*	Alter des Reisenden.
	b	***PassengerCategory***		*PassengerCategoryEnumeration*	Personengruppe, die der Reisende zugerechnet werden kann (vgl. 7.10.1).
	TravellerCard		0:*	*→TravellerCardCode*	Eine oder mehrere Vielfahrerkarten, die der Reisende erworben hat und verwenden kann (vgl. 7.10.1).
	a	*ZonesAlreadyPaid*	-0:1	*+FareZoneList InArea*	Liste von Tarifzonen, für die der Fahrgast bereits ein gültiges Ticket hat (vgl. 7.10.3).
	b	*OwnedTicket*	-0:*	*→TicketCode*	Eine oder mehrere Ids von Tickets, die der Fahrgast schon erworben hat und die der Fahrgast für diese Reise (oder zumindest für Teile davon) benutzen kann.

Tabelle 70: Beschreibung der Struktur ***FaresPassengerStructure***.

Durch die Angabe der Elemente ZonesAlreadyPaid bzw. OwnedTicket kann zum Ausdruck gebracht werden, dass der Fahrgast bereits Fahrberechtigungen z.B. in Form von Fahrscheinen (wie etwa einer Monatskarte oder eines Jobtickets) besitzt. Der Server soll dann versuchen festzustellen, ob diese Berechtigungen bereits ausreichen, um auch diese Verbindung benutzen zu dürfen, oder im anderen Fall Tickets für die verbleibenden Teile empfehlen, für die die vorhandenen Berechtigungen nicht ausreichen. Für die Teile der Verbindung, für die kein weiterer Fahrschein notwendig ist, wird in *TripFaresResult* (vgl. 7.10.6) der Fehlerzustand *FARES_ALREADYCOVERED* (aus Tabelle 69) angegeben.

7.10.8. FaresParamStructure

FaresParamStructure			*+Structure*	**Parameter für die Tarifermittlung.**
Fares-DataFil-ter	*FareAuthorityFilter*	0:*	*→FaresAutho-rityCode*	Codes für Tarifgebiete oder Unternehmenstarife, die berücksichtigt werden sollen (vgl. 7.10.1).
	PassengerCategory	0:*	*PassengerCa-tegoryEnume-ration*	Personengruppen, die berücksichtigt werden sollen (vgl. 7.10.1).
	TravelClass	0:1	*TravelClas-sEnumeration*	Reiseklasse, die berücksichtigt werden soll (vgl. 7.10.1).
	Traveller	0:*	*+FaresPassen ger*	Anzahl Reisende, für die die Tarifermittlung ausgeführt werden soll (vgl. 7.10.7).

Tabelle 71: Beschreibung der Struktur ***FaresParamStructure***.

8. Ortsinformationsdienst

8.1. Beschreibung

Der Ortsinformationsdienst umfasst vier Funktionalitäten, die in der ursprünglichen Dienstebeschreibung noch als getrennte Dienste aufgefasst wurden

- Start-/Ziel-Identifikation bei Eingabe einer Zeichenkette,
- Objektinformationsdienst zum Abrufen aller Ortsobjekte,
- Geografischer Kontextdienst zum Abrufen von Ortsobjekten in einem Kartenausschnitt,
- Koordinaten-zu-Adressdienst zum Abrufen der nächsten Adresse für gegebene Koordinaten.

Diese Funktionalitäten konnten durch Abstraktion in einem einzigen Dienst gebündelt werden. Durch diese Verallgemeinerung sind auch weitere Einsatzmöglichkeiten des Dienstes entstanden. Beispielsweise (aber nicht abschließend):

- Abruf der nächsten Haltestelle(n) für gegebene Koordinaten.
- Ortsabhängiges Patternmatching einer Zeichenkette durch Berücksichtigung von gleichzeitig übergebenen Koordinaten.

In der XML-Schema-Definition Trias_Locations.xsd werden Datentypen und Strukturen definiert, die für den Ortsinformationsdienst verwendet werden.

8.2. Einfache Datentypen

Folgende einfache Typen (simple types) werden definiert:

LocationTypeEnumeration	*stop* | *address* | *poi* | *coord* | *locality*	Typ eines Ortsobjektes.
LocationUsageEnumeration	*origin* | *destination* | *via*	Verwendungszweck eines Ortsobjektes.

Tabelle 72: Liste der einfachen Typdefinitionen in Trias_Locations.xsd.

8.3. Anfragestrukturen

Ortsobjekte werden mittels eines Elements ***LocationInformationRequest*** vom Typ *LocationInformationRequestStructure* angefordert.

8.3.1. LocationInformationRequestStructure

LocationInformationRequestStructure				*+Structure*	**Fasst die Daten Ortsobjektanfrage zusammen.**
	a	***InitialInput***	**-1:1**	*+InitialLocationInput*	Eingabedaten für eine initiale Ortsinformationsanfrage. Vgl. 8.3.2.
	b	***LocationRef***		*+LocationRef*	Referenz auf ein Ortsobjekt, welches weiter verfeinert werden soll. Bei hierarchisch organisierten Ortsobjekten kann es sinnvoll sein, die Ortsidentfikation in mehreren Stufen durchzuführen. Dabei erzeugt eine initiale Anfrage an den Ortsinformationsdienst eine Menge von „groben“ Ortsobjekten (z. B. Straßen), die ggf. noch weiter verfeinert werden müssen (z. B. zu Hausnummernbereichen). Die „groben“ Objekte werden dem Benutzer gezeigt und er wählt eines davon aus. Um dieses nun weiter zu verfeinern, wird seine Referenz hier dem Ortsinformationsdienst übergeben. Vgl. 7.5.11.
	Restrictions		0:1	*+LocationParam*	Weitere Anfrageparameter. Vgl. 8.3.7.
	Extension		0:1	*xs:anyType*	Erweiterungen.

Tabelle 73: Beschreibung der Struktur ***LocationInformationRequestStructure***.

8.3.2. InitialLocationInputStructure

InitialLocationInputStructure			*+Structure*	**Fasst die Anfrageparameter zusammen, die eine initiale Suche nach Ortsobjekten benötigt.**
	LocationName	0:1	*xs:string*	Eingabezeichenkette, die als Muster für die zu findenden Ortsobjekte dienen soll. Falls angegeben, sollen Ortsobjekte umso mehr bevorzugt werden, je ähnlicher ihr Name der Zeichenkette ist. Falls gleichzeitig *GeoPosition* angegeben wird, muss der Dienst beide Anforderungen sinnvoll zueinander gewichten.
	GeoPosition	0:1	*+GeoPosition*	Geografische Position, in dessen Nähe die zu findenden Ortsobjekte liegen sollen. Falls angegeben, sollen solche Ortsobjekte bevorzugt werden, die in der Nähe dieser Geoposition liegen. Falls gleichzeitig *LocationName* angegeben wird, muss der Dienst beide Anforderungen sinnvoll zueinander gewichten. Vgl. 7.2.3.
	GeoRestriction	0:1	*+GeoRestrictions*	Geografischer Filter. Falls angegeben, müssen alle gefundenene Ortsobjekte diesem Filter entsprechen. Vgl. 8.3.3.

Tabelle 74: Beschreibung der Struktur ***InitialLocationInputStructure***.

8.3.3. GeoRestrictionsStructure

GeoRestrictionsStructure				*+Structure*	**Definiert einen geografischen Filter.**
	a	***Circle***	**-1:1**	*+GeoCircle*	Der Filter wird durch einen Kreis definiert. Vgl. 8.3.4.
	b	***Rectangle***	**-1:1**	*+GeoRectangle*	Der Filter wird durch ein Rechteck definiert. Vgl. 8.3.5.
	c	***Area***	**-1:1**	*+GeoArea*	Der Filter wird durch ein Polygon definiert. Vgl. 8.3.6.

Tabelle 75: Beschreibung der Struktur ***GeoRestrictionsStructure***.

8.3.4. GeoCircleStructure

GeoCircleStructure			*+Structure*	**Definiert einen geografischen Kreis.**
	Center	**1:1**	*+GeoPosition*	Zentrum des Kreises. Vgl. 7.2.3.
	Radius	**1:1**	*Distance*	Radius des Kreises in Metern.

Tabelle 76: Beschreibung der Struktur ***GeoCircleStructure***.

8.3.5. GeoRectangleStructure

GeoRectangleStructure			*+Structure*	**Definiert ein geografisches Rechteck.**
	UpperLeft	**1:1**	*+GeoPosition*	Linke obere Ecke des Rechtecks. Vgl. 7.2.3.
	LowerRight	**1:1**	*+GeoPosition*	Rechte untere Ecke des Rechtecks. Vgl. 7.2.3.

Tabelle 77: Beschreibung der Struktur ***GeoRectangleStructure***.

8.3.6. GeoAreaStructure

GeoAreaStructure			*+Structure*	**Definiert ein geografisches Polygon.**
	PolylinePoint	**3:***	*+GeoPosition*	Eckpunkte des Polygons. Vgl. 7.2.3.

Tabelle 78: Beschreibung der Struktur ***GeoAreaStructure***

8.3.7. LocationParamStructure

LocationParamStructure			*+Structure*	**Fasst Anfrageparameter zusammen, die im Ortsinformationsdienst verwendet werden.**
LocationDataFilter	*Type*	0:*	*stop* | *address* | *poi* | *coord* | *locality*	Erlaubte Ortsobjektstypen. Falls welche angegeben werden, dürfen nur Ortsobjekte zurückgegeben werden, die von einem der angegebenen Typen sind. Falls keine angegeben werden, sind alle Objekttypen erlaubt.
	Usage	0:1	*origin* | *destination* | *via*	Verwendung des Ortsobjektes. Falls angegeben, teilt dies dem Dienst mit, als was das gesuchte Ortsobjekt verwendet werden soll. Der Ortsinformationsdienst darf dann nur Objekte zurückgeben, die für die angegebene Verwendung freigegeben sind.
	PtModes	0:1	*+PtModeFilter*	Erlaubte Verkehrsmittel. Falls angegeben, dürfen nur solche Ortsobjekte zurückgegeben werden, an denen Verkehre fahren, die dem Filter entsprechen. Dies schließt automatisch alle Nicht-Haltestellen aus. Vgl. 7.3.5.
	OperatorFilter	0:1	*+OperatorFilter*	Die Suche wird auf solche Ortsobjekte eingeschränkt, die von bestimmten Unternehmen betrieben/nicht betrieben werden (vgl. 7.4.4).
	LocalityRef	0:*	*→LocalityCode*	Erlaubte Lokalitäten. Falls angegeben, dürfen nur solche Ortsobjekte zurückgegeben werden, die mindestens einer der gegebenen Lokalitäten zugeordnet sind. Vgl. 7.5.1.
	PointOfInterestFilter	0:1	*+PointOfInterestFilter*	Ermöglicht, eine POI-Suche auf bestimmte POI-Kategorien einzuschränken (vgl. 7.5.6).

LocationPolicy	*Language*	0:1	*xs:language*	Bevorzugte Sprache für die zurückgegebenen Texte. Dies muss nicht unbedingt Einfluss auf die Namen der Ortsobjekte haben.
	NumberOfResults	0:1	*xs:positiveInteger*	Anzahl der maximal zurückzugebenen Ortsobjekte. Der Dienst kann durchaus weniger Objekte zurückgeben, falls sinnvoll oder falls sonst der Dienst überfordert wäre. Falls mehr Objekte die Anfrage erfüllen (z. B. wenn alle Objekte abgerufen werden sollen), kann mit diesem Parameter die Menge der Objekte, die in einem Abruf maximal übertragen werden, begrenzt werden. Ein Ortsinformationsdienst muss in der Lage sein, mindestens 500 Ortsobjekte in einer Antwort zurückzugeben.
	ContinueAt	0:1	*xs:nonNegativeInteger*	Falls angegeben, weist dieser Parameter den Dienst an, wieviele Objekte in der Rückgabe übersprungen werden sollen. Falls bei einem Abruf von Ortsobjekten nicht alle passenden Objekte geliefert werden konnten, teilt der Dienst dies in seiner Antwort im Feld *ContinueAt* mit (vgl. 8.4.1). Um die weiteren Objekte abzurufen, wird die Anfrage an den Ortsinformationsdienst exakt wiederholt, wobei dieser Parameter angegeben wird, indem der Wert aus der letzten Diensteantwort eingefüllt wird.
	IncludePtModes	0:1	*xs:boolean*	Teilt dem Dienst mit, an Haltestellen die verfügbaren Verkehrsmittel mit zurück zu geben. Default ist *false*.

Tabelle 79: Beschreibung der Struktur ***LocationParamStructure***.

8.4. Antwortstrukturen

Das Ergebnis einer Objektinformationsanfrage wird mittels eines Elements ***LocationInformationResponse*** vom Typ *LocationInformationResponseStructure* übertragen.

8.4.1. LocationInformationResponseStructure

LocationInformationResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Ortsinformationsanfrage zusammen.**
	ContinueAt	0:1	*xs:nonNegativeInteger*	In einem Folgeabruf zu überspringende Ortsobjekte. Falls gesetzt, zeigt der Dienst an, dass noch weitere Ortsobjekte zu der Anfrage passen, die in der Antwort nicht enthalten sind. Wird der Abruf wiederholt und dabei der Parameter *ContinueAt* auf den hier übermittelten Wert gesetzt (vgl. 8.3.7), liefert der Dienst die folgenden Ortsobjekte.
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	Location	0:*	*+LocationResult*	Gefundene Ortsobjektergebnisse. Die Ortsobjekte müssen nach dem Übereinstimmungsgrad mit den Eingabedaten sortiert sein, d.h. das erste ist das am besten passende Objekt. Vgl. 8.4.2.

Tabelle 80: Beschreibung der Struktur ***LocationInformationResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

LOCATION_NORESULTS	Zu den Eingabedaten konnten keine Ortsobjekte gefunden werden.
LOCATION_UNSUPPORTEDTYPE	Es wurden nur Objekttypen angefragt, die vom Dienst nicht unterstützt werden.
LOCATION_UNSUPPORTEDCOMBINATION	Die verwendete Kombination von Eingabedaten (Zeichenkette, Koordinaten, Geo-Restriktion) wird vom Dienst nicht unterstützt.
LOCATION_NOREFINEMENT	Das angegebene Ortsobjekt konnte nicht verfeinert werden.
LOCATION_USAGEIGNORED	Der Verwendungszweck wurde ignoriert.
LOCATION_UNSUPPORTEDPTMODES	Der Dienst unterstützt keine Einschränkung der Verkehrsmittel.
LOCATION_UNSUPPORTEDLOCALITY	Der Dienst unterstützt keine Einschränkung durch Lokalitäten.

Tabelle 81: Liste der Fehlerzustände in ***LocationInformationResponse***.

8.4.2. LocationResultStructure

LocationResultStructure			*+Structure*	**Resultatstruktur für ein Ortsobjekt.**
	Location	**1:1**	*+Location*	Eigentliches Ortsobjekt. Vgl. 7.5.10.
	Complete	**1:1**	*xs:boolean*	Gibt an, ob das Ortsobjekt schon vollständig ausdifferenziert ist, oder ob es noch verfeinert werden kann.
	Probability	0:1	*xs:float*	Wahrscheinlichkeit, dass dieses Ortsobjekt dem gesuchten entspricht. Wird mit einem Wert zwischen 0 und 1 angegeben.
	Mode	0:*	*+Mode*	Auflistung der Verkehrsmittel, die an dem Ortsobjekt verkehren. Sollte nur bei Haltestellen gefüllt sein und nur dann, wenn es in der Anfrage angefordert wurde. Vgl. 7.3.4.

Tabelle 82: Beschreibung der Struktur ***LocationResultStructure***.

9. Dienst Verbindungsauskunft

9.1. Beschreibung

Dieser Dienst berechnet intermodale Verbindungen von einem Startpunkt zu einem Zielpunkt. Dabei werden diverse Benutzerpräferenzen berücksichtigt.

9.2. Anfragestrukturen

Eine intermodale Verbindungsauskunft wird mittels eines Elements ***TripRequest*** vom Typ *TripRequestStructure* angefordert.

9.2.1. TripRequestStructure

TripRequestStructure			*+Structure*	**Fasst die Anfragedaten für eine Verbindungsauskunft zusammen.**
	Origin	**1:***	*+LocationContext*	Ortsdaten für den Abfahrtsort. Vgl. 7.6.11.
	Destination	**1:***	*+LocationContext*	Ortsdaten für den Zielort. Vgl. 7.6.11.
	Via	0:*	*+Via*	Ein oder mehrere Via-Orte. Die angegebenen Via-Orte müssen in der vorgegebenen Reihenfolge erreicht werden. Der Server darf eine Via-Haltestelle durch eine äquivalente Haltestelle ersetzen. Vgl. 7.6.2.
	NotVia	0:*	*+NotVia*	Haltestellen oder Haltepunkte, die die Verbindung nicht passieren darf. Vgl. 9.2.4.
	NoChangeAt	0:*	*+NoChangeAt*	Haltestellen oder Haltepunkte, wo die Verbindung keinen Umstieg vorsehen darf. Vgl. 9.2.5.
	Params	0:1	*+TripParam*	Parameter, die die Suche und Rückgabewerte beeinflussen können. Vgl. 9.2.2.

Tabelle 83: Beschreibung der Struktur ***TripRequestStructure***.

Die Elemente *Origin* und *Destination* treten in der Regel nur einfach auf. Nur im Fall dass man mehrere Start- oder Zielpunkte, die jeweils eine eigene Abfahrts- bzw. Ankunftszeit mit sich bringen, definieren möchte, sollten mehrere *Origin*- bzw. *Destination*-Elemente übergeben werden. In diesem Fall sucht der Server den für die Gesamtverbindung optimalen Punkt heraus. Die Wahl des optimalen *Origin*- oder *Destination*-Punkts kann von der Uhrzeit abhängen und sich daher mit jeder gefundenen Verbindung ändern.

9.2.2. TripParamStructure

TripParamStructure			*+Structure*	**Fasst die Anfragedaten für eine Verbindungsauskunft zusammen.**
TripDataFilter	*PtModeFilter*	0:1	*+PtModeFilter*	Filter nach Verkehrsmitteltypen. Vgl. 7.3.5.
	LineFilter	0:1	*+LineDirectionFilter*	Erlaubte Linien (ggf. verfeinert auf Richtungen). Vgl. 7.4.6.
	OperatorFilter	0:1	*+OperatorFilter*	Filter nach Verkehrsunternehmen. Vgl.7.4.4.
BaseTripMobilityFilter	*NoSingleStep*	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Stufe bewältigen kann. Falls nein, wird dieser Parameter gesetzt.
	NoStairs	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Treppe bewältigen kann. Falls nein, wird dieser Parameter gesetzt.
	NoEscalator	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Rolltreppe benutzen

				kann. Falls nein, wird dieser Parameter gesetzt.
	NoElevator	0:1	*xs.boolean*	Legt fest, ob der Benutzer einen Aufzug benutzen kann. Falls nein, wird dieser Parameter gesetzt.
	NoRamp	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Rampe bewältigen kann. Falls nein, wird dieser Parameter gesetzt.
TripMobilityFilter	*LevelEntrance*	0:1	*xs.boolean*	Legt fest, ob der Benutzer beim Ein- und Aussteigen in und aus Fahrzeugen einen ebenen Zugang benötigt. Dazu reicht u.U. auch ein Hublift am Fahrzeug oder am Bahnsteig. Falls der ebene Zugang notwendig ist, wird dieser Parameter gesetzt.
	BikeTransport	0:1	*xs.boolean*	Legt fest, ob der Benutzer ein Fahrrad an Bord der Verkehrsmittel mitnehmen will. Falls ja, wird dieser Parameter gesetzt.
	WalkSpeed	0:1	*OpenPercent*	Veränderung der Standardgehgeschwindigkeit in Prozent. Der Wert 100 stellt den Standard dar. Werte kleiner 100 stellen eine langsamere Geschwindigkeit dar, Werte größer 100 eine schnellere.
BaseTripPolicy				
	a *NumberOfResults*	-0:1	*xs:positiveInteger*	Anzahl der Verbindungsauskünfte, die der Benutzer mindestens erwartet.
	b *:::*	-0:1	*NumberOfResultsGroup*	Angabe der gewünschten Verbindungen vor/nach dem angegebenen Zeitpunkt am Start oder Ziel (vgl. 9.2.3).
	IgnoreRealtimeData	0:1	*xs:boolean*	Wenn dieser Parameter gesetzt ist, sollen in der Verbindungssuche keine Echtzeitdaten oder Störungsinformationen sondern nur Sollfahrplandaten berücksichtigt werden.
	ImmediateTripStart	0:1	*xs:boolean*	Wenn dieser Parameter gesetzt ist, soll die zu suchende Verbindung unmittelbar an der angegebenen Startsituation beginnen. Eine Optimierung der Abfahrtszeit am Start nach der Regel „Starte so spät wie möglich, solange nur die gleiche Ankunftszeit am Ziel gewährleistet ist“ ist dann nicht notwendig.
TripPolicy	*InterchangeLimit*	0:1	*xs:positiveInteger*	Anzahl der maximal zugelassenen Umsteigevorgänge.
	AlgorithmType	0:1	*fastest* | *minChanges* | *leastWalking* | *leastCost*	Art der Zielfunktion, nach der der Algorithmus die Verbindung optimieren soll.
	ItModesToCover	0:*	*IndividualModesEnumeration*	Für jeden IV-Typ (vgl. 7.3.1) in dieser Liste soll eine eigene monomodale Verbindung gefunden werden – zusätzlich zu den intermodalen Verbindungen.
BaseTripContentFilter	*IncludeTrackSections*	0:1	*xs:boolean*	Legt fest, ob im Resultat TrackSection-Elemente (vgl. 9.3.13) für die detaillierte geographische Beschreibung des Wegs mitausgegeben werden sollen. Default ist *false*.
	IncludeLegProjection	0:1	*xs:boolean*	Legt fest, ob im Resultat der detaillierte geographische Verlauf des Wegs als Koordinatenfolge mitausgegeben werden soll. Default ist *false*.
	IncludeTurnDescription	0:1	*xs:boolean*	Legt fest, ob im Resultat Routenhinweise mit Abbiegeempfehlungen mitausgegeben werden sollen. Default ist *false*.
	IncludeAccessibility	0:1	*xs:boolean*	Legt fest, ob im Resultat Informationen zur Barrierefreiheit mitausgegeben werden sollen. Default ist *false*.
TripContentFilter	*IncludeIntermediateStops*	0:1	*xs:boolean*	Legt fest, ob im Resultat die Zwischenhalte mitausgegeben werden sollen. Default ist *false*.
	IncludeFares	0:1	*xs:boolean*	Legt fest, ob im Resultat Tarifinformationen mitausgegeben werden sollen. Default ist *false*.
	IncludeOperatingDays	0:1	*xs:boolean*	Legt fest, ob im Resultat Informationen zu den Verkehrstagen mitausgegeben werden sollen. Default ist *false*.

	FaresParam	0:1	*+FaresParam*	Parameter für die Tarifermittlung (vgl. 7.10.8).
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 84: Beschreibung der Struktur ***TripParamStructure***.

9.2.3. NumberOfResultsGroup

NumberOfResultsGroup			*+Group*	**Festlegung der Anzahl gewünschter Verbindungen vor und nach dem angegebenen Zeitpunkt am Start oder Ziel. Diese Gruppe kann nicht verwendet werden, wenn am Start UND am Ziel ein Zeitpunkt vorgeschrieben ist.**
	NumberOfResultsBefore	**1:1**	*xs:positiveInteger*	Anzahl gewünschter Verbindungen vor dem angegebenen Zeitpunkt.
	NumberOfResultsAfter	**1:1**	*xs:positiveInteger*	Anzahl gewünschter Verbindungen nach dem angegebenen Zeitpunkt.

Tabelle 85: Beschreibung der Gruppe ***NumberOfResultsGroup***.

9.2.4. NotViaStructure

NotViaStructure			*+Structure*	**Angaben zu einer Nicht-Via-Bedingung. Diese Art Bedingung verhindert, dass eine Verbindungsauskunft über die angegebene Haltestelle oder Haltepunkt führt.**
	a ***StopPointRef***	**-1:1**	*→StopPoint*	Referenz auf einen Nicht-Via-Haltepunkt. Vgl. 7.5.1.
	b ***StopPlaceRef***		*→StopPlace*	Referenz auf eine Nicht-Via-Haltestelle. Vgl. 7.5.1.

Tabelle 86: Beschreibung der Struktur ***NotViaStructure***.

9.2.5. NoChangeAtStructure

NoChangeAtStructure			*+Structure*	**Angaben zu einer Nicht-Umsteigen-Bedingung. Diese Art Bedingung verhindert, dass in einer Verbindungsauskunft an der angegebenen Haltestelle oder Haltepunkt umgestiegen werden muss.**
	a ***StopPointRef***	**-1:1**	*→StopPoint*	Referenz auf einen Haltepunkt. Vgl. 7.5.1.
	b ***StopPlaceRef***		*→StopPlace*	Referenz auf eine Haltestelle. Vgl. 7.5.1.

Tabelle 87: Beschreibung der Struktur ***NoChangeAtStructure***.

9.3. Antwortstrukturen

Das Ergebnis einer intermodalen Verbindungsanfrage wird mittels eines Elements ***TripResponse*** vom Typ *TripResponseStructure* übertragen.

9.3.1. TripResponseStructure

TripResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine intermodale Verbindungsauskunft zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
	TripResponseContext	0:1	*+TripResponseContext*	Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl. 9.3.2.
	TripResult	0:*	*+TripResult*	Container für eine Verbindungsauskunft. Vgl. 9.3.3.

Tabelle 88: Beschreibung der Struktur ***TripResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

TRIP_NOTRIPFOUND	Zu den angegebenen Start- und Zielorten, der gewünschten Abfahrts- oder Ankunftszeit sowie unter Berücksichtigung der gegebenen Parameter konnte keine Verbindung gefunden werden.
TRIP_ORIGINUNKNOWN	Der angegebene Ort (Adresse, Haltestelle, …) für den Start der Verbindung ist unbekannt.
TRIP_DESTINATIONUNKNOWN	Der angegebene Ort (Adresse, Haltestelle, …) für das Ziel der Verbindung ist unbekannt.
TRIP_VIAUNKNOWN	Einer der angegebenen Via-Punkte ist unbekannt.
TRIP_NOTVIAUNKNOWN	Eine der angegebenen Nicht-Via-Haltestellen ist unbekannt.
TRIP_NOCHANGEATUNKNOWN	Eine der angegebenen Nicht-Umsteigen -Haltestellen ist unbekannt.
TRIP_NOORIGIN	Es wurde kein Startpunkt angegeben.
TRIP_NODESTINATION	Es wurde kein Zielpunkt angegeben.
TRIP_ORIGINDESTINATIONIDENTICAL	Start und Ziel sind gleich.
TRIP_DATETIMEERROR	Datum und/oder Uhrzeit sind unverständlich.
TRIP_DEPARTUREAFTERARRIVAL	Die gewünschte Abfahrtszeit an allen Startpunkten liegt nach der gewünschten Ankunftszeit an allen Zielpunkten.
TRIP_DATEOUTOFRANGE	Für das angefragte Datum liegen keine Fahrplandaten vor.

Tabelle 89: Liste der Fehlerzustände in ***TripResponse***.

9.3.2. TripResponseContextStructure

TripResponseContextStructure	*+Structure (abgeleitet von AbstractResponseContextStructure)*	Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl. 7.6.12.

Tabelle 90: Beschreibung der Struktur ***TripResponseContextStructure***.

9.3.3. TripResultStructure

TripResultStructure			*+Structure*	**Fasst die Ergebnisdaten für eine einzelne intermodale Verbindungsauskunft zusammen.**
	ResultId	**1:1**	*xs:NMTOKEN*	ID des Resultats für spätere Referenzierung bzw. für Debug-Zwecke.
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf dieses Verbindungsresultat. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
	Trip	**1:1**	*+Trip*	Daten zu einer intermodalen Verbindung. Vgl. 9.3.4.
	TripFares	0:*	*+TripFaresResult*	Ticket- und Fahrpreisinformationen zur Verbindung als Ganzes oder zu Teilen der Verbindung (vgl. 7.10.6).

Tabelle 91: Beschreibung der Struktur ***TripResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

TRIP_ORIGINEQUIVALENT	Die gewünschte Starthaltestelle wurde durch eine äquivalente Haltestelle ersetzt.
TRIP_DESTINATIONEQUIVALENT	Die gewünschte Zielhaltestelle wurde durch eine äquivalente Haltestelle ersetzt.
TRIP_VIAEQUIVALENT	Eine gewünschte Via-Haltestelle wurde durch eine äquivalente Haltestelle ersetzt.
TRIP_REALTIMEINCOMPLETE	Für mindestens ein Verkehrsmittel in dieser Verbindung liegen keine Echtzeitdaten vor.
TRIP_ITTIMEEXTENDED	Die vorgegebene maximale Zeit im Individualverkehr (meist Fußweg oder Fahrrad) wurde vom System verlängert, da ansonsten keine Verbindung gefunden werden kann.
TRIP_ITMODECHANGED	Das vorgegebene Individualverkehrsmittel wurde vom System ersetzt, da ansonsten keine Verbindung gefunden werden kann. Üblicherweise ist dies ein Wechsel von Fußweg auf Taxi.
TRIP_INCONVENIENTWAITING	Die Verbindung enthält eine lange Wartezeit.

Tabelle 92: Liste der Fehlerzustände in ***TripResult***.

9.3.4. TripStructure

TripStructure			*+Structure*	**Daten zu einer einzelnen intermodalen Verbindung.**
	TripId	**1:1**	*xs:NMTOKEN*	ID der Verbindung für spätere Referenzierung bzw. für Debug-Zwecke.
	Duration	**1:1**	*xs:duration*	Gesamtdauer der Verbindung.
	StartTime	**1:1**	*xs:dateTime*	Startzeitpunkt der Verbindung.
	EndTime	**1:1**	*xs:dateTime*	Endzeitpunkt der Verbindung.
	Interchanges	**1:1**	*xs:nonNegativeInteger*	Anzahl der notwendigen Umsteigevorgänge.
	Distance	0:1	*Distance*	Gesamtdistanz der Verbindung als Länge des zurückzulegenden Weges.
	TripLeg	**1:***	*+TripLeg*	Teilweg/e dieser Verbindung. Vgl. 9.3.5.
Operating-Days	*OperatingDays*	0:1	*+OperatingDays*	Verkehrstage für diese Verbindung. Vgl. 7.4.8.
	OperatingDaysDescription	0:1	*+InternationalText*	Menschenlesbare Beschreibung der Verkehrstage, z. B. „Montag bis Freitag“ oder „Sonn- und Feiertag“.
	SituationFullRef	0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im *TripResponseContext* (vgl. 9.3.2) zu finden sein oder auf anderem Wege bekannt gemacht werden. Vgl. 7.8.2.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 93: Beschreibung der Struktur ***TripStructure***

9.3.5. TripLegStructure

TripLegStructure				*+Structure*	**Teilweg zu einer Verbindung.**
		LegId	**1:1**	*xs:NMTOKEN*	ID des Teilwegs dieser Verbindung für spätere Referenzierung. Eindeutig innerhalb *TripResult*.
	a	***TimedLeg***	**-1:1**	*+TimedLeg*	Ausprägung des Teilwegs als fahrplanbehafteter Teilweg. Vgl. 9.3.6.
	b	***InterchangeLeg***		*+InterchangeLeg*	Ausprägung des Teilwegs als Umstieg zwischen Verkehrsmitteln. Vgl. 9.3.7.
	c	***ContinuousLeg***		*+ContinuousLeg*	Ausprägung des Teilwegs als Fortbewegung mit einem kontinuierlich verfügbaren Verkehrsmittel. Vgl. 9.3.8.

Tabelle 94: Beschreibung der Struktur ***TripLegStructure***.

9.3.6. TimedLegStructure

TimedLegStructure			*+Structure*	Beinhaltet einen fahrplangebundenen Verbindungs-anteil (Teilweg).
	LegBoard	**1:1**	*+LegBoard*	Beginn (Haltepunkt) des Teilwegs. Vgl. 9.3.9.
	LegIntermediates	0:*	*+LegIntermediate*	Dazwischenliegende, durchfahrene Haltepunkte auf dem Teilweg zwischen *LegBoard* und *LegAlight*. Vgl. 9.3.11..
	LegAlight	**1:1**	*+LegAlight*	Ende (Haltepunkt) des Teilwegs. Vgl. 9.3.10.
	Service	**1:1**	*+DatedJourney*	Angaben zum Verkehrsmittel, wie Linie, Verkehrs-mitteltyp etc. Vgl. 7.6.4
	LegAttribute	0:*	*+LegAttribute*	Attribute, die nur auf einem Teil der Verbindung gültig sind und nicht für die ganze Fahrt. Vgl. 7.6.13.
Opera-ting-Days	*OperatingDays*	0:1	*+OperatingDays*	Verkehrstage für diese Verbindung. Vgl.7.4.8.
	OperatingDaysDescrip-tion	0:1	*+InternationalText*	Menschenlesbare Beschreibung der Verkehrstage, z. B. „Montag bis Freitag“ oder „Sonn- und Feiertag“.
	LegTrack	0:1	*+LegTrack*	Detaillierter (geometrischer) Verlauf. Vgl. 9.3.12.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 95: Beschreibung der Struktur ***TimedLegStructure***.

9.3.7. InterchangeLegStructure

InterchangeLegStructure				*+Structure*	**Beinhaltet einen Verbindungsanteil (Teilweg), der einen Umsteigevorgang zwischen zwei Verkehrsmitteln darstellt.**
	a	***InterchangeMode***	**-1:1**	*walk* | *parkAndRide* | *bikeAndRide* | *carHire* | *bikeHire* | *protectedConnection* | *guaranteedConnection* | *remainInVehicle* | *changeWithinVehicle* | *checkIn* | *checkOut*	Klassifizierung von Umsteigevorgängen
	b	***ContinuousMode***		*walk* | *demandResponsive* | *replacementService*	Modalität für kontinuierliche Verkehre.
	LegStart		**1:1**	*+LocationRef*	Beginn (Ort) dieses Teilwegs. Vgl. 7.5.11.
	LegEnd		**1:1**	*+LocationRef*	Ende (Ort) dieses Teilwegs. Vgl. 7.5.11.
TimeWindow	*TimeWindowStart*		0:1	*xs:dateTime*	Frühester Zeitpunkt für den Start dieses Teilwegs.
	TimeWindowEnd		0:1	*xs:dateTime*	Spätester Zeitpunkt für das Ende dieses Teilwegs.
InterchangeDuration	***Duration***		**1:1**	*xs:duration*	Notwendige Gesamtumsteigezeit.
	WalkDuration		0:1	*xs:duration*	Fußweganteil der Gesamtumsteigezeit.
	BufferTime		0:1	*xs:duration*	Pufferzeitanteil der Gesamtumsteigezeit. Pufferzeiten („Check-In-Zeiten“) sind bei manchen Verkehrsmitteln vorgeschrieben, z. B. im Flugverkehr, bei Fähren oder auch Hochgeschwindigkeitszügen.
	LegDescription		0:1	*+InternationalText*	Beschreibung des Umsteigevorgangs.
	Length		0:1	*LengthType*	Länge des Umsteigewegs.
	Attribute		0:*	*+GeneralAttribute*	Hinweise und Attribute (mit Klassifikationen) zum Umsteigevorgang. Vgl. 7.4.10.
	NavigationPath		0:1	*+NavigationPath*	Detaillierte Informationen zum geometrischen Verlauf, der Wegfolge und Zugänglichkeit. Vgl. 9.3.14.
	SituationFullRef		0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im Trip*ResponseContext* (vgl. 9.3.2) zu finden sein oder auf anderem Wege bekannt gemacht werden.
	Extension		0:1	*xs:anyType*	Erweiterungen.

Tabelle 96: Beschreibung der Struktur ***InterchangeLegStructure***.

9.3.8. ContinuousLegStructure

ContinuousLegStructure			*+Structure*	**Beinhaltet einen Verbindungsanteil (Teilweg), der nicht fahrplangebunden ist (z. B. Fußweg).**
	LegStart	**1:1**	*+LocationRef*	Beginn (Ort) des Teilwegs dieser Verbindung. Vgl. 7.5.11.
	LegEnd	**1:1**	*+LocationRef*	Ende (Ort) des Teilwegs dieser Verbindung. Vgl. 7.5.11.
	Service	**1:1**	*+ContinuousService*	Angaben zum „Verkehrsmittel“ (z. B. Fußweg). Vgl. 7.6.8.
Time-Window	*TimeWindowStart*	0:1	*xs:dateTime*	Frühester Zeitpunkt für den Start dieses Teilwegs.
	TimeWindowEnd	0:1	*xs:dateTime*	Spätester Zeitpunkt für das Ende dieses Teilwegs.
	Duration	**1:1**	*xs:duration*	Dauer dieses Teilwegs.
	LegDescription	0:1	*+InternationalText*	Beschreibung dieses Verbindungsanteils.
	Length	0:1	*LengthType*	Länge dieses Teilwegs.
	LegTrack	0:1	*+LegTrack*	Detaillierter (geometrischer) Verlauf. Vgl. 9.3.12.
	NavigationPath	0:1	*+NavigationPath*	Detaillierte Informationen zum geometrischen Verlauf, der Wegfolge und Zugänglichkeit. Vgl. 9.3.14.
	SituationFullRef	0:*	*+SituationFullRef*	Verweis auf eine Störungsnachricht. Diese Nachricht kann im Trip*ResponseContext* (vgl. 9.3.2) zu finden sein oder auf anderem Wege bekannt gemacht werden. Vgl. 7.8.2.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 97: Beschreibung der Struktur ***ContinuousLegStructure***.

9.3.9. LegBoardStructure

LegBoardStructure			*+Structure*	**Beschreibt die Einstiegssituation in ein Verkehrsmittel.**
Stop-Point	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+International Text*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
ServiceArrival	*TimetabledTime*	0:1	*xs:dateTime*	Ankunftszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Ankunftszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Ankunftszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Ankunftszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Ankunftszeit.
ServiceDeparture	***TimetabledTime***	**1:1**	*xs:dateTime*	Abfahrtszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Abfahrtszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Abfahrtszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Abfahrtszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Abfahrtszeit.
	MeetsViaRequest	0:1	*xs:boolean*	Dieser Halt erfüllt eine der in der Anfrage vorgegebenen Via-Bedingungen.
StopCallStatus	***StopSeqNumber***	**1:1**	*xs:positiveInteger*	Laufende Nummer des Halts im Fahrweg der Fahrt.
	DemandStop	0:1	*xs:boolean*	Bedarfshalt. Fahrzeug bedient diesen Halt nur nach Voranmeldung.
	UnplannedStop	0:1	*xs:boolean*	Halt, der laut Planung nicht vorgesehen war.
	NotServicedStop	0:1	*xs:boolean*	Entgegen der Planung findet kein Halt statt.

Tabelle 98: Beschreibung der Struktur ***LegBoardStructure***.

9.3.10. LegAlightStructure

LegAlightStructure			*+Structure*	**Beschreibt die Ausstiegssituation aus einem Verkehrsmittel.**
Stop-Point	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+International Text*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
Ser-viceAr-rival	***TimetabledTime***	**1:1**	*xs:dateTime*	Ankunftszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Ankunftszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Ankunftszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Ankunftszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Ankunftszeit.
Ser-viceDe-parture	*TimetabledTime*	0:1	*xs:dateTime*	Abfahrtszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Abfahrtszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Abfahrtszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Abfahrtszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Abfahrtszeit.
	MeetsViaRequest	0:1	*xs:boolean*	Dieser Halt erfüllt eine der in der Anfrage vorgegebenen Via-Bedingungen.
Stop-CallSta-tus	***StopSeqNumber***	**1:1**	*xs:positiveInteger*	Laufende Nummer des Halts im Fahrweg der Fahrt.
	DemandStop	0:1	*xs:boolean*	Bedarfshalt. Fahrzeug bedient diesen Halt nur nach Voranmeldung.
	UnplannedStop	0:1	*xs:boolean*	Halt, der laut Planung nicht vorgesehen war.
	NotServicedStop	0:1	*xs:boolean*	Entgegen der Planung findet kein Halt statt.

Tabelle 99: Beschreibung der Struktur ***LegAlightStructure***.

9.3.11. LegIntermediateStructure

LegIntermediateStructure			*+Structure*	**Zwischenhalt auf einem Teilweg.**
Stop-Point	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+International Text*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
ServiceArrival	***TimetabledTime***	**1:1**	*xs:dateTime*	Ankunftszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Ankunftszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Ankunftszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Ankunftszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Ankunftszeit.
ServiceDeparture	***TimetabledTime***	**1:1**	*xs:dateTime*	Abfahrtszeit nach Fahrplan.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Abfahrtszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Abfahrtszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Abfahrtszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Abfahrtszeit.
	MeetsViaRequest	0:1	*xs:boolean*	Dieser Halt erfüllt eine der in der Anfrage vorgegebenen Via-Bedingungen.
StopCallStatus	***StopSeqNumber***	**1:1**	*xs:positiveInteger*	Laufende Nummer des Halts im Fahrweg der Fahrt.
	DemandStop	0:1	*xs:boolean*	Bedarfshalt. Fahrzeug bedient diesen Halt nur nach Voranmeldung.
	UnplannedStop	0:1	*xs:boolean*	Halt, der laut Planung nicht vorgesehen war.
	NotServicedStop	0:1	*xs:boolean*	Entgegen der Planung findet kein Halt statt.

Tabelle 100: Beschreibung der Struktur ***LegIntermediateStructure***.

9.3.12. LegTrackStructure

LegTrackStructure			*+Structure*	**Container für die Streckenabschnitte entlang eines Verbindungsteilwegs.**
	TrackSection	**1:***	*+TrackSection*	Ein oder mehrere Streckenabschnitte. Vgl. 9.3.13.

Tabelle 101: Beschreibung der Struktur ***LegTrackStructure***.

9.3.13. TrackSectionStructure

TrackSectionStructure			*+Structure*	**Ein Streckenabschnitt in einem Verbindungsteilweg.**
	TrackStart	0:1	*+LocationRef*	Beginn (Ort) des Streckenabschnitts. Vgl. 7.5.11.
	TrackEnd	0:1	*+LocationRef*	Ende (Ort) des Streckenabschnitts. Vgl. 7.5.11.
Projection	***Position***	**2:***	*+GeoPosition*	Geographische Projektion des Streckenabschnitts als Polygonzug. Vgl. 7.2.3.
	RoadName	0:1	*xs:string*	Name der Straße, auf der dieser Streckenabschnitt liegt.
	Duration	0:1	*xs:duration*	Zeitdauer, die der Fahrgast braucht, um diesen Streckenabschnitt zu bewältigen.
	Length	0:1	*LengthType*	Länge des Streckenabschnitts.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 102: Beschreibung der Struktur ***TrackSectionStructure***.

9.3.14. NavigationPathStructure

NavigationPathStructure			*+Structure*	Container für Wegbeschreibungen.
	NavigationSection	**1:***	*+NavigationSection*	Ein oder mehrere Streckenabschnitte. Vgl. 9.3.15

Tabelle 103: Beschreibung der Struktur ***NavigationPathStructure***.

9.3.15. NavigationSectionStructure

NavigationSectionStructure			*+Structure*	Beschreibung eines Wegstücks, evtl. mit Angabe der geographischen Einbettung, der Abbiegeanweisungen und der Wegbeschaffenheit (Zugänglichkeit für mobilitätseigeschränkte Personen).
	TrackSection	0:1	*+TrackSection*	Geographische Beschreibung des Streckenabschnitts. Vgl. 9.3.13.
	TurnDescription	0:1	*+InternationalText*	Beschreibung des durchzuführenden Manövers. Es sollte in textueller Form die Inhalte von *Manoeuvre*, *TurnAction* und *TrackSection.RoadName* beschreiben.
	Manoeuvre	0:1	*origin* | *destination* | *continue* | *keep* | *turn* | *leave* | *enter*	Codierung des durchzuführenden Manövers.
	TurnAction	0:1	*sharp left* | *left* | *half left* | *straight on* | *half right* | *right* | *sharp right* | *uturn*	Codierung der Abbiegevorgänge.
	DirectionHint	0:1	*+InternationalText*	Textueller Richtungshinweis zum besseren Verständnis des nachfolgenden Streckenabschnittes, z. B. „Folgen Sie der Beschilderung nach Hamburg“.
	Bearing	0:1	*AbsoluteBearing*	Himmelsrichtung, die nach dem Manöver eingeschlagen ist. Sie bezieht sich nicht auf das gesamte Wegstück.
	SituationFullRef	0:*	*+SituationFullRef*	Verweise auf Störungsnachrichten. Diese Nachrichten können im *TripResponseContext* (vgl. 9.3.2) zu finden sein oder auf anderem Wege bekannt gemacht werden. Vgl. 7.8.2.
	AccessPath	0:1	*+AccessPath*	Beschreibung der Zugänglichkeit des Wegstücks. Vgl. 9.3.16.

Tabelle 104: Beschreibung der Struktur ***NavigationSectionStructure***.

9.3.16. AccessPathStructure

AccessPathStructure			*+Structure*	**Beschreibung der Zugänglichkeit eines Weg-stücks.**
	Transition	0:1	*up* | *down* | *level* | *upAndDown* | *downAndUp*	Angabe, ob Weg eben geht oder aufwärts/abwärts führt.
	AccessFeatureType	0:1	*lift* | *stairs* | *seriesOfStairs* | *escalator* | *ramp* | *foot-path*	Wegtyp.
	Count	0:1	*xs:positiveInteger*	Anzahl, wie oft der Wegtyp vorkommt.

Tabelle 105: Beschreibung der Struktur ***AccessPathStructure***.

10. Dienst Abfahrtstafeln

10.1. Beschreibung

Dieser Dienst informiert über Ankünfte und Abfahrten von ÖV-Fahrten an Haltestellen für einen bestimmten Zeitpunkt oder Zeitraum. In den Anfrageparametern können Einschränkungen vorgegeben werden, die sich als Filter auf die Ergebnisse auswirken.

10.2. Anfragestrukturen

Eine Abfahrtstafel (bzw. Ankunftstafel) wird mittels eines Elements ***StopEventRequest*** vom Typ *StopEventRequestStructure* angefordert.

10.2.1. StopEventRequestStructure

StopEventRequestStructure			*+Structure*	**Fasst die Anfragedaten für eine Abfahrts- oder Ankunftstafel zusammen.**
	Location	**1:1**	*+LocationContext*	Ortsdaten für die Abfahrts-/Ankunftstafel. Vgl. 7.6.11.
	Params	0:1	*+StopEventParam*	Spezifische Anfrageparameter. Vgl. 10.2.2.

Tabelle 106: Beschreibung der Struktur ***StopEventRequestStructure***.

10.2.2. StopEventParamStructure

StopEventParamStructure			*+Structure*	**Fasst die Anfrageparameter zusammen, die die Berechnung einer Abfahrts- oder Ankunftstafel steuern.**
StopEventDataFilter	*PtModeFilter*	0:1	*+PtModeFilter*	Erlaubte Verkehrsmittel. Vgl. 7.3.5.
	LineFilter	0:1	*+LineDirectionFilter*	Erlaubte Linien (ggf. verfeinert auf Richtungen). Vgl. 7.4.6.
	OperatorFilter	0:1	*+OperatorFilter*	Erlaubte Verkehrsunternehmen. Vgl. 7.4.4.
StopEventPolicy	*NumberOfResults*	0:1	*xs:positiveInteger*	Maximale Zahl von Abfahrts-/Ankunftsereignissen, die in der Antwort zurückgegeben werden sollen.
	TimeWindow	0:1	*xs:duration*	Zeitfenster, in dem Abfahrts-/Ankunftsereignisse in der Antwort zurückgegeben werden sollen. Wird gerechnet ab dem in *LocationContext* angegebenen Zeitpunkt.
	StopEventType	0:1	*departure* | *arrival* | *both*	Gibt an, ob Abfahrts- oder Ankunftsereignisse oder beides zurückgegeben werden sollen.
StopEventContentFilter	*IncludePreviousCalls*	0:1	*xs:boolean*	Gibt an, ob je Fahrt die vorausgehenden Halte angeführt werden sollen.
	IncludeOnwardCalls	0:1	*xs:boolean*	Gibt an, ob je Fahrt die nachfolgenden Halte angeführt werden sollen.
	IncludeOperatingDays	0:1	*xs:boolean*	Gibt an, ob die Verkehrstage der Fahrten angegeben werden sollen.
	IncludeRealtimeData	0:1	*xs:boolean*	Steuert, ob Echtzeitdaten berücksichtigt und ausgegeben werden sollen.

Tabelle 107: Beschreibung der Struktur ***StopEventParamStructure***.

10.3. Antwortstrukturen

Das Ergebnis einer Abfahrtstafelanfrage wird mittels eines Elements ***StopEventResponse*** vom Typ *StopEventResponseStructure* übertragen.

10.3.1. StopEventResponseStructure

StopEventResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Abfahrts- oder Ankunftstafelanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	StopEventResponse-Context	0:1	*+StopEventResponse-Context*	Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl.10.3.2.
	StopEventResult	0:*	*+StopEventResult*	Container für ein Abfahrts- oder Ankunftsereignis. Vgl. 10.3.3.

Tabelle 108: Beschreibung der Struktur ***StopEventResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

STOPEVENT_DATEOUTOFRANGE	Für das angefragte Datum liegen keine Fahrplandaten vor.
STOPEVENT_LOCATIONUNKNOWN	Der Ort (Adresse, Haltestelle, …), für den die Abfahrtstafel/Ankunftstafel angefordert wurde, ist unbekannt.
STOPEVENT_LOCATIONUNSERVED	Der Ort (Adresse, Haltestelle, …), für den die Abfahrtstafel/Ankunftstafel angefordert wurde, wird überhaupt nicht von öffentlichen Verkehrsmitteln bedient.
STOPEVENT_NOEVENTFOUND	Im fraglichen Zeitraum wurde keine Abfahrt/Ankunft unter Einhaltung der gegebenen Optionen gefunden.

Tabelle 109: Liste der Fehlerzustände in ***StopEventResponse*.**

10.3.2. StopEventResponseContextStructure

StopEventResponseContextStructure	*+Structure (abgeleitet von AbstractResponseContextStructure)*	**Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl. 7.6.12.**

Tabelle 110: Beschreibung der Struktur ***StopEventResponseContextStructure***.

10.3.3. StopEventResultStructure

StopEventResultStructure			*+Structure*	**Fasst die Ergebnisdaten für ein einzelnes Abfahrts- oder Ankunftsereignis zusammen.**
	ResultId	**1:1**	*xs:NMTOKEN*	ID des Resultats für spätere Referenzierung bzw. für Debug-Zwecke.
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf dieses Abfahrts-/Ankunftsereignis. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
	StopEvent	**1:1**	*+StopEvent*	Daten zu einem Abfahrts- oder Ankunftsereignis. Vgl. 10.3.4.

Tabelle 111: Beschreibung der Struktur ***StopEventResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

STOPEVENT_LASTSERVICEOFTHISLINE	Diese Abfahrt/Ankunft ist die letzte dieser Linie an dieser Haltestelle an diesem Betriebstag.
STOPEVENT_NOREALTIME	Für diese Abfahrt/Ankunft liegen keine Echtzeitdaten oder Prognosen vor.

Tabelle 112: Liste der Fehlerzustände in ***StopEventResult*.**

10.3.4. StopEventStructure

StopEventStructure			*+Structure*	**Daten zu einem einzelnen Abfahrts- oder Ankunftsereignis.**
	PreviousCall	0:*	*+CallAtNearStop*	Abfahrts-/Ankunftsereignisse an Haltestellen vor der angefragten Haltestelle. Vgl. 10.3.5.
	ThisCall	**1:1**	*+CallAtNearStop*	Abfahrts-/Ankunftsereignis an der angefragten Haltestelle. Vgl. 10.3.5.
	OnwardCall	0:*	*+CallAtNearStop*	Abfahrts-/Ankunftsereignis nach der angefragten Haltestelle. Vgl. 10.3.5.
	Service	**1:1**	*+DatedJourney*	Angaben zum Verkehrsmittel, Linie etc. Vgl. 7.6.2
Operating-Days	*OperatingDays*	0:1	*+OperatingDays*	Verkehrstage für dieses Abfahrts-/Ankunftsereignis. Vgl. 7.4.8..
	OperatingDaysDescription	0:1	*+InternationalText*	Menschenlesbare Beschreibung der Verkehrstage, z. B. „Montag bis Freitag“ oder „Sonn- und Feiertag“.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 113: Beschreibung der Struktur ***StopEventStructure***.

10.3.5. CallAtNearStopStructure

CallAtNearStopStructure			*+Structure*	**Abfahrt oder Ankunft an einer Haltestelle in der Umgebung.**
	CallAtStop	**1:1**	*+CallAtStop*	Abfahrt oder Ankunft an einem Haltepunkt. Vgl.7.6.6..
	WalkDistance	0:1	*Distance*	Distanz des Haltepunkts vom angefragten Ort in Metern. Der angefragte Ort kann z. B. eine Adresse sein.
	WalkDuration	0:1	*xs:duration*	Zeitliche Distanz des Haltepunkts vom angefragten Ort. Der angefragte Ort kann z. B. eine Adresse sein. Der Zeitbedarf ergibt sich durch die IV-Einstellungen in der Anfrage: es wird also z. B. berücksichtigt, ob ein Fahrrad benutzt werden kann, um vom angefragten Ort zum Abfahrtshaltepunkt zu gelangen.

Tabelle 114: Beschreibung der Struktur ***CallAtNearStopStructure***.

11. Dienst Logische Ortung

11.1. Beschreibung

Der Dienst Logische Ortung hat die Aufgabe, den Aufenthaltsort des Fahrgasts im ÖV-Netz zu bestimmen. Er benutzt dabei das Bewegungsmuster des Fahrgasts, das entweder sein Mobilgerät aufgezeichnet hat oder durch den Fahrplan des Fahrzeugs bestimmt wird, in dem er sich gerade befindet. Als Resultat erhält man mögliche Aufenthaltsorte mit Angabe der jeweiligen Wahrscheinlichkeit.

11.2. Anfragestrukturen

Eine logische Ortung wird mittels eines Elements ***PositioningRequest*** vom Typ *PositioningRequestStructure* angefordert.

11.2.1. PositioningRequestStructure

PositioningRequestStructure				*+Structure*	**Fasst die Anfragedaten für eine logische Ortung zusammen.**
	a	***LastPositions***	**-1:1**	*+TimedPosition*	Bewegungsmuster des Fahrgasts als Folge von Koordinaten mit Zeitstempel. Vgl. 11.2.3.
	b	***StopSequence***		*+TimedStop*	Bewegungsmuster des Fahrgasts als Folge von Haltepunkten mit Uhrzeiten. Vgl. 11.2.4.
	Params		0:1	*+PositioningParam*	Spezifische Anfrageparameter. Vgl. 11.2.2.

Tabelle 115: Beschreibung der Struktur ***PositioningRequestStructure***.

11.2.2. PositioningParamStructure

PositioningParamStructure			*+Structure*	**Fasst die Anfrageparameter zusammen, die die Bestimmung einer logischen Ortung des Fahrgasts im ÖV-Netz steuern.**
Positioning-DataFilter	*PtModeFilter*	0:1	*+PtModeFilter*	Verkehrsmittel-Filter. Vgl. 7.3.5.
	LineFilter	0:1	*+LineDirectionFilter*	Linien-Filter (ggf. verfeinert auf Richtungen). Vgl. 7.4.6.
	OperatorFilter	0:1	*+OperatorFilter*	Verkehrsunternehmen-Filter. Vgl. 7.4.4.
Positioning-Policy	*NumberOfResults*	0:1	*xs:positiveInteger*	Maximale Zahl von Ortungsvorschlägen, die in der Antwort zurückgegeben werden dürfen.

Tabelle 116: Beschreibung der Struktur ***PositioningParamStructure***.

11.2.3. TimedPositionStructure

TimedPositionStructure			*+Structure*	**Geographische Position mit Zeitstempel.**
	Timestamp	**1:1**	*xs:dateTime*	Zeitstempel, wann der Fahrgast diesen Punkt passiert hat.
	Position	**1:1**	*+GeoPosition*	Koordinatenposition. Vgl. 7.2.3.
	Speed	0:1	*Speed*	Geschwindigkeit, mit der der Fahrgast die Position passiert hat. Vgl. 7.2.1.
	Direction	0:1	*AbsoluteBearing*	Kompassrichtung, in der der Fahrgast die Position passiert hat. Vgl. 7.2.1.

Tabelle 117: Beschreibung der Struktur ***TimedPositionStructure***.

11.2.4. TimedStopStructure

TimedStopStructure			*+Structure*	**Beschreibt einen Haltepunkt evtl. mit Zeiten, wann er erreicht und/oder verlassen wird.**
Stop-Point	***StopPointRef***	**1:1**	*→StopPoint*	Referenz auf einen Code für einen Haltepunkt. Vgl. 7.5.1.
	StopPointName	**1:1**	*+International Text*	Name des Haltepunkts für Fahrgastinformation.
	NameSuffix	0:1	*+International Text*	Namenszusatz, der bei Platzmangel evtl. auch weggelassen werden kann, z. B.: „gegenüber vom Haupteingang“.
	PlannedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach Planungsstand.
	EstimatedBay	0:1	*+International Text*	Name des Steigs/Haltepunkts, wo in das Fahrzeug ein- oder ausgestiegen werden muss (bei Verwendung in Zusammenhang mit einer konkreten Verbindungsauskunft, wenn in StopPointName ein allgemeiner Name angegeben ist, ähnlich Haltestellenname). Nach letztem Prognosestand.
Arrival-Time	*TimetabledTime*	0:1	*xs:dateTime*	Geplante Ankunftszeit.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Ankunftszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Ankunftszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Ankunftszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Ankunftszeit.
Departure-Time	*TimetabledTime*	0:1	*xs:dateTime*	Geplante Abfahrtszeit.
	RecordedAtTime	0:1	*xs:dateTime*	Tatsächliche Abfahrtszeit.
	EstimatedTime	0:1	*xs:dateTime*	Erwartete Abfahrtszeit.
	EstimatedTimeLow	0:1	*xs:dateTime*	Untere Schranke für erwartete Abfahrtszeit.
	EstimatedTimeHigh	0:1	*xs:dateTime*	Obere Schranke für erwartete Abfahrtszeit.

Tabelle 118: Beschreibung der Struktur ***TimedStopStructure***.

11.3. Antwortstrukturen

Das Ergebnis einer Ortungsanfrage wird mittels eines Elements ***PositioningResponse*** vom Typ *PositioningResponseStructure* übertragen.

11.3.1. PositioningResponseStructure

PositioningResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Ortungsanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	PositioningResult	0:1	*+PositioningResult*	Struktur für ein Ortungsresultat. Vgl. 11.3.2.

Tabelle 119: Beschreibung der Struktur ***PositioningResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

POSITIONING_NOMATCH	Kein passendes Resultat gefunden.
POSITIONING_DATEOUTOFRANGE	Für das angefragte Datum liegen keine Fahrplandaten vor.
POSITIONING_SPEEDTOOFAST	Die dem Bewegungsmuster zugrundeliegende Geschwindigkeit ist zu schnell.
POSITIONING_COORDOUTOFRANGE	Die angegebenen Koordinaten liegen außerhalb des betrachteten Gebiets.
POSITIONING_STOPUNKNOWN	Eine angegebene Haltestelle ist unbekannt.

Tabelle 120: Liste der Fehlerzustände in ***PositioningResponse*.**

11.3.2. PositioningResultStructure

PositioningResultStructure			*+Structure*	**Resultatstruktur für das Ortungsergebnis.**
	ResultId	**1:1**	*xs:NMTOKEN*	ID des Resultats für spätere Referenzierung bzw. für Debug-Zwecke.
	Positioning	**1:1**	*+Positioning*	Container für die Ortungsvorschläge. Vgl. 11.3.3
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 121: Beschreibung der Struktur ***PositioningResultStructure***.

11.3.3. PositioningStructure

PositioningStructure			*+Structure*	**Container für die Ortungsvorschläge.**
	RankedPosition	**1:***	*+RankedPosition*	Ein oder mehrere Ortungsvorschläge. Vgl. 11.3.4.

Tabelle 122: Beschreibung der Struktur ***PositioningStructure***.

11.3.4. RankedPositionStructure

RankedPositionStructure				*+Structure*	**Positionsvorschlag mit Wahrscheinlichkeitseinstufung.**
	a	***StationaryLocation***	**-1:1**	*+LocationRef*	Position im ÖV-Netz außerhalb Fahrzeugen. Vgl. 7.5.11.
	b	***TripLocation***		*+DatedJourney*	Position im ÖV-Netz in einer Fahrt. Vgl.7.6.2.
	Ranking		**1:1**	*Percent*	Wahrscheinlichkeitseinstufung in Prozent. Der Wert 100 bedeutet absolute Gewissheit. Vgl. 7.2.1.

Tabelle 123: Beschreibung der Struktur ***RankedPositionStructure***.

12. Dienst Fahrtinformation (EKAP)

12.1. Beschreibung

Im Dienst Fahrtinformation (EKAP) liefert eine EKAP Informationen zu einer bestimmten Fahrt.

12.2. Anfragestrukturen

Die Fahrtinformationen werden mittels eines Elements ***TripInfoRequest*** vom Typ *TripInfo-RequestStructure* angefordert.

12.2.1. TripInfoRequestStructure

TripInfoRequestStructure				*+Structure*	**Fasst die Anfragedaten für eine Fahrtinformationsanfrage zusammen.**
	a	***JourneyRef***	**-1:1**	*→Journey*	Referenz auf eine Fahrt. Vgl. 7.4.1.
		OperatingDayRef	**-1:1**	*→Operating-Day*	Referenz auf einen Verkehrstag. Vgl. 7.4.1.
	b	***VehicleRef***	**-1:1**	*→Vehicle*	Referenz auf ein Fahrzeug. Vgl. 7.4.1.
		TimeOfOperation		*xs:dateTime*	Zeitpunkt, wann das Fahrzeug unterwegs ist. Dieser Wert wird in den meisten Anwendungsfällen dem Zeitpunkt „Jetzt“ entsprechen.
	Params		0:1	*+TripInfoParam*	Parameter, die die Suche und Rückgabewerte beeinflussen können. Vgl. 12.2.2.

Tabelle 124: Beschreibung der Struktur ***TripInfoRequestStructure***.

Die Informationen zu einer Fahrt können mittels einer Fahrt-ID (*JourneyRef*) oder einer Fahrzeug-ID (*VehicleRef*) angefragt werden.

Bei Verwendung der Fahrzeug-ID wird mit der zusätzlichen Information des Zeitpunkts in *TimeOfOperation* eindeutig die Fahrt aus der Menge aller Fahrten ausgewählt, die das Fahrzeug an dem gewählten Tag durchführt.

12.2.2. TripInfoParamStructure

TripInfoParamStructure			*+Structure*	**Fasst die Parameter für eine Fahrtinformationsanfrage zusammen.**
TripInfoPolicy	*UseTimetabledDataOnly*	0:1	*xs:boolean*	Legt fest, ob im Resultat Informationen zu den Verkehrstagen mitausgegeben werden sollen. Default ist *false*.
TripInfoContentFilter	*IncludeCalls*	0:1	*xs:boolean*	Legt fest, ob im Resultat die Halte der Fahrt ausgegeben werden sollen. Default ist *true*.
	IncludePosition	0:1	*xs:boolean*	Legt fest, ob im Resultat die aktuelle Position der ausgegeben werden soll. Default ist *true*.
	IncludeService	0:1	*xs:boolean*	Legt fest, ob im Resultat Verkehrsmittelinformationen zur Fahrt ausgegeben werden sollen. Default ist *true*.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 125: Beschreibung der Struktur ***TripInfoParamStructure***.

12.3. Antwortstrukturen

Das Ergebnis einer Fahrtinformationsanfrage wird mittels eines Elements ***TripInfoResponse*** vom Typ *TripInfoResponseStructure* übertragen.

12.3.1. TripInfoResponseStructure

TripInfoResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Fahrtinformationsanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	TripInfoResponse-Context	0:1	*+TripInfoRespon seContext*	Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl. 12.3.2.
	TripInfoResult	0:1	*+TripInfoResult*	Container für die Fahrtinformationen. Vgl. 12.3.3.

Tabelle 126: Beschreibung der Struktur ***TripInfoResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

TRIPINFO_JOURNEYUNKNOWN	Die angefragte Fahrt-ID (*JourneyRef*) ist unbekannt.
TRIPINFO_VEHICLEUNKNOWN	Die angefragte Fahrzeug-ID (*VehicleRef*) ist unbekannt.
TRIPINFO_NOJOURNEYFOUND	Für die angefragte Fahrzeug-ID (*VehicleRef*) kann für die Uhrzeit keine passende Fahrt gefunden werden.

Tabelle 127: Liste der Fehlerzustände in ***TripInfoResponse***.

12.3.2. TripInfoResponseContextStructure

TripInfoResponseContextStructure	*+Structure (abgeleitet von AbstractResponseContextStructure)*	**Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl. 7.6.12.**

Tabelle 128: Beschreibung der Struktur ***TripInfoResponseContextStructure***.

12.3.3. TripInfoResultStructure

TripInfoResultStructure			*+Structure*	**Resultatstruktur, die die Fahrtinformationen zusammenfasst.**
	PreviousCall	0:*	*+CallAtNearStop*	Bereits zurückgelegte Halte. Umfasst auch den aktuellen Halt, falls sich die Fahrt gerade an einer Haltestelle befindet. Vgl. 10.3.5.
	CurrentPosition	0:1	*+VehiclePosition*	Aktuelle Position des Fahrzeugs. Vgl. 7.6.9.
	OnwardCall	0:*	*+CallAtNearStop*	Die noch bevorstehenden Halte der Fahrt. Vgl. 10.3.5.
	Service	0:1	*+DatedJourney*	Angaben zum Verkehrsmittel, Linie etc. Vgl. 7.6.2.
Operating-Days	*OperatingDays*	0:1	*+OperatingDays*	Verkehrstage für diese Verbindung. Vgl.7.4.8.
	OperatingDaysDescription	0:1	*+InternationalText*	Menschenlesbare Beschreibung der Verkehrstage, z. B. „Montag bis Freitag“ oder „Sonn- und Feiertag“.
ServiceFacility	*:::*	0:1	*+siri:ServiceFacilityGroup*	Klassifizierung der Ausstattungsmerkmale. Vgl. 7.7.3.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 129: Beschreibung der Struktur ***TripInfoResultStructure***.

13. Anschlussdienste

13.1. Beschreibung

Unter dem Begriff „Anschlussdienste“ werden unterschiedliche Dienste des TRIAS-Standards zusammengefasst, die der Kommunikation zu Anschlüssen dienen. Die Anschlussdienste setzen sich aus den Diensten

- Anschlussmeldung,
- Anschlussstatus,
- Info zu Anschlussverlust und
- Anschlussrückmeldung

zusammen. Nachfolgend sind zwei Abläufe dokumentiert, die eine mögliche Nutzung der Dienste darstellen.

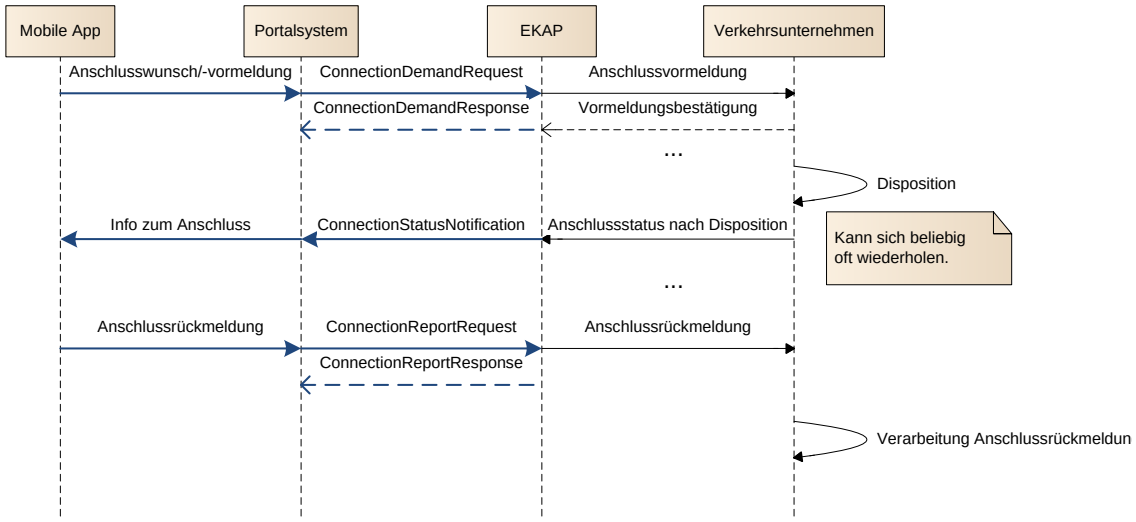

Abbildung 7: Ablauf der Anschlussdienste mit aktiver Benachrichtigung bei Statusänderung

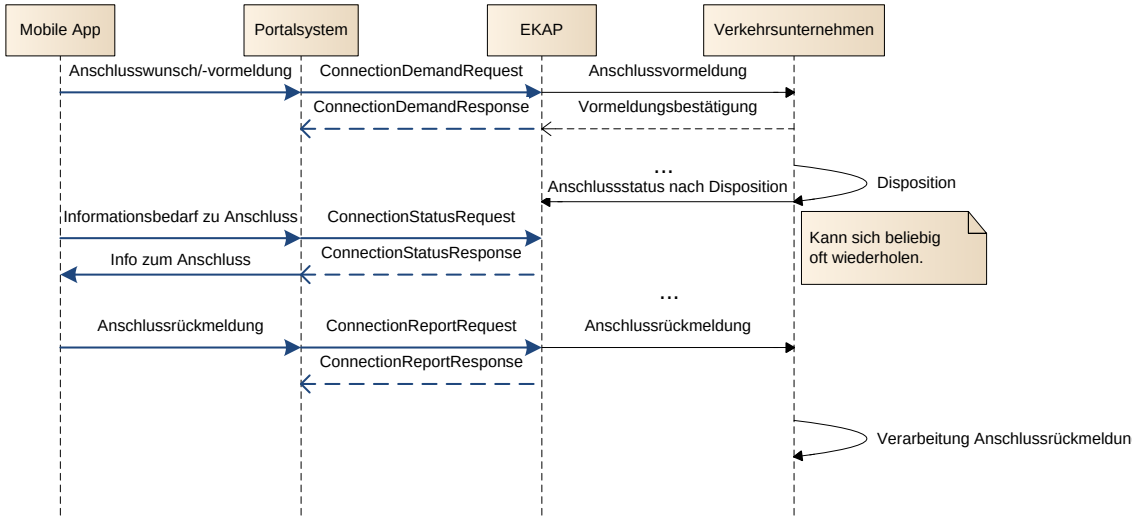

Abbildung 8: Ablauf der Anschlussdienste mit passiver Benachrichtigung bei Statusänderung

Der Ablauf gestaltet sich im Allgemeinen folgendermaßen:

1. Anschlussmeldung durch Reisenden/Applikation, Zugbegleiter oder System über das Portalsystem und die EKAP an das/die beteiligte/n Verkehrsunternehmen
2. Verarbeitung der Anschlussmeldung durch Verkehrsunternehmen und Anschlussdisposition
3. Information über Dispositionsmaßnahme
    a. aktive Information durch EKAP (Abbildung 7)
    b. passive Information mittels Anfrage an die EKAP (Abbildung 8)
4. Rückmeldung des Reisenden oder seiner Applikation zur Anschlusserreichung.

Der Ablauf ist nicht zwingend vorgegeben. Insbesondere kann es sinnvoll sein, einzelne Dienste ohne Bezug zu den anderen zu nutzen. Beispielsweise ist die Abfrage des Anschlussstatus durch die Verbindungsauskunft denkbar. Die Dienste sind genauer in den folgenden Unterkapiteln beschrieben.

13.1.1. Dienst Anschlussvoranmeldung

Mit Hilfe dieses Dienstes können Reisende ihre Anschlusswünsche mitteilen. Auf diese Weise werden Dispositionsverantwortliche in die Lage versetzt, Umsteigerzahlen abzuschätzen und in der Entscheidungsfindung einer Anschlussdisposition zu berücksichtigen. Der Mehrwert für die Reisenden ist entsprechend eine verbesserte Anschlussdisposition.

Anschlussbeziehungen umfassen dabei allerdings nicht nur ein Zubringer/Abbringer-Paar, sondern schließen auch Anschlussbeziehungen von einem Startort auf einen Abbringer mit ein. Der Abbringer wiederum kann ein normal verkehrendes Angebot sein, aber auch ein Anrufsammeltaxi (AST) oder Bedarfsverkehr mit fester oder variabler Linienführung und festen oder variablen Halten.

Somit kann und soll die Anschlussmeldung auch als Bestellung für einen Bedarfsverkehr eingesetzt werden.

Der Dienst übermittelt unterschiedliche Grade der Wahrscheinlichkeit, mit der ein Nutzer die gewählte Verbindung nimmt. Auf diese Weise können auch nicht sicher gewählte Verbindungen anhand der Wahrscheinlichkeiten für die Disposition verwendet werden.

Weiterhin besteht die Möglichkeit für die beteiligten Verkehrsunternehmen auf Basis der gemeldeten Umsteiger Rückschlüsse auf die Anzahl der Reisenden im Fahrzeug zu schließen und entsprechende Kapazitäten zu disponieren. Dies gilt insbesondere für die Einstiegs- und Ausstiegsmeldungen.

13.1.2. Dienst Anschlussstatus

Der Dienst Anschlussstatus ermöglicht es den Verkehrsunternehmen, andere Beteiligte über den Status einer Anschlussbeziehung (erwartetes Zustandekommen des Anschlusses) zu informieren.

In erster Linie dient dies der Kundeninformation. Aufgrund der Information über den Anschlussstatus wissen ein Kunde und auch seine Applikation, ob er seine Reisekette (auch bei Verspätung des Zubringers) in der geplanten Weise fortsetzen kann. Seine Applikation kann entsprechend reagieren und Alternativen suchen.

Auch andere Verkehrsunternehmen können Nutzer dieser Information sein. Sie können auf eine Disposition reagieren und von sich aus weitere Maßnahmen im Fall eines abgelehnten Anschlusses einleiten. Ferner lassen sich Prognosen zu Reisendenströmen aufgrund der aktuellen Verkehrslage stellen.

Zur sinnvollen Nutzung des Dienstes ist es erforderlich, dass die Betriebsleitsysteme Anschlussstatusinformationen liefern, sobald sie bekannt werden. Das kann durch eine Dispositionshandlung des Disponenten geschehen oder implizit durch Einflüsse des Betriebsablaufes. Das Betriebsleitsystem meldet die Anschlussinformationen an die Datendrehscheibe einer oder mehrerer EKAPs. Dort kann das Benachrichtigungssystem auf diese Daten zugreifen.

Der Anschlussstatus kann auch im Rahmen einer Verbindungsüberwachung durch den Benachrichtigungsdienst übermittelt werden (siehe Kapitel 19).

13.1.3. Dienst Info bei Anschlussverlust

Durch das Nicht-Zustandekommen eines Anschlusses kann ein Reisender nicht mehr seine ursprünglich geplante Reisekette wahrnehmen. Mit diesem Dienst kann ein Verkehrsunternehmen auf alternative Abbringer, auf die Bestellung von Taxen, Bussen oder Hotelzimmern, die Bereitstellung von Ersatz- oder Sonderfahrten, Umleitungen oder eine Kombination aus unterschiedlichen Maßnahmen verweisen. Die App des Fahrgastes kann anhand der vorgeschlagenen Alternativen prüfen, ob es sich um einen für den Fahrgast sinnvollen Vorschlag handelt und ihn im positiven Fall in die Suche nach Alternativen mit einbeziehen.

Der Dienst ist in die Antworten des Anschlussstatus integriert.

13.1.4. Dienst Anschlussrückmeldung

Mit Hilfe dieses Dienstes können Reisende das Transportunternehmen darüber informieren, ob ein Anschluss aus Sicht des Reisenden erfolgreich disponiert wurde bzw. erfolgreich zustande gekommen ist. Dazu sendet der Reisende eine Nachricht mit einem Anschluss, bestehend aus Zu- und Abbringer und einer Information, ob der Anschluss für ihn zustande kam.

Die Übermittlung der Anschlussrückmeldung kann aber auch automatisch durch die Applikation erfolgen, wenn beispielsweise anhand einer geplanten Verbindung bekannt ist, welche Umstiegsverbindungen genutzt werden sollen. Diese können durch die Applikation überprüft werden, und es kann ein automatisches Feedback bei erkanntem Anschluss oder Anschlussbruch erfolgen. Ein weiteres Beispiel ist mit dem automatischen Erkennen der Fahrzeuge verbunden. Mit Hilfe dieser Funktion kann erkannt werden, wann das Fahrzeug gewechselt wird und dazu eine Anschlusserfolgsmeldung versendet werden.

13.2. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

ConnectionStatusEnumeration	*unknown* | *planned* | *confirmed* | *broken*	Klassifizierung der Verbindungsstaus.
RecommendationTypeEnumeration	*NextService* | *DifferentRoute* | *Hotel* |*Taxi* | *Bus* | *Helpdesk* | *Hotline* | *Driver* | *Other*	Klassifizierung der Ausweichempfehlungen bei Verbindungsverlust.

Tabelle 130: Beschreibung der einfachen Typen

13.3. Komplexe Strukturen

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_Connections definiert sind.

13.3.1. DatedConnectionStructure

DatedConnectionStructure			*+Structure*	**Beinhaltet einen Zubringer und einen Abbringer für einen konkreten Betriebstag.**
	ConnectionId	**1:1**	*xs:NMTOKEN*	ID des Anschlusses für spätere Referenzierung bzw. für Debug-Zwecke.
	Feeder	**1:1**	*+FeederDistributor*	Zubringer des gemeldeten Anschlusswunsches, vgl. 13.3.2.
	Distributor	**1:1**	*+FeederDistributor*	Abbringer des gemeldeten Anschlusswunsches, vgl. 13.3.2.

Tabelle 131: Beschreibung der Struktur ***DatedConnectionStructure***

13.3.2. FeederDistributorStructure

FeederDistributorStructure			*+Structure*	**Beinhaltet einen Zubringer oder einen Abbringer an einem definierten Ort zu einer definierten Betriebszeit.**
DatedJourney-Ref	***JourneyRef***	**1:1**	*→Journey*	Referenz auf die Fahrt des Zubringers oder Abbringers. Vgl. 7.4.1.
	OperatingDayRef	**1:1**	*→Operating-Day*	Referenz auf den Betriebstag. Vgl. 7.4.1.
	ConnectionLocation	**1:1**	*+CallAtStop*	Ort des Anschlusses, vgl. 7.6.6.

Tabelle 132: Beschreibung der Struktur ***FeederDistributorStructure***

13.3.3. GeneralizedConnectionStructure

GeneralizedConnectionStructure				*+Structure*	**Definiert eine Umstiegsrelation. Enthält die Verbindungsarten Einstieg (Pickup), Ausstieg (SetDown) und Umstieg (DatedConnection)**
	a	***DatedConnection***	**-1:1**	*+DatedConnection*	Umstiegsverbindung, für die der Status abgefragt werden soll. Vgl. 13.3.1.
	b	***PickUpLocation***		*+DatedCallAtLocation*	Zustieg in ÖV-Mittel, für den der Status abgefragt werden soll. Vgl. 7.6.7.
	c	***SetDownLocation***		*+DatedCallAtLocation*	Ausstieg aus ÖV-Mittel, für den der Status abgefragt werden soll. Vgl. 7.6.7.

Tabelle 133: Beschreibung des Typs ***GeneralizedConnectionStructure***

13.3.4. ConnectionStatusStructure

ConnectionStatusStructure			*+Structure*	**Enthält den tatsächlichen Verbindungsstatus. Dieser besteht aus einem Ein-, Aus- oder Umstieg und einem zugehörigen Status.**
	Connection	**1:1**	*+GeneralizedConnection*	Ein-, Aus- oder Umstieg. Vgl. 13.3.3.
	Status	**1:1**	*ConnectionStatusEnumeration*	Verbindungsstatus. Vgl. 13.2.
	Alternative	0:*	*+Recommendation*	Alternativen nach Verbindungsbruch, vgl. 13.3.5.

Tabelle 134: Beschreibung der Struktur ***ConnectionStatusStructure***

13.3.5. RecommendationStructure

Dieses Element bildet den Dienst „Info bei Anschlussverlust“ ab.

RecommendationStructure			*+Structure*	**Enthält Alternativenvorschläge bei Anschlussverlust, die über eine bloße Verbindungsalternative hinausgehen. Als Alternative kann neben anderen bestehenden Fahrten auf bereitgestellte Ersatzverkehre, Übernachtungsmöglichkeiten oder andere, durch Disposition neu zu schaffende, Möglichkeiten hingewiesen werden.**
	RecommendationId	**1:1**	*xs:NMTOKEN*	ID der Anfrage für spätere Referenzierung bzw. für Debug-Zwecke
	Text	**1:1**	*+InternationalText*	Beschreibung der Alternative.
	Type	**1:1**	*RecommendationTypeEnumeration*	Typ der Empfehlung, vgl. 13.2.

Tabelle 135: Beschreibung der Struktur ***RecommendationStructure***

13.4. Anfragestrukturen Anschlussmeldung

13.4.1. ConnectionDemandRequestStructure

ConnectionDemandRequestStructure			*+Structure*	**Beinhaltet die Anfragedaten für eine Meldung von Umsteigern, Einsteigern oder Aussteigern bei regulären oder Bedarfsverkehren.**
	RequestId	**1:1**	*xs:NMTOKEN*	ID der Anfrage für spätere Referenzierung bzw. für Debug-Zwecke
	Connection	**1:1**	*+Generalized Connection*	Anschlussrelation. Vgl. 13.3.3.
	NumberOfPersons	0:1	*xs:positiveInteger*	Anzahl der Um-, Ein- oder Aussteiger.
	TravelProbability	0:1	*Percent*	(Kumulierte) Reisewahrscheinlichkeit für die angegebenen Um-, Ein- oder Aussteiger. Vgl. 7.2.1.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 136: Beschreibung der Struktur ***ConnectionDemandRequestStructure***.

13.4.2. ConnectionDemandDeleteRequestStructure

ConnectionDemandDeleteRequestStructure			*+Structure*	Anfragestruktur, um eine frühere Anschlussvoranmeldung zu stornieren.
	RequestId	**1:1**	*xs:NMTOKEN*	ID der Anfrage, die storniert werden soll.

Tabelle 137: Beschreibung der Struktur ***ConnectionDemandDeleteRequestStructure***.

13.5. Antwortstrukturen Anschlussmeldung

13.5.1. ConnectionDemandResponseStructure

ConnectionDemandResponseStructure			*+Structure*	**Antwort auf eine Anfrage vom Typ ConnectionDemandRequest**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Vgl. auch 7.4.2.

Tabelle 138: Beschreibung der Struktur ***ConnectionDemandResponseStructure.***

In *ErrorMessage* können folgende Fehlerzustände auftreten:

CONNECTIONDEMAND_FEEDER_UNKNOWN	Der Zubringer ist der EKAP nicht bekannt.
CONNECTIONDEMAND_DISTRIBUTOR_UNKNOWN	Der Abbringer ist der EKAP nicht bekannt.
CONNECTIONDEMAND_DEPARTURE_BEFORE_ARRIVAL	Die Sollabfahrt des Abbringers liegt vor der Sollankunft des Zubringers. Umstieg folglich nicht möglich.
CONNECTIONDEMAND_FEEDER_LOCATION_UNKNOWN	Der referenzierte Ort des Umstiegs für den Zubringer ist unbekannt.
CONNECTIONDEMAND_DISTRIBUTOR_LOCATION_UNKNOWN	Der referenzierte Ort des Umstiegs für den Abbringer ist unbekannt.

Tabelle 139: Liste der Fehlerzustände in ***ConnectionDemandResponse***.

13.5.2. ConnectionDemandDeleteResponseStructure

ConnectionDemandDeleteResponseStructure			*+Structure*	**Antwort auf eine Anfrage vom Typ ConnectionDemandDeleteRequest.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Vgl. auch 7.4.2.

Tabelle 140: Beschreibung der Struktur ***ConnectionDemandDeleteResponseStructure.***

In *ErrorMessage* können folgende Fehlerzustände auftreten:

CONNECTIONDEMAND_REQUESTID_UNKNOWN	Die Anfrage-ID ist der EKAP nicht bekannt.
CONNECTIONDEMAND_DELETIONNOTPOSSIBLE	Die Anschlussmeldung konnte nicht storniert werden.

Tabelle 141: Liste der Fehlerzustände in ConnectionDemandDeleteResponse.

13.6. Anfragestrukturen Anschlussstatus

13.6.1. ConnectionStatusRequestStructure

ConnectionStatusRequestStructure			*+Structure*	**Dient der aktiven Abfrage eines Verbindungsstatus.**
	RequestId	**1:1**	*xs:NMTOKEN*	ID der Anfrage für spätere Referenzierung bzw. für Debug-Zwecke.
	Connection	**1:1**	*+GeneralizedConnection*	Anschlussrelation. Vgl. 13.3.3.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 142: Beschreibung der Struktur ***ConnectionStatusRequestStructure***

13.6.2. ConnectionStatusNotificationStructure

ConnectionStatusNotificationStructure			+Structure	**Push-Information zu Anschlussstatus. Genutzt von TripMonitoring oder optional aktiv gesendet von EKAP nach ConnectionDemands.**
	ConnectionStatus	**1:1**	*+ConnectionStatus*	Enthält den tatsächlichen Verbindungsstatus. Vgl. 13.3.4.

Tabelle 143: Beschreibung der Struktur ***ConnectionStatusNotificationStructure***

13.7. Antwortstrukturen Anschlussstatus

13.7.1. ConnectionStatusResponseStructure

ConnectionStatusResponseStructure			*+Structure*	**Liefert den Verbindungsstatus oder eine Fehlermeldung auf die Anfrage ConnectionStatus-Request.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Vgl. 7.4.2.
	ConnectionStatus	0:1	*+ConnectionStatus*	Enthält den tatsächlichen Verbindungsstatus. Vgl. 13.3.4.

Tabelle 144: Beschreibung der Struktur ***ConnectionStatusResponseStructure***

In *ErrorMessage* können folgende Fehlerzustände auftreten:

CONNECTIONSTATUS_FEEDER_UNKNOWN	Der Zubringer ist der EKAP nicht bekannt.
CONNECTIONSTATUS_DISTRIBUTOR_UNKNOWN	Der Abbringer ist der EKAP nicht bekannt.
CONNECTIONSTATUS_DEPARTURE_BEFORE_ARRIVAL	Die Sollabfahrt des Abbringers liegt vor der Sollankunft des Zubringers. Umstieg folglich nicht möglich.
CONNECTIONSTATUS_FEEDER_LOCATION_UNKNOWN	Der referenzierte Ort des Umstiegs für den Zubringer ist unbekannt.
CONNECTIONSTATUS_DISTRIBUTOR_LOCATION_UNKNOWN	Der referenzierte Ort des Umstiegs für den Abbringer ist unbekannt.

Tabelle 145: Liste der Fehlerzustände in ConnectionStatusResponseStructure.

13.8. Anfragestrukturen Anschlussrückmeldung

13.8.1. ConnectionReportRequestStructure

ConnectionReportRequestStructure			*+Structure*	**Beinhaltet die Daten für eine Meldung, die besagt, ob der Anschluss für den Reisenden zustande gekommen ist.**
	RequestId	**1:1**	*xs:NMTOKEN*	ID der Anfrage für spätere Referenzierung bzw. für Debug-Zwecke.
	Connection	**1:1**	*+Generalized Connection*	Anschlussrelation, über die berichtet wird. Vgl. 13.3.3.
	Succeeded	**1:1**	*xs:boolean*	Gibt an, ob der Anschluss aus Sicht des Reisenden zustande gekommen ist.
	Reason	0:1	*+International Text*	Optional ist die Angabe eines Grundes für den Anschlussbruch oder das Erreichen des Anschlusses möglich.
	Extensions	0:1	*xs:anyType*	Erweiterungen.

Tabelle 146: Beschreibung der Struktur ***ConnectionReportRequestStructure***.

13.9. Antwortstrukturen Anschlussrückmeldung

13.9.1. ConnectionReportResponseStructure

ConnectionReportResponseStructure			*+Structure*	**Antwort auf eine Anfrage vom Typ Connection-ReportRequest**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Vgl. 7.4.2.

Tabelle 147: Beschreibung der Struktur ***ConnectionReportResponseStructure***

In *ErrorMessage* können folgende Fehlerzustände auftreten:

CONNECTIONRE-PORT_FEEDER_UNKNOWN	Der Zubringer ist der EKAP nicht bekannt.
CONNECTIONREPORT _DISTRIBUTOR_UNKNOWN	Der Abbringer ist der EKAP nicht bekannt.
CONNECTIONREPORT _DEPARTURE_BEFORE_ARRIVAL	Die Sollabfahrt des Abbringers liegt vor der Sollankunft des Zubringers. Umstieg folglich nicht möglich.
CONNECTIONREPORT _FEEDER_LOCATION_UNKNOWN	Der referenzierte Ort des Umstiegs für den Zubringer ist unbekannt.
CONNECTIONREPORT _DISTRIBUTOR_LOCATION_UNKNOWN	Der referenzierte Ort des Umstiegs für den Abbringer ist unbekannt.

Tabelle 148: Liste der Fehlerzustände in ConnectionReportResponse.

14. Dienst Fahrpreis- und Tarifberechnung

14.1. Beschreibung

Dieser Dienst stellt allgemeine, haltestellenbezogene oder verbindungsbezogene Tarifinformationen bereit.

14.2. Anfragestrukturen

Eine Anfrage an den Tarifberechnungsdienst wird mittels eines Elements ***FaresRequest*** vom Typ *FaresRequestStructure* gestellt.

14.2.1. FaresRequestStructure

FaresRequestStructure				*+Structure*	**Fasst die Daten für eine Tarifanfrage zusammen.**
	a	***StopFaresRequest***	**-1:1**	*+StopFaresRequest*	Haltestellenbezogene Tarifanfrage. Vgl. 14.2.2.
	b	***StaticFaresRequest***		*+StaticFaresRequest*	Allgemeine Tarifanfrage. Vgl. 14.2.3
	c	***TripFaresRequest***		*+TripFaresRequest*	Verbindungsbezogene Tarifanfrage. Vgl. 14.2.3..
	d	***MultiTripFaresRe-quest***		*+MultiTripFaresRequest*	Aggregierte Tarifanfrage für mehrere Verbindungen. Vgl. 14.2.5.
	Params		0:1	*+ FaresParam*	Parameter für Tarifanfrage. Vgl. 7.10.8.
	Extension		0:1	*xs:anyType*	Erweiterungen.

Tabelle 149: Beschreibung der Struktur ***FaresRequestStructure***.

14.2.2. StopFaresRequestStructure

Die haltestellenbezogene Tarifanfrage ermittelt Tarifinformationen, die für eine bestimmte Haltestelle gelten, z. B. die Tarifzonen, in denen die Haltestelle liegt.

StopFaresRequestStructure			*+Structure*	**Fasst die Daten für eine haltestellenbezogene Tarifanfrage zusammen.**
	StopPointRef	**1:1**	*→StopPoint-Code*	Referenziert den Haltepunkt. Vgl. 7.5.1.
	Date	0:1	*xs:date*	Stichtag für die Gültigkeit der Tarifauskunft.

Tabelle 150: Beschreibung der Struktur ***StopFaresRequestStructure***.

14.2.3. StaticFaresRequestStructure

Die statische Tarifanfrage ermittelt allgemeine Tarifinformationen wie z. B. eine Liste der erhältlichen Fahrscheinarten oder eine URL auf weiterführende Tarifinformationen (z. B. Tarifzonenpläne, Tarifbestimmungen etc.).

StaticFaresRequestStructure			*+Structure*	**Fasst die Daten für eine allgemeine (statische) Tarifanfrage zusammen.**
	Date	0:1	*xs:date*	Stichtag für die Gültigkeit der Tarifauskunft.
	TicketRef	0:*	*→TicketCode*	Code von Tickets, für die weitere Informationen angefragt werden. Falls kein TicketRef angegeben wird, soll der Server über alle verfügbaren Tickets Auskunft geben.

Tabelle 151: Beschreibung der Struktur ***StaticFaresRequestStructure***.

14.2.4. TripFaresRequestStructure

Die verbindungsbezogene Tarifanfrage ermittelt zu einer bestimmten Verbindung die in Frage kommenden Fahrscheine und ihre Preise.

TripFaresRequestStructure			*+Structure*	**Fasst die Daten für eine verbindungsbezogene Tarifanfrage zusammen.**
	Trip	**1:1**	*+Trip*	Beinhaltet die Verbindung, für die die Tarifauskunft zu ermitteln ist. Vgl. 9.3.4.

Tabelle 152: Beschreibung der Struktur ***TripFaresRequestStructure***.

14.2.5. MultiTripFaresRequestStructure

Der Unterschied von *MultiTripFaresRequestStructure* zu *TripFaresRequestStructure* besteht darin, dass der Server bei *MultiTripFaresRequestStructure* aufgefordert ist, eine möglichst günstige Ticketkombination herauszufinden, die die Verbindungen abdeckt, also z. B. eine Tageskarte, wenn genügend Verbindungen am gleichen Tag stattfinden sollen.

MultiTripFaresRequestStructure			*+Structure*	**Fasst die Daten für eine Tarifanfrage für mehrere Verbindungen zusammen.**
	Trip	**1:***	*+Trip*	Beinhaltet die Verbindungen, für die die Tarifauskunft zu ermitteln ist. Vgl. 9.3.4.

Tabelle 153: Beschreibung der Struktur ***MultiTripFaresRequestStructure***.

14.3. Antwortstrukturen

Das Ergebnis einer Tarifanfrage wird mittels eines Elements ***FaresResponse*** vom Typ *FaresResponseStructure* übertragen.

14.3.1. FaresResponseStructure

FaresResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Tarifanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	FaresResult	0:*	*+FaresResult*	Struktur für ein Tarifergebnis. Vgl. 14.3.2.

Tabelle 154: Beschreibung der Struktur ***FaresResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

FARES_DATEOUTOFRANGE	Die Tarifanfrage kann nicht bearbeitet werden, weil für das gewünschte Datum keine Informationen zur Verfügung stehen.
FARES_STOPPOINTUNKNOWN	Die Tarifanfrage kann nicht bearbeitet werden, weil der angefragte Haltepunkt unbekannt ist.

Tabelle 155: Liste der Fehlerzustände in ***FaresResponse***.

14.3.2. FaresResultStructure

FaresResultStructure				*+Structure*	**Resultatstruktur für die Buchungsinformationen.**
	ResultId		**1:1**	*xs:NMTOKEN*	ID des Resultats für spätere Referenzierung.
	a	***StopFaresResult***	**-1:1**	*+StopFaresResult*	Antwort auf haltestellenbezogene Tarifanfrage. Vgl. 14.3.3.
	b	***StaticFaresResult***		*+StaticFaresResult*	Antwort auf allgemeine Tarifanfrage. Vgl. 14.3.4.
	c	***TripFaresResult***		*+TripFaresResult*	Antwort auf verbindungsbezogene Tarifanfrage. Vgl. 7.10.6.
	d	***MultiTripFaresResult***		*+MultiTripFaresResult*	Antwort auf Tarifanfrage für mehrere Verbindungen. Vgl. 14.3.6.

Tabelle 156: Beschreibung der Struktur ***FaresResultStructure***.

14.3.3. StopFaresResultStructure

StopFaresResultStructure			*+Structure*	**Resultatstruktur für haltestellenbezogene Tarifinformationen.**
	FareZoneListInArea	**1:***	*+FareZoneListInArea*	Liste der Tarifzonen, in denen die angefragte Haltestelle liegt. Vgl. 7.10.3.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 157: Beschreibung der Struktur ***StopFaresResultStructure***.

14.3.4. StaticFaresResultStructure

StaticFaresResultStructure			*+Structure*	**Resultatstruktur für allgemeine Tarifinformationen.**
	Ticket	0:*	*+Ticket*	Liste der verfügbaren Tickets. Vgl. 7.10.5.
	StaticInfoUrl	0:1	*+WebLinkI*	Links auf Informationsseiten im Web (vgl. 7.2.4).
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 158: Beschreibung der Struktur ***StaticFaresResultStructure***.

14.3.5. TripTicketReferenceStructure

TripTicketReferenceStructure			*+Structure*	**Verknüpfung von Tickets mit Verbindungen (oder Teilen davon).**
	TicketRef	**1:1**	*→TicketCode*	Referenz auf ein Ticket.
	FromTripIdRef	**1:1**	*xs:NMTOKEN*	Referenz auf eine Verbindung, ab der ein Ticket gültig ist.
	FromTripLegIdRef	0:1	*xs:NMTOKEN*	Referenz auf einen Teilweg, ab dem das Ticket gültig ist.
	ToTripIdRef	**1:1**	*xs:NMTOKEN*	Referenz auf eine Verbindung, bis zu der ein Ticket gültig ist.
	ToTripLegIdRef	0:1	*xs:NMTOKEN*	Referenz auf einen Teilweg, bis zu dem das Ticket gültig ist.

Tabelle 159: Beschreibung der Struktur ***TripTicketReferenceStructure***.

14.3.6. MultiTripFaresResultStructure

MultiTripFaresResultStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Tarifauskunft zu mehreren Verbindungen zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf diese Tarifauskunft. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
	TripTicketReference	**1:***	*+TripTicketReference*	Verknüpfung von Tickets mit Verbindungen (oder Teilen davon). Vgl. 14.3.5.
	Ticket	0:*	*+Ticket*	Fahrscheine, die auf diesem Abschnitt der Verbindung gültig sind (vgl. 7.10.5).
	PassedZones	0:1	*+FareZoneListInArea*	Die Menge der durchfahrenen Tarifzonen, über alle Verbindungen gesehen (vgl. 7.10.3).
	StaticInfoURL	0:*	*+WebLink*	URL zu Informationsseiten (vgl. 7.2.4).

Tabelle 160: Beschreibung der Struktur ***MultiTripFaresResultStructure.***

In *ErrorMessage* können folgende Fehlerzustände auftreten:

FARES_OUTOFAREA	Der in der Verbindungsauskunft gefundene Weg verlässt das Tarifgebiet.
FARES_JOURNEYNOTPERMITTED	Ein in der Verbindungsauskunft benutztes Verkehrsmittel ist für den Tarif nicht zulässig.
FARES_ADDITIONALCHARGES	Zusätzliche Gebühren sind voraussichtlich zu entrichten (z. B. Mautzuschläge oder Reservierungsgebühren).
FARES_ADDITIONALTICKETS	Zusätzliche Fahrscheine sind notwendig, da nicht für alle Verkehrsmittel oder für alle angegebenen Verbindungen ein passender Fahrschein ermittelt werden konnte.
FARES_ROUTENOTFEASIBLE	Es kann kein Fahrschein ermittelt werden, weil die Route der Verbindungsauskunft nicht in Einklang mit den Tarifregeln steht (z. B. wegen Rundreisen, Stichfahrten oder Überschreitung der zulässigen Gesamtdauer).

Tabelle 161: Liste der Fehlerzustände in ***MultiTripFaresResultStructure***.

15. Dienst Buchungsinformationen

15.1. Beschreibung

Der Dienst Buchungsinformation stellt Informationen zur Verfügung, mit deren Hilfe Kontakt zu einem Buchungssystem hergestellt werden kann. Das zuständige Buchungssystem kann für ein Verkehrsunternehmen oder für eine einzelne ÖV-Fahrt abgefragt werden. Ein Buchungssystem führt z. B. die Vorbestellung eines Bedarfsverkehrs, eine Sitzplatzreservierung oder auch den Kauf eines Fahrscheins durch.

15.2. Anfragestrukturen

Eine Buchungsinformation wird mittels eines Elements ***BookingInfoRequest*** vom Typ *BookingInfoRequestStructure* angefordert.

15.2.1. BookingInfoRequestStructure

In einer *BookingInfoRequestStructure* kann wahlweise eine bestimmte ÖV-Fahrt oder ein Verkehrsunternehmen angegeben werden, für die die Buchungsinformation ermittelt werden soll.

BookingInfoRequestStructure				*+Structure*	**Fasst die Anfragedaten nach Buchungsinformationen zusammen.**
	a	***Service***	**-1:1**	*+DatedJourney*	Definition einer ÖV-Fahrt an einem bestimmten Tag. Vgl. 7.6.2.
	b	***OperatorRef***		*→Operator*	Referenz auf ein Verkehrsunternehmen. Vgl. 7.4.1.
	Extension		0:1	*xs:anyType*	Erweiterungen.

Tabelle 162: Beschreibung der Struktur ***BookingInfoRequestStructure***.

15.3. Antwortstrukturen

Das Ergebnis einer Buchungsinformationsanfrage wird mittels eines Elements ***BookingInfoResponse*** vom Typ *BookingInfoResponseStructure* übertragen.

15.3.1. BookingInfoResponseStructure

BookingInfoResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Buchungsinformationsanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	BookingInfoResult	0:1	*+BookingInfoResult*	Struktur für ein Buchungsinformationsresultat. Vgl. 15.3.2.

Tabelle 163: Beschreibung der Struktur ***BookingInfoResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

BOOKINGINFO_DATEINVALID	Zum angegebenen Datum liegen keine Informationen vor.
BOOKINGINFO_VEHICLEUNKNOWN	Das angegebene Fahrzeug ist unbekannt.
BOOKINGINFO_OPERATORUNKNOWN	Das angegebene Verkehrsunternehmen ist unbekannt.
BOOKINGINFO_JOURNEYUNKNOWN	Die angegebene Fahrt ist unbekannt.
BOOKINGINFO_LINEUNKNOWN	Die angegebene Linie ist unbekannt.
BOOKINGINFO_MODEUNKNOWN	Die angegebene Verkehrsmittelart ist unbekannt.
BOOKINGINFO_NOINFORMATION	Es liegt keine passende Information vor.

Tabelle 164: Liste der Fehlerzustände in ***BookingInfoResponse***.

15.3.2. BookingInfoResultStructure

BookingInfoResultStructure			*+Structure*	**Resultatstruktur für die Buchungsinformationen.**
	BookingInfo	**1:***	*+BookingInfo*	Container für die Buchungsinformationen. Vgl. 7.10.4.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 165: Beschreibung der Struktur ***BookingInfoResultStructure***.

16. Dienst IV-Routing

In der XML-Schema-Definition Trias_IndividualTrips.xsd werden Datentypen und Strukturen definiert, die für ein IV-Routing verwendet werden.

16.1. Einfache Typen

Folgender einfacher Typ (simple type) wird definiert:

IndividualTripsAlgorithmTypeEnumeration	*fastest* | *shortest* | *beautiful* | *optimal* | *economic*	Algorithmus-Typ für die Berechnung von IV-Routen.

Tabelle 166: Liste der einfachen Typdefinitionen in Trias_IndividualTrips.xsd.

16.2. Anfragestrukturen

Eine Route im Individualverkehr wird mittels eines Elements ***IndividualRouteRequest*** vom Typ *IndividualRouteRequestStructure* angefordert.

16.2.1. IndividualRouteRequestStructure

IndividualRouteRequestStructure			*+Structure*	**Fasst die Anfragedaten für ein IV-Routing zusammen.**
	Origin	**1:1**	*+IndividualRouteLocationContext*	Ortsdaten für den Abfahrtsort. Vgl. 16.2.3.
	Destination	**1:1**	*+IndividualRouteLocationContext*	Ortsdaten für den Zielort. Vgl. 16.2.3.
	Via	0:*	*+Via*	Ein oder mehrere Via-Orte. Die angegebenen Via-Orte müssen in der vorgegebenen Reihenfolge erreicht werden. Der Server darf eine Via-Haltestelle durch eine äquivalente Haltestelle ersetzen. Vgl. 7.6.2.
	Mode	**1:***	*+IndividualTransportOptions*	IV-Modi, für die eine IV-Route ermittelt werden soll. Für jeden IV-Modus können weitere steuernde Parameter angegeben werden. Vgl. 7.3.2.
	Params	0:1	*+IndividualTripParam*	Parameter, die die Suche und Rückgabewerte beeinflussen können. Vgl. 16.2.2.

Tabelle 167: Beschreibung der Struktur ***IndividualRouteRequestStructure***.

16.2.2. IndividualTripParamStructure

IndividualTripParamStructure			*+Structure*	**Fasst die Parameter zusammen, die die IV-Routen-Suche und Rückgabewerte beeinflussen können. Diese Parameter gelten für alle IV-Modi, für die das IV-Routing durchgeführt werden soll. Sollen unterschiedliche Parametersätze für unterschiedliche IV-Modi verwendet werden, müssen mehrere unabhängige IV-Routensuchen durchgeführt werden.**
BaseTripMobilityFilter	*NoSingleStep*	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Stufe bewältigen kann. Falls nein, wird dieser Parameter gesetzt.
	NoStairs	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Treppe bewältigen kann. Falls nein, wird dieser Parameter gesetzt.
	NoEscalator	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Rolltreppe benutzen kann. Falls nein, wird dieser Parameter gesetzt.
	NoElevator	0:1	*xs.boolean*	Legt fest, ob der Benutzer einen Aufzug benutzen kann. Falls nein, wird dieser Parameter gesetzt.
	NoRamp	0:1	*xs.boolean*	Legt fest, ob der Benutzer eine Rampe bewältigen kann. Falls nein, wird dieser Parameter gesetzt.

Base-TripPolicy				
	a *NumberOfResults*	-0:1	*xs:positiveInteger*	Anzahl der Verbindungsauskünfte, die der Benutzer mindestens erwartet.
	b *:::*	-0:1	*NumberOfResultsGroup*	Angabe der gewünschten Verbindungen vor/nach dem angegebenen Zeitpunkt am Start oder Ziel (vgl. 9.2.3).
	IgnoreRealtimeData	0:1	*xs:boolean*	Wenn dieser Parameter gesetzt ist, sollen in der Verbindungssuche keine Echtzeitdaten oder Störungsinformationen sondern nur Sollfahrplandaten berücksichtigt werden.
	ImmediateTripStart	0:1	*xs:boolean*	Wenn dieser Parameter gesetzt ist, soll die zu suchende Verbindung unmittelbar an der angegebenen Startsituation beginnen. Eine Optimierung der Abfahrtszeit am Start nach der Regel „Starte so spät wie möglich, solange nur die gleiche Ankunftszeit am Ziel gewährleistet ist“ ist dann nicht notwendig.
IndividualTripPolicy	*AlgorithmType*	0:1	*fastest* | *shortest* | *beautiful* | *optimal* | *economic*	Art der Zielfunktion, nach der der Routing-Algorithmus die Route optimieren soll. Falls nicht vorgegeben, benutzt der Dienst seine eigene Voreinstellung.
	BanMotorways	0:1	*xs:boolean*	Falls gesetzt, sollen im IV-Routing keine Autobahnen genutzt werden. Default ist *false*.
	BanTollRoads	0:1	*xs:boolean*	Falls gesetzt, sollen im IV-Routing keine Mautstraßen genutzt werden. Default ist *false*.
	BanFerries	0:1	*xs:boolean*	Falls gesetzt, sollen im IV-Routing keine Fähren oder Schiffe genutzt werden. Default ist *false*.
	BanTunnels	0:1	*xs:boolean*	Falls gesetzt, sollen im IV-Routing keine Tunnel (Unterführungen aber schon) genutzt werden. Der IV-Routendienst legt den Unterschied zwischen Tunnel und Unterführung selbst fest. Default ist *false*.
	BanBridges	0:1	*xs:boolean*	Falls gesetzt, sollen im IV-Routing keine größeren Brücken (Überführungen aber schon) genutzt werden. Der IV-Routendienst legt selbst fest, welche Brücken als „größer“ gelten. Default ist *false*.
	AllowUnpavedRoads	0:1	*xs:boolean*	Falls gesetzt, dürfen im IV-Routing auch nicht-befestigte Straßen genutzt werden, sonst nicht. Default ist *false*.
BaseTripContentFilter	*IncludeTrackSections*	0:1	*xs:boolean*	Legt fest, ob im Resultat TrackSection-Elemente (vgl. 9.3.13) für die detaillierte geographische Beschreibung des Wegs mitausgegeben werden sollen. Default ist *false*.
	IncludeLegProjection	0:1	*xs:boolean*	Legt fest, ob im Resultat der detaillierte geographische Verlauf des Wegs als Koordinatenfolge mitausgegeben werden soll. Default ist *false*.
	IncludeTurnDescription	0:1	*xs:boolean*	Legt fest, ob im Resultat Routenhinweise mit Abbiegeempfehlungen mitausgegeben werden sollen. Default ist *false*.
	IncludeAccessibility	0:1	*xs:boolean*	Legt fest, ob im Resultat Informationen zur Barrierefreiheit mitausgegeben werden sollen. Default ist *false*.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 168: Beschreibung der Struktur ***IndividualTripParamStructure***.

16.2.3. IndividualRouteContextStructure

Elemente vom Typ IndividualRouteContextStructure werden vor allem dazu benutzt, um den Start- bzw. Ziel-Kontext zu beschreiben, der für den Reisenden am Anfang bzw. Ende seiner Reise angenommen werden soll. Beim Dienst IV-Routing dienen Elemente dieses Typs als Angabe von Start und Ziel. Die Implementierung des Suchalgorithmus ist dabei selbst dafür zuständig, die Ortsangaben, etwa eine Koordinate, auf die internen Elemente (z. B. Knoten oder Kanten) des Such-Netzwerks abzubilden.

IndividualRouteContextStructure			*+Structure*	**Ortsangabe für Start- oder Zielorte von IV-Routen.**
	LocationRef	**1:1**	*+LocationRef*	Referenz auf ein Ortsobjekt. Vgl. 7.5.11.
	DepArrTime	0:1	*xs:dateTime*	Abfahrts- bzw. Ankunftszeit.

Tabelle 169: Beschreibung der Struktur ***IndividualRouteContextStructure***.

16.3. Antwortstrukturen

Das Ergebnis einer IV-Routing-Anfrage wird mittels eines Elements ***IndividualRoute-Response*** vom Typ *IndividualRouteResponseStructure* übertragen.

16.3.1. IndividualRouteResponseStructure

IndividualRouteResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine IV-Routing-Auskunft zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
	IndividualRouteResponseContext	0:1	*+TripResponseContext*	Container für Daten, die in der Antwort mehrfach auftreten und referenziert werden. Vgl. 9.3.2.
	RouteResult	0:*	*+RouteResult*	Container für eine Verbindungsauskunft. Vgl. 16.3.2.

Tabelle 170: Beschreibung der Struktur ***IndividualRouteResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

ROUTE_NOROUTEFOUND	Zu den angegebenen Start- und Zielorten, der gewünschten Abfahrts- oder Ankunftszeit sowie unter Berücksichtigung der gegebenen Parameter konnte keine IV-Route gefunden werden.
ROUTE_ORIGINUNKNOWN	Der angegebene Ort (Adresse, Haltestelle, …) für den Start der IV-Route ist unbekannt.
ROUTE_DESTINATIONUNKNOWN	Der angegebene Ort (Adresse, Haltestelle, …) für das Ziel der IV-Route ist unbekannt.
ROUTE_VIAUNKNOWN	Einer der angegebenen Via-Punkte ist unbekannt.
ROUTE_ORIGINDESTINATIONIDENTICAL	Start und Ziel sind gleich.
ROUTE_UNSUPPORTEDMODE	Einer der angefragten IV-Modi wird nicht unterstützt.
ROUTE_UNSUPPORTEDMOBILITYFILTER	Einer der angefragten Mobilitätsfilter wird nicht unterstützt.
ROUTE_UNSUPPORTEDALGORITHM	Der angefragte Algorithmustyp wird nicht unterstützt.
ROUTE_UNSUPPORTEDBAN	Einer der angefragten Ausschlussfilter (Autobahn, Mautstraßen, Fähren) wird nicht unterstützt.
ROUTE_NODATETIME	Weder Abfahrts- noch Ankunftszeit wurden angegeben.
ROUTE_DATETIMEERROR	Datum und/oder Uhrzeit sind unverständlich.
ROUTE_DEPARTUREAFTERARRIVAL	Die gewünschte Abfahrtszeit an allen Startpunkten liegt nach der gewünschten Ankunftszeit an allen Zielpunkten.
ROUTE_DATEOUTOFRANGE	Für das angefragte Datum liegen keine Routingdaten vor, z. B. weil das Datum in der Vergangenheit oder weit in der Zukunft liegt.

Tabelle 171: Liste der Fehlerzustände in ***IndividualRouteResponse***.

16.3.2. RouteResultStructure

RouteResultStructure			*+Structure*	**Fasst die Ergebnisdaten für eine einzelne IV-Route zusammen.**
	ResultId	**1:1**	*xs:NMTOKEN*	ID des Resultats für spätere Referenzierung bzw. für Debug-Zwecke.
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf diese IV-Route. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. auch 7.4.2.
	Route	**1:1**	*+Route*	Daten zu einer IV-Route. Vgl. 16.3.3.

Tabelle 172: Beschreibung der Struktur ***RouteResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

ROUTE_MODEPARAMETERSIGNORED	Bei dieser IV-Route wurde mindestens einer der Parameter für diesen IV-Modus ignoriert. Vgl. auch 7.3.2.
ROUTE_MOBILITYFILTERIGNORED	Bei dieser IV-Route wurde mindestens einer der Mobilitätsfilter ignoriert.
ROUTE_BANIGNORED	Bei dieser IV-Route wurde mindestens einer der Ausschlussfilter (Autobahn, Mautstraßen, Fähren) ignoriert.

Tabelle 173: Liste der Fehlerzustände in ***RouteResult***.

16.3.3. RouteStructure

RouteStructure			*+Structure*	**Daten zu einer einzelnen IV-Route.**
	RouteId	**1:1**	*xs:NMTOKEN*	ID der Verbindung für spätere Referenzierung bzw. für Debug-Zwecke.
	Duration	**1:1**	*xs:duration*	Gesamtdauer der IV-Route.
	StartTime	**1:1**	*xs:dateTime*	Startzeitpunkt der IV-Route.
	EndTime	**1:1**	*xs:dateTime*	Endzeitpunkt der IV-Route.
	Distance	0:1	*Distance*	Gesamtdistanz der IV-Route als Länge des zurückzulegenden Weges.
	RouteLeg	**1:***	*+ContinuousLeg*	Teilwege dieser IV-Route. Es muss genau ein RouteLeg mehr existieren, als Vias angefragt wurden. Für IV-Routen ist in *ContinuousLeg.Service* lediglich *IndividualMode* und ggf. *SituationFullRef* gefüllt. Vgl. 9.3.8.
	SituationFullRef	0:*	*+SituationFullRef*	Verweis auf Störungsnachrichten. Diese Nachrichten können im Element *IndividualRouteResponseContext* vom Typ *TripResponseContext* (vgl. 9.3.2) zu finden sein oder auf anderem Wege bekannt gemacht werden. Vgl. 7.8.2.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 174: Beschreibung der Struktur ***RouteStructure***

17. Dienst Kartendienst

In der XML-Schema-Definition Trias_Maps.xsd werden Datentypen und Strukturen definiert, die für den Kartendienst verwendet werden.

17.1. Einfache Datentypen

Folgende einfache Typen (simple types) werden definiert:

MapLayersEnumeration	*physical* | *satellite* | *street* | *rail* | *names* | *stops* | *traffic*	Zusätzliche Layer in der Karte.

Tabelle 175: Liste der einfachen Typdefinitionen in Trias_Maps.xsd.

17.2. Anfragestrukturen

17.2.1. MapServiceRequestStructure

Eine Karte wird mittels eines Elements ***MapServiceRequest*** vom Typ *MapServiceRequest-Structure* angefordert. Dieser liefert eine Bilddatei zurück, welche die angeforderte Karte enthält. Sollen zusätzliche Objekte auf Karten eingezeichnet (z. B. Haltestellen) oder aktive Elemente integriert werden (z. B. für Mouse-Over-Effekte, für Verlinkungen, usw.), dann muss dies durch den Client auf Basis einer vom Kartendienst generierten Hintergrundkarte gemacht werden.

MapServiceRequestStructure			*+Structure*	**Fasst die Anfragedaten für den Abruf einer Karte zusammen.**
MapProperties	***Aspect***	**1:1**	*+MapAspect*	Geografischer Ausschnitt der zu generierenden Karte. Die generierte Karte darf einen anderen Kartenausschnitt als angegeben umfassen. Allerdings muss der Mittelpunkt nahezu unverändert bleiben und der tatsächliche Kartenausschnitt soll dem angefragten so ähnlich wie möglich sein. Vgl. 17.2.4.
	Size	**1:1**	*+MapSize*	Bildgröße der zu generierenden Karte. Der Kartendienst muss in der Lage sein, Bildgrößen bis mindestens 1920x1080 Bildpunkten („FullHD“) zu erzeugen. Die generierte Karte muss exakt dieser vorgegebenen Bildgröße entsprechen. Vgl. 17.2.5.
	ImageType	0:1	*xs:string*	Datenformat der zu generierenden Karte. Dieser muss als Medientyp (ehemals MIME-Typ) eines Bilddatenformates (Subtypen des Typs „image“) angegeben werden. Die Liste erlaubter Werte wird durch die IANA definiert. Falls nicht angegeben, muss der Kartendienst “image/png“ nutzen.
	Layer	**1:***	*physical* | *satellite* | *street* | *rail* | *names* | *stops* | *traffic*	Gibt die Layer der Karte an. Dazu zählt die Hintergrundkarte, aber auch zusätzliche Informationen, die auf der Karte integriert werden sollen.
	Opaqueness	0:1	*Percent*	Undurchsichtigkeit des Kartenhintergrundes, soweit kein HIntergrundlayer gewählt wurde. Zwischen 0 (ganz durchsichtig) und 100 (ganz undurchsichtig).
	BackgroundColor	0:1	*xs:string*	Farbe des Kartenhintergrundes, soweit kein Hintergrundlayer gewählt wurde. Erlaubte Werte sind alle Farbangaben, die dem CSS3-Standard der W3C entsprechen.
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 176: Beschreibung der Struktur ***MapServiceRequestStructure***.

17.2.2. ImageCoordinatesRequestStructure

In einigen Anwendungsfällen sollen geografische Objekte auf einer Karte eingezeichnet werden, oder als aktives, ggf. sogar verschiebbares Objekt zur Verfügung stehen. Um dies zu erreichen, muss der Client das Objekt nachträglich über das Kartenbild einzeichnen. Damit er dies tun kann, ist es hilfreich, aus den geografischen Koordinaten eines Objektes Bildkoordinaten generieren zu können. Solche Bildkoordinaten werden mittels eines Elements ***ImageCoordinatesRequest*** vom Typ *ImageCoordinatesRequestStructure* angefordert.

ImageCoordinatesRequestStructure			*+Structure*	**Fasst die Anfragedaten für den Abruf von Bildkoordinaten zusammen.**
MapProperties	***Aspect***	**1:1**	*+MapAspect*	Tatsächlicher geografischer Ausschnitt der referenzierten Karte. Vgl. 17.2.4.
	Size	**1:1**	*+MapSize*	Bildgröße der referenzierten Karte. Vgl. 17.2.5.
	Point	**1:***	*+GeoPosition*	Geografische Punkte, für die Bildkoordinaten berechnet werden sollen. Diese dürfen außerhalb des angegebenen Kartenausschnitt liegen, allerdings kann ein Kartendienst die Bearbeitung bei zu weit außerhalb liegenden Punkten ablehnen. Vgl. 7.2.3.

Tabelle 177: Beschreibung der Struktur ***ImageCoordinatesRequestStructure***.

17.2.3. GeoCoordinatesRequestStructure

Um einer Position auf einem Kartenbild die entsprechende geografische Position zuordnen zu können (z. B. nach dem Klick in eine Karte), benötigt man eine passende Umrechnungsfunktion. Solche geografischen Positionen werden mittels eines Elements ***GeoCoordinatesRequest*** vom Typ *GeoCoordinatesRequestStructure* angefordert.

GeoCoordinatesRequestStructure			*+Structure*	**Fasst die Anfragedaten für den Abruf von geografischen Koordinaten zusammen.**
MapProperties	***Aspect***	**1:1**	*+MapAspect*	Tatsächlicher geografischer Ausschnitt der referenzierten Karte. Vgl. 17.2.4.
	Size	**1:1**	*+MapSize*	Bildgröße der referenzierten Karte. Vgl. 17.2.5.
	ImagePoint	**1:***	*+MapCoordinate*	Bildpunkte, für die geografische Koordinaten berechnet werden sollen. Diese können außerhalb des Kartenbildes liegen, allerdings kann ein Kartendienst die Bearbeitung von zu weit außerhalb liegenden Punkten ablehnen. Vgl. 17.2.6.

Tabelle 178: Beschreibung der Struktur ***GeoCoordinatesRequestStructure***.

17.2.4. MapAspectStructure

MapAspectStructure			*+Structure*	**Geografischer Kartenausschnitt.**
	UpperLeft	**1:1**	*+GeoPosition*	Obere linke Ecke des geografischen Kartenausschnitts. Vgl. 7.2.3.
	LowerRight	**1:1**	*+GeoPosition*	Untere rechte Ecke des geografischen Kartenausschnitts. Vgl. 7.2.3.

Tabelle 179: Beschreibung der Struktur ***MapAspectStructure***.

17.2.5. MapSizeStructure

MapSizeStructure			*+Structure*	**Bildgröße einer Karte.**
	Width	**1:1**	*xs:nonNegativeInteger*	Breite der Karte in Pixeln.
	Height	**1:1**	*xs:nonNegativeInteger*	Höhe der Karte in Pixeln.

Tabelle 180: Beschreibung der Struktur ***MapSizeStructure***.

17.2.6. MapCoordinateStructure

MapCoordinateStructure			*+Structure*	**Koordinaten einen Bildpunktes.**
	X	**1:1**	*xs:integer*	X-Koordinate. Kann negativ sein oder größer als die Breite des zugrundeliegenden Bildes.
	Y	**1:1**	*xs:integer*	Y-Koordinate. Kann negativ sein oder größer als die Breite des zugrundeliegenden Bildes.

Tabelle 181: Beschreibung der Struktur ***MapCoordinateStructure***.

17.3. Antwortstrukturen

17.3.1. MapServiceResponseStructure

Das Ergebnis einer Kartenanfrage wird mittels eines Elements ***MapServiceResponse*** vom Typ *MapServiceResponseStructure* übertragen.

MapServiceResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Kartenanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	MapResult	0:1	*+MapResult*	Ergebnis der Kartenanfrage.

Tabelle 182: Beschreibung der Struktur ***MapServiceResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

MAP_UNSUPPORTEDSIZE	Die angefragte Bildgröße wird nicht unterstützt.
MAP_UNSUPPORTEDMEDIATYPE	Der angefragte Medientyp (ehemals MIME-Type) wird nicht unterstützt.
MAP_UNSUPPORTEDASPECT	Der angeforderte Kartenausschnitt liegt außerhalb des vom Kartendienst unterstützten Gebietes, ist zu groß oder ist zu klein.
MAP_LAYERIGNORED	Mindestens einer der angefragten Layer wurde ignoriert.
MAP_UNSUPPORTEDSTYLE	Der Kartendienst unterstützt die Angabe einer Hintergundfarbe oder einer Undurchsichtigkeit nicht.
MAP_NOMAP	Es konnte keine der Anfrage entsprechende Karte generiert werden.

Tabelle 183: Liste der Fehlerzustände in ***MapServiceResponse***.

17.3.2. ImageCoordinatesResponseStructure

Das Ergebnis einer Bildkoordinatenanfrage wird mittels eines Elements ***ImageCoordinates-Response*** vom Typ *ImageCoordinatesResponseStructure* übertragen.

ImageCoordinatesResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Bildkoordinatenanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	Result	0:*	*+ImagePointResult*	Einzelergebnisse der Umrechnung von Geo- zu Bildkoordinaten. Es können maximal soviele Elemente sein, wie geografische Punkte in der Anfrage enthalten waren. Zur Identifizierung ist in jedem Einzelergebnis der angefragte Punkt enthalten. Vgl. 17.3.5.

Tabelle 184: Beschreibung der Struktur ***ImageCoordinatesResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

MAP_UNSUPPORTEDSIZE	Die angefragte Bildgröße wird nicht unterstützt.
MAP_UNSUPPORTEDASPECT	Der angeforderte Kartenausschnitt liegt außerhalb des vom Kartendienst unterstützten Gebietes, ist zu groß oder ist zu klein.
MAP_TOOMANYPOINTS	Es wurden zu viele Punktobjekte zur Umrechnung angegeben.
MAP_UNSUPPORTEDPOINT	Mindestens einer der angefragten Punkte liegt außerhalb des umrechenbaren Gebietes.

Tabelle 185: Liste der Fehlerzustände in ***ImageCoordinatesResponse***.

17.3.3. GeoCoordinatesResponseStructure

Das Ergebnis einer Geokoordinatenanfrage wird mittels eines Elements ***GeoCoordinates-Response*** vom Typ *GeoCoordinatesResponseStructure* übertragen.

GeoCoordinatesResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Geokoordinatenanfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	Result	0:*	*+GeoCoordinateResult*	Einzelergebnisse der Umrechnung von Bild- zu Geokoordinaten. Es können maximal soviele Elemente sein, wie Bildpunkte in der Anfrage enthalten waren. Zur Identifizierung ist in jedem Einzelergebnis der angefragte Punkt enthalten. Vgl. 17.3.6.

Tabelle 186: Beschreibung der Struktur ***GeoCoordinatesResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

MAP_UNSUPPORTEDSIZE	Die angefragte Bildgröße wird nicht unterstützt.
MAP_UNSUPPORTEDASPECT	Der angeforderte Kartenausschnitt liegt außerhalb des vom Kartendienst unterstützten Gebietes, ist zu groß oder ist zu klein.
MAP_TOOMANYPOINTS	Es wurden zu viele Punktobjekte zur Umrechnung angegeben.
MAP_UNSUPPORTEDPOINT	Mindestens einer der angefragten Punkte liegt außerhalb des umrechenbaren Gebietes.

Tabelle 187: Liste der Fehlerzustände in ***GeoCoordinatesResponse***.

17.3.4. MapResultStructure

MapResultStructure			*+Structure*	**Fasst die Daten einer generierten Karte zusammen.**
	File	**1:1**	*xs:base64Binary*	Die Bilddaten der generierten Karte.
	ImageType	**1:1**	*xs:string*	Datenformat der generierten Karte. Dieser muss als Medientyp (ehemals MIME-Typ) eines Bilddatenformates (Subtypen des Typs „image“) angegeben werden. Die Liste erlaubter Werte wird durch die IANA definiert.
	Aspect	**1:1**	*+MapAspect*	Tatsächlicher geografischer Ausschnitt der generierten Karte. Er darf von dem angefragten Kartenausschnitt abweichen. Allerdings muss der Mittelpunkt nahezu unverändert bleiben und der tatsächliche Kartenausschnitt soll dem angefragten so ähnlich wie möglich sein. Vgl. 17.2.4.

Tabelle 188: Beschreibung der Struktur ***MapResultStructure***.

17.3.5. ImagePointResultStructure

ImagePointResultStructure			*+Structure*	**Fasst die Daten einer Umrechnung von Geo- zu Bildkoordinaten zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Umrechnung des nachfolgenden Geokoordinaten-Paars. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	Point	**1:1**	*+GeoPosition*	Geografischer Punkt, für den Bildkoordinaten berechnet werden sollten. Dies muss einer der Punkte aus der zugehörigen Dienstanfrage sein. Vgl. 7.2.3.
	ImagePoint	0:1	*+MapCoordinate*	Bildkoordinaten für den angefragten geografischen Punkt, falls die Umrechnung durchgeführt werden konnte. Der Bildpunkt kann außerhalb des zugrundeliegenden Kartenbildes liegen (inklusive negativer Koordinatenwerte). Falls dieses Element fehlt, muss mindestens ein Fehlercode gesetzt sein. Vgl. 17.2.6.

Tabelle 189: Beschreibung der Struktur ***ImagePointResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

MAP_POINTNOTONMAP	Der angegebene Punkt liegt außerhalb der Karte. Trotzdem konnte eine Konversion durchgeführt werden.
MAP_UNSUPPORTEDPOINT	Der angefragte Punkt liegt außerhalb des umrechenbaren Gebietes.

Tabelle 190: Liste der Fehlerzustände in ***ImagePointResultStructure***.

17.3.6. GeoCoordinateResultStructure

GeoCoordinateResultStructure			*+Structure*	**Fasst die Daten einer Umrechnung von Bild- zu Geokoordinaten zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Umrechnung des nachfolgenden Geokoordinaten-Paars. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	Point	0:1	*+GeoPosition*	Geografische Koordinaten für den angefragten Bildpunkt, falls die Umrechnung durchgeführt werden konnte, berechnet werden sollten. Falls dieses Element fehlt, muss mindestens ein Fehlercode gesetzt sein. Vgl. 7.2.3.
	ImagePoint	**1:1**	*+MapCoordinate*	Bildkoordinaten, für den geografische Koordinaten berechnet werden sollten. Dies muss einer der Punkte aus der zugehörigen Dienstanfrage sein. Vgl. 17.2.6.

Tabelle 191: Beschreibung der Struktur ***GeoCoordinateResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

MAP_POINTNOTONMAP	Der angegebene Punkt liegt außerhalb der Karte. Trotzdem konnte eine Konversion durchgeführt werden.
MAP_UNSUPPORTEDPOINT	Der angefragte Punkt liegt außerhalb des umrechenbaren Gebietes.

Tabelle 192: Liste der Fehlerzustände in ***GeoCoordinateResultStructure***.

18. Dienst Schadensmeldung / Zustand von Einrichtungen

18.1. Beschreibung

Dieser Dienst erlaubt es, den Zustand einer Haltestelleneinrichtung oder Fahrzeugausstattung abzufragen oder zu melden (Schadensmeldung).

18.2. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

FacilityStatusTypeEnumeration	*OK* | *dirty* | *destroyed* | *damaged* | *stolen* | *out of order*	Zustand der Einrichtung.
FacilityAvailabilityEnumeration	*unknown* | *available* | *notAvailable* | *partiallyAvailable* | *added* | *removed*	Verfügbarkeit der Einrichtung.

Tabelle 193: Beschreibung der Struktur einfacher Typen

18.3. Komplexe Strukturen

18.3.1. FacilityStructure

FacilityStructure				*+Structure*	**Beschreibung und Zustand einer Einrichtung.**
	a	***VehicleFacility***	**-1:1**	*+VehicleFacility*	Definition einer Fahrzeugeinrichtung. Vgl. 18.3.2.
	b	***InfrastructureFacility***		*+InfrastructureFacility*	Definition einer Infrastruktureinrichtung. Vgl. 18.3.3.
	Condition		**1:1**	*+FacilityStatus*	Zustand der Einrichtung. Vgl. 18.3.7.
	Extension		0:1	*xs:anyType*	Erweiterung.

Tabelle 194: Beschreibung der Struktur ***FacilityStructure***.

18.3.2. VehicleFacilityStructure

VehicleFacilityStructure			*+Structure*	**Beschreibung einer Fahrzeugeinrichtung.**
VehicleFacilityRef	***:::***	**1:1**	*+VehicleFacilityRefGroup*	Referenz auf eine Einrichtung. Vgl. 18.3.4.
ServiceFacility	***:::***	**1:1**	*+ServiceFacilityGroup*	Klassifizierung der Einrichtung. Vgl. 7.7.3.
	FacilityDescription	0:1	*+InternationalText*	Name oder Beschreibung der Einrichtung. Vgl. 7.2.2.
	LocationDescription	0:1	*+InternationalText*	Beschreibung, wo die Einrichtung zu finden ist. Vgl. 7.2.2.

Tabelle 195: Beschreibung der Struktur ***VehicleFacilityStructure***.

18.3.3. InfrastructureFacilityStructure

InfrastructureFacilityStructure			*+Structure*	**Beschreibung einer Infrastruktureinrichtung.**
Infra-structureFa-cilityRef	***:::***	**1:1**	*+Infrastructure FacilityRef-Group*	Referenz auf eine Einrichtung. Vgl. 18.3.5.
StopFa-cility	***:::***	**1:1**	*+StopFacility Group*	Klassifizierung der Einrichtung. Vgl. 7.7.2.
	FacilityDescription	0:1	*+International Text*	Name oder Beschreibung der Einrichtung. Vgl. 7.2.2.
	Location	0:1	*+GeoPosition*	Koordinatenposition der Einrichtung. Vgl. 7.2.3.
	LocationDescription	0:1	*+International Text*	Beschreibung, wo die Einrichtung zu finden ist. Vgl. 7.2.2.

Tabelle 196: Beschreibung der Struktur ***InfrastructureFacilityStructure***.

18.3.4. VehicleFacilityRefGroup

VehicleFacilityRefGroup			*+Group*	**Referenzierung einer Fahrzeugeinrichtung durch Referenz auf die Einrichtung selbst oder auf ein übergeordnetes Objekt.**
	FacilityRef	0:1	*→FacilityCode*	Referenz auf Einrichtung. Vgl. 7.4.1.
	OwnerRef		*→OwnerCode*	Referenz auf Eigentümer/Verantwortlichen. Vgl. 7.4.1.
	OperatorRef		*→Operator-Code*	Referenz auf Verkehrsunternehmen. Vgl. 7.4.1.
	LineRef		*→LineCode*	Referenz auf Linie. Vgl. 7.4.1.
	JourneyRef		*→Jour-neyCode*	Referenz auf Fahrt. Vgl. 7.4.1.
	VehicleRef		*→Vehicle-Code*	Referenz auf Fahrzeug. Vgl. 7.4.1.

Tabelle 197: Beschreibung der Gruppe ***VehicleFacilityRefGroup***.

18.3.5. InfrastructureFacilityRefGroup

InfrastructureFacilityRefGroup			*+Group*	**Referenzierung einer Infrastruktureinrichtung durch Referenz auf die Einrichtung selbst oder auf ein übergeordnetes Objekt.**
	FacilityRef	0:1	*→FacilityCode*	Referenz auf Einrichtung. Vgl. 7.4.1.
	OwnerRef		*→OwnerCode*	Referenz auf Eigentümer/Verantwortlichen. Vgl. 7.4.1.
	StopPointRef		*→StopPoint*	Referenz auf einen Haltepunkt. Vgl. 7.5.1.
	StopPlaceRef		*→StopPlace*	Referenz auf eine Haltestelle. Vgl. 7.5.1.
	OperatorRef		*→Operator-Code*	Referenz auf Verkehrsunternehmen. Vgl. 7.4.1.
	LineRef		*→LineCode*	Referenz auf Linie. Vgl. 7.4.1.

Tabelle 198: Beschreibung der Gruppe ***InfrastructureFacilityRefGroup***.

18.3.6. FacilityDataFilterGroup

FacilityDataFilterGroup			*+Group*	**Eine Menge von Objektreferenzen als Filter zum Eingrenzen der in Frage kommenden Einrichtungen.**
	FacilityRef	0:*	*→FacilityCode*	Referenz auf Einrichtung. Vgl. 7.4.1.
	OwnerRef	0:*	*→OwnerCode*	Referenz auf Eigentümer/Verantwortlichen. Vgl. 7.4.1.
	StopPointRef	0:*	*→StopPoint*	Referenz auf einen Haltepunkt. Vgl. 7.5.1.
	StopPlaceRef	0:*	*→StopPlace*	Referenz auf eine Haltestelle. Vgl. 7.5.1.
	OperatorRef	0:*	*→Operator-Code*	Referenz auf Verkehrsunternehmen. Vgl. 7.4.1.
	LineRef	0:*	*→LineCode*	Referenz auf Linie. Vgl. 7.4.1.
	JourneyRef	0:*	*→Jour-neyCode*	Referenz auf Fahrt. Vgl. 7.4.1.
	VehicleRef	0:*	*→Vehicle-Code*	Referenz auf Fahrzeug. Vgl. 7.4.1.

Tabelle 199: Beschreibung der Gruppe ***FacilityDataFilterGroup***.

18.3.7. FacilityStatusStructure

FacilityStatusStructure			*+Structure*	**Der Zustand einer Einrichtung.**
	Availability	**1:1**	*FacilityAvaila-bilityEnumera-tion*	Verfügbarkeit der Einrichtung. Vgl. 18.2
	Status	**1:1**	*FacilitySta-tusTypeEnu-meration*	Klassifizierung des Zustands. Vgl. 18.2
	StatusDescription	0:1	*+International Text*	Beschreibung des Zustands. Vgl. 7.2.2.

Tabelle 200: Beschreibung der Struktur ***FacilityStatusStructure***.

18.4. Anfrage Schadensmeldung

Eine Schadensmeldung an einer Einrichtung wird mittels eines Elements ***FacilityStatusReport*** vom Typ *FacilityStatusReportStructure* gestellt.

18.4.1. FacilityStatusReportStructure

FacilityStatusReportStructure				*+Structure*	**Fasst die Daten für eine Schadensmeldung an einer Einrichtung zusammen.**
	a	***VehicleFacility***	**-1:1**	*+VehicleFacilit y*	Definition einer Fahrzeugeinrichtung. Vgl. 18.3.2.
	b	***InfrastructureFacili-ty***		*+Infrastructure Facility*	Definition einer Infrastruktureinrichtung. Vgl. 18.3.3.
	Condition		**1:1**	*+FacilityStatu s*	Zustand der Einrichtung. Vgl. 18.3.7.
	Extension		0:1	*xs:anyType*	Erweiterung.

Tabelle 201: Beschreibung der Struktur ***FacilityStatusReportStructure***.

18.5. Antwort Schadensmeldung

Die Antwort auf eine Schadensmeldung (FacilityStatusReport, siehe 18.4) wird mittels eines Elements ***FacilityStatusReportResponse*** vom Typ *FacilityStatusReportResponseStructure* übertragen.

18.5.1. FacilityStatusReportResponseStructure

FacilityStatusReportResponseStructure			*+Structure*	**Antwort auf die Schadensmeldung.**
	ErrorMessage	**1:1**	*+ErrorMessage*	Fehlermeldung. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.

Tabelle 202: Beschreibung der Struktur ***FacilityStatusReportResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

FACILITYSTA-TUSREPORT_FACILITYUNKNOWN	Die angegebene Einrichtung ist unbekannt.
FACILITYSTA-TUSREPORT_OWNERUNKNOWN	Der angegebene Eigentümer ist unbekannt.
FACILITYSTA-TUSREPORT_OPERATORUNKNOWN	Das angegebene Verkehrsunternehmen ist unbekannt.
FACILITYSTA-TUSREPORT_LINEUNKNOWN	Die angegebene Linie ist unbekannt.
FACILITYSTA-TUSREPORT_JOURNEYUNKNOWN	Die angegebene Fahrt ist unbekannt.
FACILITYSTA-TUSREPORT_VEHICLEUNKNOWN	Das angegebene Fahrzeug ist unbekannt.
FACILITYSTA-TUSREPORT_STOPPOINTUNKNOWN	Der angegebene Haltepunkt ist unbekannt.
FACILITYSTA-TUSREPORT_STOPPLACEUNKNOWN	Die angegebene Haltestelle ist unbekannt.

Tabelle 203: Liste der Fehlerzustände in ***FacilityStatusReportResponse***.

18.6. Anfrage Zustand von Einrichtungen

Eine Anfrage nach dem aktuellen Zustand von Einrichtungen wird mittels eines Elements ***FacilityRequest*** vom Typ *FacilityRequestStructure* gestellt.

18.6.1. FacilityRequestStructure

FacilityRequestStructure			*+Structure*	**Fasst die Daten für eine Schadensmeldung an einer Einrichtung zusammen.**
FacilityData-Filter	***:::***	**1:1**	*+FacilityDataFilterGroup*	Objektreferenzen als Filter. Vgl. 18.3.6.
	Extension	0:1	*xs:anyType*	Erweiterung.

Tabelle 204: Beschreibung der Struktur ***FacilityRequestStructure***.

18.7. Antwort Zustand von Einrichtungen

Die Antwort auf eine Zustandsanfrage wird mittels eines Elements ***FacilityResponse*** vom Typ *FacilityResponseStructure* übertragen.

18.7.1. FacilityResponseStructure

FacilityResponseStructure			*+Structure*	**Antwort auf die Schadensmeldung.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldung bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	FacilityResult	0:*	*+FacilityResult*	Resultatstruktur. Vgl. 18.7.2.

Tabelle 205: Beschreibung der Struktur ***FacilityResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

FACILITYREQUEST_FACILITYUNKNOWN	Die angegebene Einrichtung ist unbekannt.
FACILITYREQUEST_OWNERUNKNOWN	Der angegebene Eigentümer ist unbekannt.
FACILITYREQUEST_OPERATORUNKNOWN	Das angegebene Verkehrsunternehmen ist unbekannt.
FACILITYREQUEST_LINEUNKNOWN	Die angegebene Linie ist unbekannt.
FACILITYREQUEST_JOURNEYUNKNOWN	Die angegebene Fahrt ist unbekannt.
FACILITYREQUEST_VEHICLEUNKNOWN	Das angegebene Fahrzeug ist unbekannt.
FACILITYREQUEST_STOPPOINTUNKNOWN	Der angegebene Haltepunkt ist unbekannt.
FACILITYREQUEST_STOPPLACEUNKNOWN	Die angegebene Haltestelle ist unbekannt.

Tabelle 206: Liste der Fehlerzustände in ***FacilityResponse***.

18.7.2. FacilityResultStructure

FacilityResultStructure			*+Structure*	**Resultatstruktur für die Facility-Zustandsanfrage.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf das einzelne FacilityResult. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	Facility	**1:1**	*+Facility*	Informationen zur Einrichtung. Vgl. 18.3.1

Tabelle 207: Beschreibung der Struktur ***FacilityResultStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

FACILITYREQUEST_STATUSNOTCONFIRMED	Für die Einrichtung liegt ein Zustandsbericht vor, der aber noch nicht bestätigt ist.

Tabelle 208: Liste der Fehlerzustände in ***FacilityResultStructure***.

19. Dienst Benachrichtigungsdienst

19.1. Beschreibung

Der Benachrichtigungsdienst dient der aktiven Benachrichtigung von Benutzern über aktuelle Geschehnisse. Die Benutzer können Abonnements einrichten, um sich bei Auftreten neuer Informationen benachrichtigen zu lassen

Der Dienst informiert unter anderem über

- Störungen oder Ereignisse auf einer Verbindung, einer Strecke oder an einer Haltestelle,
- alternative Verbindungen (aufgrund von Störungen oder der Prozessdatenlage),
- den Status eines Anschlusses und zusätzliche Informationen bei Anschlussverlust.

Diese Aufzählung ist nicht abschließend.

Dabei verfügt der Benachrichtigungsdienst über Funktionalität, die es ihm ermöglicht, betroffene Objekte (Verbindungen, Anschlüsse etc.) zu ermitteln. Teile des Benachrichtigungsdienstes (Ereignismeldungen und Änderungen an der Fahrzeugausstattung bzw. an Haltestelleneinrichtungen) wurden aus dem SIRI-Standard übernommen.

Eine Benachrichtigung besteht aus einer eindeutigen ID, einem Typ und den Nutzdaten. Folgende Arten von Abonnements können eingerichtet werden (vgl. auch die allgemeine Beschreibung von Abonnementanfragen in 7.1.2):

- SituationExchangeSubscriptionRequest (aus SIRI SX):
  Benachrichtigung über Ereignisse und Störungen,
- FacilityMonitoringSubscriptionRequest (aus SIRI FM):
  Änderungen an der Fahrzeugausstattung bzw. an Haltestelleneinrichtungen,
- TripMonitoringSubscriptionRequest (neu in TRIAS):
  Überwachung einer geplanten Verbindung.

Die Funktionsweise des Benachrichtigungsdienstes ist in Abbildung 9 zu sehen. Um Nachrichten zu erhalten, muss ein Portalsystem ein Abonnement beim Benachrichtigungsdienst erstellen (1). Bei der Erstellung des Abonnements (SubscriptionRequest) kann angegeben werden, welche Typen von Nachrichten dem Portalsystem übermittelt werden sollen. Die Erstellung des Abonnements wird vom Benachrichtigungsdienst synchron mit einer siri:SubscriptionResponse beantwortet.

Bei jeder neuen Meldung von einem Datenlieferant (2) ermittelt der Benachrichtigungsdienst die betroffenen Abonnements und leitet die Nachrichten an das zugehörige Portalsystem weiter. (3) Eine Nachricht vom Benachrichtigungsdienst, die eine Datenlieferung (ServiceDelivery) im Rahmen eines bestehenden Abonnements überliefert (z.B. als TripMonitoringDelivery), wird vom Portalsystem synchron mit einer DataReceivedAcknowledgement-Nachricht bestätigt.

Soll das Portalsystem keine neuen Nachrichten mehr empfangen, so muss es das Abonnement am Benachrichtigungsdienst abmelden (4) (oder das Abonnement erlischt von selbst nach Ablauf des Gültigkeitszeitraums).

Eine Aktualisierung des Abonnements durch das Portalsystem (z.B. wegen geänderter TripMonitoringParam) ist nicht vorgesehen, stattdessen muss das Abonnement abgemeldet (4) und neu eingerichtet (1) werden.

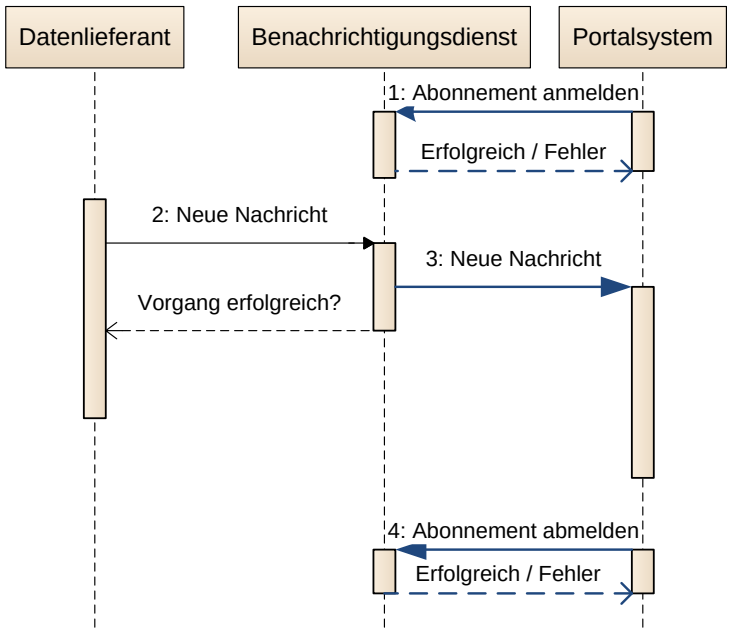

Abbildung 9: Funktionsweise des Benachrichtigungsdienstes

19.2. Komplexe Strukturen

Im Folgenden werden die komplexen Strukturen beschrieben, die in Trias_Alerts definiert sind.

19.2.1. TripMonitoringParamStructure

TripMonitoringParamStructure			*+Structure*	**Parameter für die Verbindungsüberwachung.**
TripMonitoringPolicy	*Severity*	0:1	*unknown* | *verySlight* | *slight* | *normal* | *severe* | *verySevere* | *noImpact* | *undefined*	Prioritäten von Ereignissen (nach TPEG-Tabelle 26).
	MinimumDelayChangeThreshold	0:1	*xs:duration*	Verspätungsänderung, ab der wieder benachrichtigt wird.
	AcceptThirdPartyInformation	0:1	*xs:boolean*	Gibt an, ob Informationen von anderen Plattformen abgeholt werden sollen. Default ist *false*.
	IncludeAlternatives	0:1	*xs:boolean*	Gibt an, ob direkt Alternativen mit zurückgegeben werden sollen. Default ist *false*.

Tabelle 209: Beschreibung der Struktur ***TripMonitoringParamStructure***

Der Parameter *AcceptThirdPartyInformation* gibt an, ob die angefragte EKAP die Antwort rein aus eigenen Informationsquellen erstellen soll, oder ob sie (fehlende) Informationen aus weiteren Datenquellen, wie beispielsweise weitere EKAPs, einbeziehen darf. Somit lässt sich durch die anfragende Plattform steuern, woher die Daten stammen. Der Anfragende kann damit entweder selbst fehlende Informationen an verschiedenen Plattformen anfragen oder diese Funktion von dem angefragten System ausführen lassen. Der Parameter ist insbeson-

dere dann sinnvoll, wenn mehrere EKAPs zusammengeschaltet werden und dabei unterschiedliche (geografische) Zuständigkeiten haben.

19.3. Anfragestrukturen

19.3.1. TripMonitoringSubscriptionRequestStructure

TripMonitoringSubscriptionRequestStructure			*+Structure (abgeleitet von siri:AbstractSubscription-Structure)*	**Einrichten einer Verbindungsüberwachung.**
	Trip	**1:1**	*+Trip*	Zu überwachende Verbindung. Vgl. 9.3.4.
	TripRequest	0:1	*+TripRequest*	Ursprüngliche Verbindungsanfrage. Vgl. 9.2.1.
	MonitoringParameter	0:1	*+TripMonitoringParam*	Weitere Parameter zur Konfiguration der Verbindungsüberwachung. Vgl. 19.2.1.

Tabelle 210: Beschreibung der Struktur ***TripMonitoringSubscriptionRequestStructure***

19.4. Antwortstrukturen

19.4.1. TripMonitoringDeliveryStructure

TripMonitoringDeliveryStructure			*+Structure (abgeleitet von siri:AbtractServiceDeliveryStructure)*	**Liefert Informationen zu einer überwachten Verbindung.**
MonitoringAlertReason	*Situations*	0:1	*Situations*	(Störungs-)Ereignisse als Grund für die Meldung (vgl. 7.8.1).
	FacilityCondition	0:*	*+siri:FacilityCondition*	Ein oder mehrere Zustände von Facilities als Grund für die Meldung, vgl. 7.7.
	ConnectionStatus	0:*	*+ConnectionStatus*	Der Status eines Anschlusses in der überwachten Verbindung. Vgl. 13.3.4.
AlternativeTrip	*TripResponse*	0:1	*+TripResponse*	Enthält eine Verbindungsalternative (vgl. 9.3.1).
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Vgl. 7.4.2.

Tabelle 211: Beschreibung der Struktur ***TripMonitoringDeliveryStructure***

In *ErrorMessage* können folgende Fehlerzustände auftreten:

ALERT_TRIPREQUEST_ORIGIN_UNKNOWN	Der Abfahrtsort (Adresse, Haltestelle, …), des TripRequests ist unbekannt.
ALERT_TRIPREQUEST_DESTINATION_UNKNOWN	Der Ankunftsort (Adresse, Haltestelle, …), des TripRequests ist unbekannt.
ALERT_TRIP_UNKNOWN	Die zu überwachende Verbindung ist unbekannt.
ALERT _THRESHOLD_NEGATIVE	Verspätungsänderung, ab der wieder benachrichtigt wird, hat einen negativen Wert.
ALERT_FACILITY_UNKNOWN	Die zu überwachende Einrichtung ist unbekannt.
ALERT_MONITORED_OBJECT_UNKNOWN	Das Objekt, zu dem Ereignisse und Störungen gemeldet werden sollen, ist unbekannt.

Tabelle 212: Liste der Fehlerzustände in ***TripMonitoringDeliveryStructure***.

20. Dienst Personalisierungsdienst

20.1. Beschreibung

Dieser Dienst stellt Funktionen bereit, über die Daten für die personalisierte Konfiguration beliebiger Dienste hinterlegt werden können. Unter „Konfiguration“ sind hier nicht nur explizite Einstellungen im engeren Sinn zu verstehen, sondern allgemein benutzerbezogene Daten. All diese Informationen können das Verhalten derjenigen Dienste, die den Personalisierungsdienst benutzen, beeinflussen und stellen somit eine Konfiguration für den verwendenden Dienst dar.

Es ist wichtig, zu beachten, dass der Dienst keine eigenen personalisierten Funktionen zur Verfügung stellt. Seine Aufgabe besteht in der Verwaltung von Benutzereinstellungen. Andere Dienste können auf den Personalisierungsdienst zurückgreifen, um ihre Funktionen personalisiert zur Verfügung zu stellen. Der Zugriff auf die personalisierten Daten erfolgt mit Hilfe des Authentifizierungsdienstes, um einen Schutz der Daten, entsprechend der rechtlichen Vorgaben zum Datenschutz, zu ermöglichen. Diese Vorgaben sind bei der Umsetzung des Personalisierungsdienstes zu berücksichtigen, werden in der Dienstbeschreibung des Personalisierungsdienstes jedoch nicht behandelt. Generell gilt, dass Benutzer nur auf von ihnen gespeicherte Daten zugreifen können. Daten von anderen Benutzern bleiben stets vollkommen unsichtbar und unerreichbar. Wie diese Trennung der benutzerbezogenen Daten erreicht wird, ist herstellerabhängig und wird hier nicht vorgeschrieben.

Der Dienst speichert beliebige Datenwerte, wobei jedem Datenwert ein – für den aktuellen Benutzer – eindeutiger Schlüssel zugeordnet wird. Über diesen Schlüssel kann der Datenwert wieder abgerufen werden. Bei den Werten handelt es sich um Zeichenketten mit beliebigem Format, sodass prinzipiell jegliche Datenstrukturen abgelegt werden können.

Der Dienst bietet keine Zuordnung von Werten zu einem oder mehreren anderen Diensten an. Diese kann über herstellerspezifische Schnittstellen unterstützt werden.

20.2. Interaktionen

Die Funktionen dieses Dienstes stehen oft für sich alleine und sind sehr generisch. Daher werden im Folgenden drei beispielhafte Abläufe im Zusammenhang mit dem Dienst beschrieben, bei denen die Funktionsaufrufe in einen größeren Zusammenhang eingebettet sind.

Auf den Personalisierungsdienst kann von verschiedenen Komponenten aus zugegriffen werden, zum Beispiel von der EKAP oder von Mehrwertdiensten aus. Bei den im Folgenden gezeigten Zugriffen auf den Personalisierungsdienst handelt es sich um beispielhafte Abläufe. Dabei sollen die anderen beteiligten Komponenten nicht konkret festgelegt werden. Aus diesem Grund wird in den untenstehenden Diagrammen von beliebigen Mehrwertdiensten - welche in beliebigen Ausprägungen vorkommen können und im Rahmen dieser Schrift ebenfalls nicht näher spezifiziert werden - gesprochen. Zugriffe von anderen Komponenten aus finden nach demselben Muster statt.

20.2.1. Lebenszyklus eines Wertes

Das erste Beispiel zeigt auf, wie ein Wert über den Personalisierungsdienst abgelegt und wieder gelöscht werden kann.

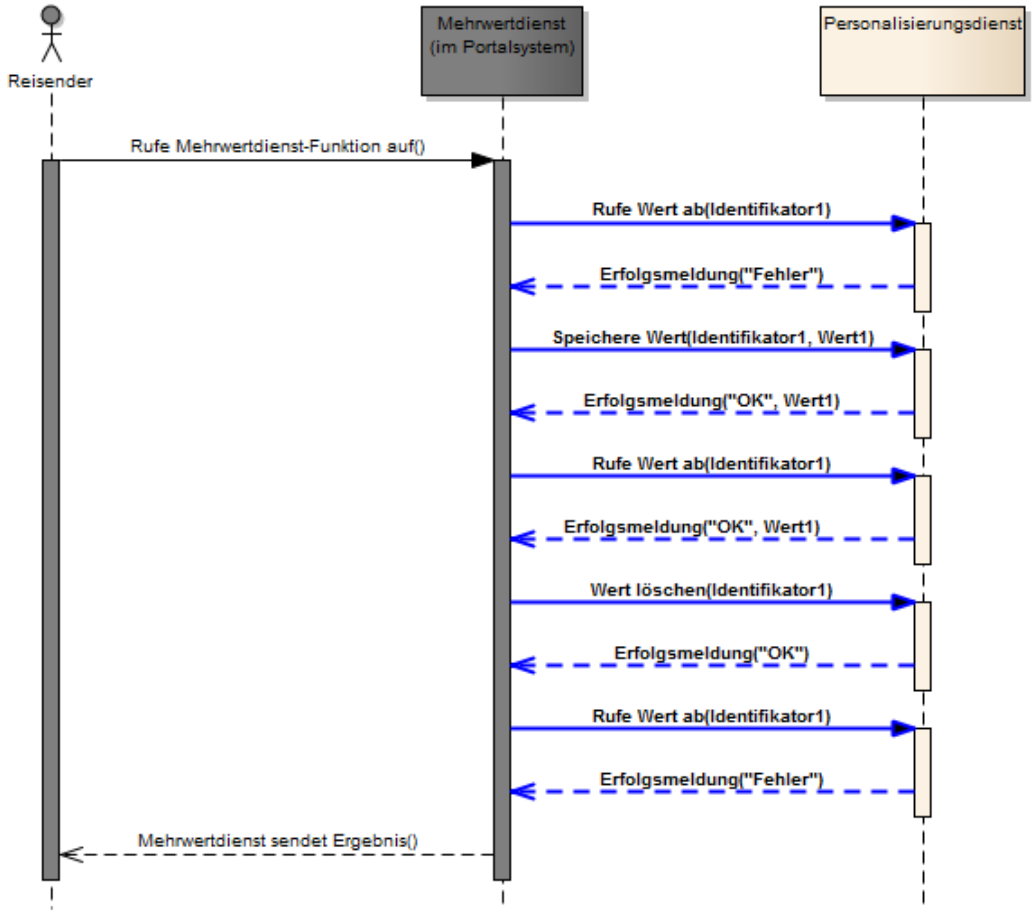

Abbildung 10: Sequenzdiagramm Lebenszyklus eines Wertes

20.2.2. Werteliste ermitteln

In diesem Beispiel ist dargestellt, wie die Funktion zum Auflisten der verfügbaren Werte funktioniert.

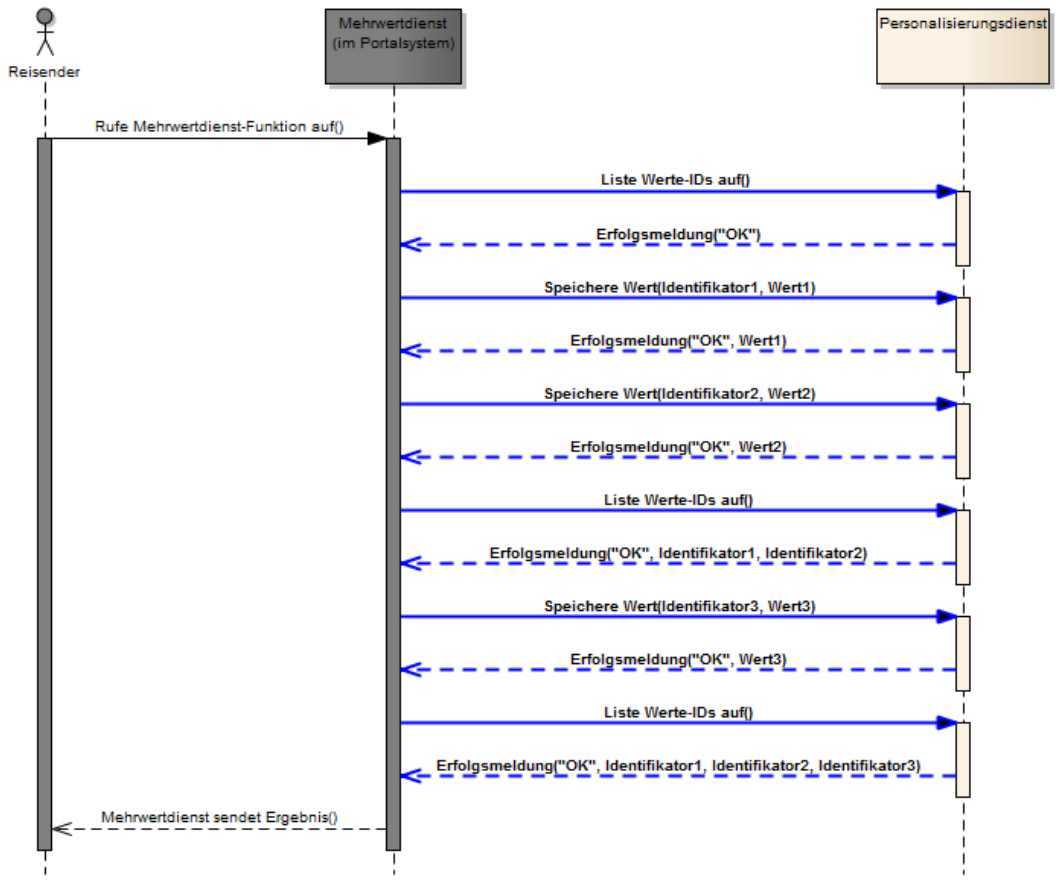

Abbildung 11: Sequenzdiagramm Werteliste ermitteln

20.2.3. Werte speichern und abrufen

Im folgenden Beispiel wird gezeigt, wie ein Reisender auf unterschiedliche Mehrwertdienste zugreift, welche den Personalisierungsdienst nutzen, um Konfigurationseinstellungen abzulegen und wieder abzurufen.

Dabei wird illustriert, dass zu jedem gespeicherten Wert ein für den Benutzer eindeutiger Identifikator gehört. Erstmaliges Speichern eines Wertes für einen Identifikator legt einen Wert im Speicher an, erneutes Speichern unter demselben Identifikator überschreibt den ursprünglichen Wert.

Des Weiteren wird im Beispiel gezeigt, dass die Werte vom Personalisierungsdienst prinzipiell dauerhaft gespeichert werden, auch wenn beispielsweise inzwischen ein anderer Mehrwertdienst genutzt wird. Hierbei ist zu beachten, dass Anbieter gespeicherte Werte mit einem Löschdatum versehen können, um ungenutzte Daten nicht unbegrenzt vorhalten zu müssen. Die genaue Umsetzung entsprechender Löschungen ist anbieterspezifisch und wird in dieser Schrift nicht festgelegt.

Um klarzustellen, dass in der standardisierten Form keine Zuordnung zwischen Werten und Diensten stattfindet, wird ferner vorgeführt, dass ein Mehrwertdienst (MWD 2 in der Abbildung) einen ursprünglich von einem anderen Mehrwertdienst (MWD 1 in der Abbildung) abgelegten Wert überschreiben kann. Die Voraussetzung dazu ist lediglich, dass MWD 2 den Identifikator des Werts kennt, sei es, weil MWD 1 und 2 vom selben Hersteller stammen, oder weil der Entwickler von MWD 1 die in MWD 1 verwendeten Identifikatoren öffentlich bekanntgegeben hat.

Der Dienst kann bei Bedarf durch herstellerspezifische Schnittstellen um die Funktionalität, den Zugriff auf bestimmte Werte dienstspezifisch einzuschränken, erweitert werden und somit den Zugriff ausgewählter Dienste auf einzelne Werte einschränken. Hierzu kann, je nach Implementierung und Konfiguration, auch der Authentifizierungsdienst genutzt werden.

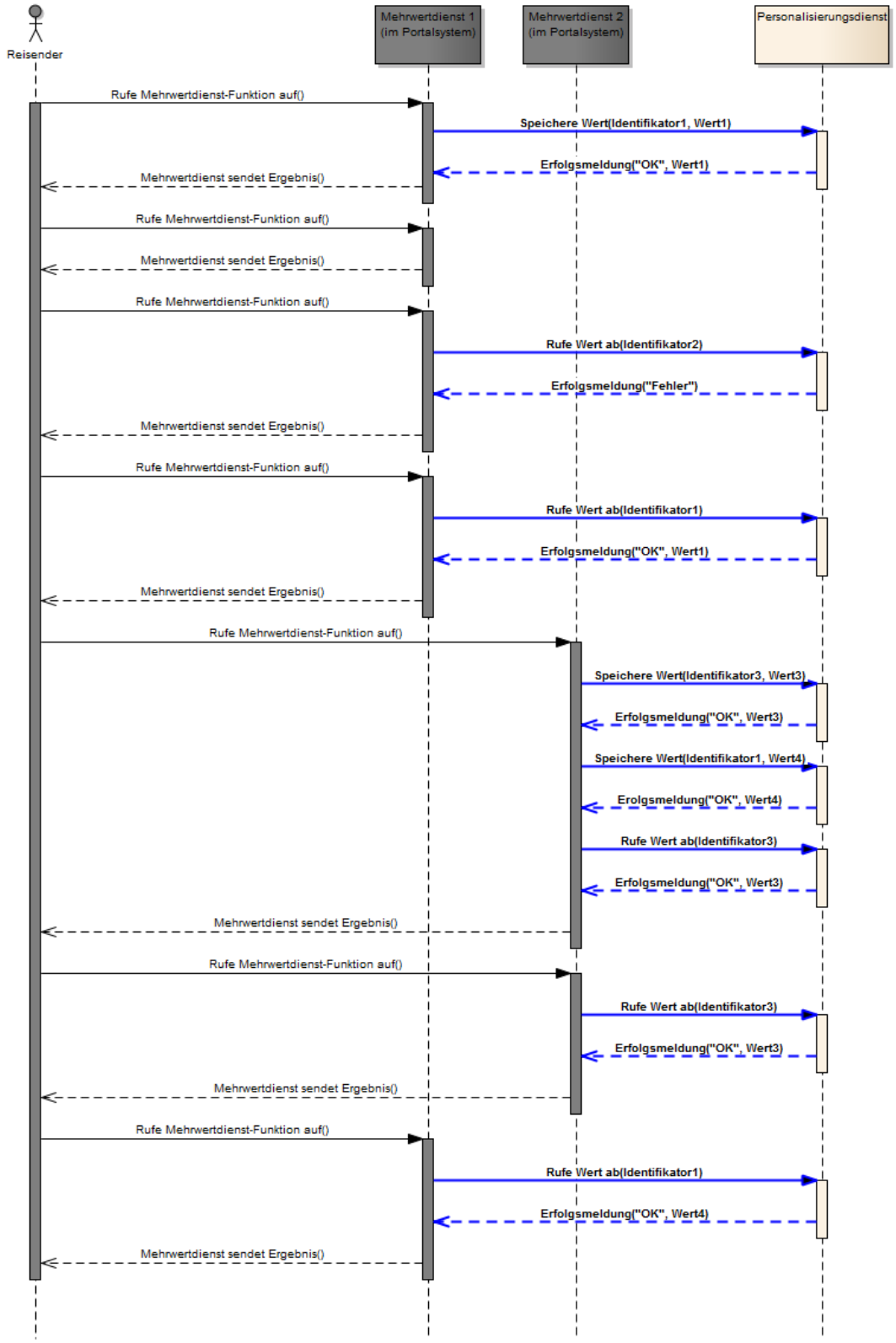

Abbildung 12: Sequenzdiagramm Personalisierung

20.3. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

ValueIdType	*xs:string*	Identifikator eines Wertes.

Tabelle 213: Liste der einfachen Typdefinitionen in Trias_Personalisation.xsd.

20.4. Anfragestrukturen

20.4.1. PersonalisationRequestStructure

PersonalisationRequest				*+Structure*	**Repräsentiert eine Anfrage an den Personalisierungsdienst.**
	a	***SaveValue***	**–1:1**	*+PersonalisationSaveValueRequest*	Falls dies vorhanden ist, soll durch die Anfrage ein personalisierter Wert gespeichert werden; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.4.2).
	b	***RetrieveValue***		*+PersonalisationRetrieveValueRequest*	Falls dies vorhanden ist, soll durch die Anfrage ein personalisierter Wert abgerufen werden; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.4.3).
	c	***DeleteValue***		*+PersonalisationDeleteValueRequest*	Falls dies vorhanden ist, soll durch die Anfrage ein personalisierter Wert gelöscht werden; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.4.4).
	d	***EnumerateValue***		*+PersonalisationEnumerateValuesRequest*	Falls dies vorhanden ist, sollen durch die Anfrage verfügbare personalisierte Werte abgerufen werden; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.4.5).

Tabelle 214: Beschreibung der Struktur ***PersonalisationRequestStructure***.

20.4.2. PersonalisationSaveValueRequestStructure

PersonalisationSaveValueRequest			*+Structure*	**Enthält genauere Angaben zum Speichern eines personalisierten Wertes.**
	ValueId	**1:1**	*→ValueId*	Der Identifikator des zu speichernden Wertes. Vgl. 20.3.
	Value	**1:1**	*xs:string*	Der zu speichernde Wert.

Tabelle 215: Beschreibung der Struktur ***PersonalisationSaveValueRequestStructure***.

20.4.3. PersonalisationRetrieveValueRequestStructure

PersonalisationRetrieveValueRequest			*+Structure*	**Enthält genauere Angaben zum Abrufen eines personalisierten Wertes.**
	ValueId	**1:1**	*→ValueId*	Der Identifikator des abzurufenden Wertes. Vgl. 20.3.

Tabelle 216: Beschreibung der Struktur ***PersonalisationRetrieveValueRequestStructure***.

20.4.4. PersonalisationDeleteValueRequestStructure

PersonalisationDeleteValueRequest			*+Structure*	**Enthält genauere Angaben zum Löschen eines personalisierten Wertes.**
	ValueId	**1:1**	*→ValueId*	Der Identifikator des zu löschenden Wertes. Vgl. 20.3.

Tabelle 217: Beschreibung der Struktur ***PersonalisationDeleteValueRequestStructure***.

20.4.5. PersonalisationEnumerateValuesRequestStructure

PersonalisationEnumerateValuesRequest	*+Structure*	**Drückt aus, dass eine Liste der gespeicherten Werte abgerufen werden soll.**

Tabelle 218: Beschreibung der Struktur ***PersonalisationEnumerateValuesRequestStructure***.

20.5. Antwortstrukturen

20.5.1. PersonalisationResponseStructure

PersonalisationResponse				*+Structure*	**Repräsentiert die Antwort auf eine Anfrage an den Personalisierungsdienst.**
	ErrorMessage		0:*	*+ErrorMessage*	Enthält etwaige Fehlermeldungen, die die generelle Verarbeitung der Nachricht durch einen Personalisierungsdienst betreffen.
	a	***SaveValue***	**–1:1**	*+PersonalisationSaveValueResponse*	Falls dies vorhanden ist, drückt die Antwort aus, ob das Speichern eines personalisierten Wertes erfolgreich war; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.5.2).
	b	***RetrieveValue***		*+PersonalisationRetrieveValueResponse*	Falls dies vorhanden ist, drückt die Antwort aus, ob das Abrufen eines personalisierten Wertes erfolgreich war; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.5.3).
	c	***DeleteValue***		*+PersonalisationDeleteValueResponse*	Falls dies vorhanden ist, drückt die Antwort aus, ob das Löschen eines personalisierten Wertes erfolgreich war; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.5.4).
	d	***EnumerateValue***		*+PersonalisationEnumerateValuesResponse*	Falls dies vorhanden ist, drückt die Antwort aus, ob das Auflisten aller vorhandenen personalisierten Werte erfolgreich war; genauere Angaben dazu sind in dieser Struktur enthalten (vgl. 20.5.5).

Tabelle 219: Beschreibung der Struktur ***PersonalisationResponseStructure***.

20.5.2. PersonalisationSaveValueResponseStructure

PersonalisationSaveValueResponse			*+Structure*	**Enthält genauere Angaben zum abgeschlossenen Speichern eines personalisierten Wertes.**
	ErrorMessage	0:*	*+ErrorMessage*	Gibt an, ob der Speichervorgang erfolgreich war.
	Value	0:1	*xs:string*	Der soeben gespeicherte Wert.

Tabelle 220: Beschreibung der Struktur ***PersonalisationSaveValueResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

PERSONALISATIONSAVEVALU-EREQUEST_INVALID_ID	Der angegebene Identifikator hat ein ungültiges Format.

Tabelle 221: Liste der Fehlerzustände in ***PersonalisationSaveValueResponseStructure***.

20.5.3. PersonalisationRetrieveValueResponseStructure

PersonalisationRetrieveValueResponse			*+Structure*	**Enthält genauere Angaben zum abgeschlossenen Abrufen eines personalisierten Wertes.**
	ErrorMessage	0:*	*+ErrorMessage*	Gibt an, ob der Abruf erfolgreich war.
	Value	0:1	*xs:string*	Falls der Abruf erfolgreich war, der abgerufene Wert.

Tabelle 222: Beschreibung der Struktur ***PersonalisationRetrieveValueResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

PERSONALISATIONRETRIEVEVALU-EREQUEST_UNKNOWN_ID	Zum angegebenen Identifikator ist im Personalisierungsdienst kein Wert abgelegt.

Tabelle 223: Liste der Fehlerzustände in ***PersonalisationRetrieveValueResponseStructure***.

20.5.4. PersonalisationDeleteValueResponseStructure

PersonalisationDeleteValueResponse			*+Structure*	**Enthält genauere Angaben zum abgeschlossenen Löschen eines personalisierten Wertes.**
	ErrorMessage	0:*	*+ErrorMessage*	Gibt an, ob die Löschung erfolgreich war.

Tabelle 224: Beschreibung der Struktur ***PersonalisationDeleteValueResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

PERSONALISATIONDELETEVALU-EREQUEST_UNKNOWN_ID	Zum angegebenen Identifikator ist im Personalisierungsdienst kein Wert abgelegt.

Tabelle 225: Liste der Fehlerzustände in ***PersonalisationDeleteValueResponseStructure***.

20.5.5. PersonalisationEnumerateValuesResponseStructure

PersonalisationEnumerateValuesResponse			*+Structure*	**Enthält genauere Angaben zur Auflistung aller gespeicherten personalisierten Werte.**
	ErrorMessage	0:*	*+ErrorMessage*	Gibt an, ob der Abruf erfolgreich war.
	ValueId	0:*	*→ValueId*	Falls der Abruf erfolgreich war, ist für jeden gespeicherten Wert ein solches Element vorhanden, welches jeweils einen Identifikator eines gespeicherten Wertes enthält. Vgl. 20.3.

Tabelle 226: Beschreibung der Struktur ***PersonalisationEnumerateValuesResponseStructure***.

21. Dienst Fahrzeuginformationen

21.1. Beschreibung

Dieser Dienst dient dazu, dass zwischen einem Fahrzeug und einer mobilen Applikation, die von einem Fahrgast benutzt wird, Informationen, die das Fahrzeug betreffen, ausgetauscht werden können.

21.2. Anfragestrukturen

21.2.1. VehicleDataRequestStructure

VehicleDataRequestStructure			*+Structure*	**Fasst die Abfragedaten für eine Abfrage der Fahrzeugdaten zusammen.**
	VehicleStatus	0:1	*xs:boolean*	Es sollen vom Fahrzeug Fahrzeug-Status-Informationen geschickt werden.
	VehicleActivity	0:1	*xs:boolean*	Es sollen vom Fahrzeug Fahrzeug-Aktivitäts-Informationen geschickt werden.

Tabelle 227: Beschreibung der Struktur ***VehicleDataRequestStructure***.

21.3. Antwortstrukturen

21.3.1. VehicleDataResponseStructure

VehicleDataResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Abfrage der Fahrzeugdaten zusammen**
	VehicleCode	**1:1**	*→Vehicle-Code*	Eindeutige Fahrzeug-ID, mit der eine Referenzierung auf die EKAP-Daten ermöglicht werden soll, vgl. 7.4.1.
	VehicleStatus	0:1	*+VehicleStatus*	Informationen, die den Zustand des Fahrzeugs betreffen, siehe 21.3.2.
	VehicleActivity	0:1	*+VehicleActivity*	Informationen, die die Aktivität des Fahrzeugs betreffen, siehe 21.3.3.

Tabelle 228: Beschreibung der Struktur ***VehicleDataResponseStructure***.

21.3.2. VehicleStatusStructure

VehicleStatusStructure			*+Structure*	**Informationen, die den Zustand des einzelnen Fahrzeugs betreffen.**
	DoorState	0:1	*DoorsOpen | AllDoorsClosed*	Informationen über den Türzustand.
	VehicleStopRequested	0:1	*xs:boolean*	Information darüber, ob bereits ein Haltewunsch für die kommende Haltestelle dem Fahrzeugsystem bekannt ist.
	VehicleTypeRef	0:1	*→VehicleType*	Informationen über den Fahrzeugtyp und damit der Ausstattung des Fahrzeugs, vgl. 7.4.1.

Tabelle 229: Beschreibung der Struktur ***VehicleStatusStructure***.

21.3.3. VehicleActivityStructure

VehicleActivityStructure			*+Structure*	**Informationen, die die Aktivität des einzelnen Fahrzeugs betreffen.**
	TimetableDelay	0:1	*xs:int*	Abweichung gegenüber dem Fahrplan in Sekunden, Verfrühungen werden als negative Werte dargestellt.
	RouteDeviation	0:1	*onroute* | *offroute* | *unknown*	Information darüber, ob das Fahrzeug sich auf dem geplanten Linienweg befindet oder nicht.
	JourneyMode	0:1	*NoTrip* | *AdditionalTrip* | *ServiceTrip*	Information über die Art der Fahrt, die das Fahrzeug durchführt (Planfahrt, Verstärkerfahrt u. ä.).
Ser-vicePat-ternPo-sition	*StopSequence*	0:*	*+StopSequence*	Informationen zur Haltepunktsfolge, die benötigt wird, um bspw. eine Perlschnur darzustellen, siehe 21.3.4.
	CurrentStopIndex	0:1	*xs:int*	Index über den nächsten angefahrenen Haltepunkt in der Haltepunktfolge.
	LocationState		*AfterStop* | *AtStop* | *BeforeStop* | *BetweenStop*	Information darüber, ob das Fahrzeug sich noch an der Haltestelle befindet, kurz davor oder dahinter oder zwischen zwei Halten.
	NextExitSide	0:1	*both* |*left* |*right* | *unknown*	Information über die Ausstiegsseite am nächsten Haltepunkt.

Tabelle 230: Beschreibung der Struktur ***VehicleActivityStructure***.

21.3.4. StopSequenceStructure

StopSequenceStructure			*+Structure*	**Informationen zur Haltepunktfolge.**
	StopPoint	**2:***	*+StopInformation*	Informationen über den Haltepunkt, siehe 21.3.5.

Tabelle 231: Beschreibung der Struktur ***StopSequenceStructure***.

21.3.5. StopInformationStructure

StopInformationStructure			*+Structure*	**Informationen über den einzelnen Haltepunkt.**
	StopIndex	**1:1**	*xs:int*	Index des aktuellen Haltepunkts in der Haltepunkt-folge.
	StopRef	0:1	*→StopPoint*	Referenz auf den Haltepunkt, siehe 7.5.1.
	StopName	**1:***	*+InternationalText*	Name des Haltepunkts.
	StopAlternativeName	0:*	*+InternationalText*	Alternativbezeichnung des Haltepunkts.
	Platform	0:*	*xs:string*	Bezeichnung des Haltestellensteiges
	DisplayContent	0:*	*+DisplayContent*	Information über die Zusammensetzung des Ziel-textinhalts, siehe 21.3.6.
	ArrivalScheduled	0:1	*xs:dateTime*	Angabe der geplanten Ankunftszeit.
	DepartureScheduled	0:1	*xs:dateTime*	Angabe der geplanten Abfahrtszeit.
	RecordedArrivalTime	0:1	*xs:dateTime*	Information über die tatsächliche Ankunftszeit (wird für den Feldtest in Stuttgart und während des Mig-rationszeitraums benötigt).
	DistanceToNextStop	0:1	*xs:double*	Abstand zum nächsten Halt in [m].
	AnnouncementNext-Stop	0:*	*+Announcement*	Informationen über die Haltestellenansage, siehe 21.3.10.
	Farezone	0:*	*xs:NMTOKEN*	Informationen über die Tarifzonen, in denen dieser Haltepunkt liegt.
	Connection	0:*	*+Connection*	Informationen über Anschlüsse, siehe 21.3.11.

Tabelle 232: Beschreibung der Struktur ***StopInformationStructure***.

21.3.6. DisplayContentStructure

DisplayContentStructure			*+Structure*	**Information über die Zusammensetzung des einzelnen Zieltextinhalts.**
	Line	0:1	*+LineInformation*	Informationen zur Bezeichnung der Linie, siehe 21.3.7.
	Destination	**1:1**	*+Destination*	Informationen über den Inhalt des Zieltextes, siehe 21.3.8.
	Via	0:*	*+ViaPoint*	Informationen über Zwischenhaltestellen, siehe 21.3.9.
	AdditionalInformation	0:*	*+International Text*	Zusatzinformationen wie „Eilkurs“, „Entlastungsbus“ etc.
Dis-playPo-licy	***PeriodDuration***	**1:1**	*xs:duration*	Information über die Periodendauer bei Wechsel zwischen verschiedenen Anzeigeinhalten.
	Duration	**1:1**	*xs:duration*	Information über die Anzeigedauer dieses Anzeige-inhalts innerhalb einer Anzeigeperiode (bei Wechsel zwischen verschiedenen Anzeigeinhalten).

Tabelle 233: Beschreibung der Struktur ***DisplayContentStructure***.

21.3.7. LineInformationStructure

LineInformationStructure			*+Structure*	**Informationen zur Bezeichnung der Linie.**
	LineRef	**1:1**	*→Line*	Referenz auf eine Linie, siehe 7.4.1.
	LineName	**1:***	*+International Text*	Name der Linie.
	LineShortName	0:*	*+International Text*	Kurzbezeichnung der Linie.
	LineNumber	**1:1**	*xs:int*	Nummer der Linie.

Tabelle 234: Beschreibung der Struktur ***LineInformationStructure***

21.3.8. DestinationStructure

DestinationStructure			*+Structure*	**Informationen über den Inhalt des Zieltextes.**
	DestinationRef	**1:1**	*xs:NMTOKEN*	Index des Zieltextes.
	DestinationName	0:*	*+International Text*	Zieltext.
	DestinationShortName	0:*	*+International Text*	Zieltextkurzbezeichnung.

Tabelle 235: Beschreibung der Struktur ***DestinationStructure***

21.3.9. ViaPointStructure

ViaPointStructure			*+Structure*	**Informationen über Zwischenhaltestellen.**
	ViaPointRef	**1:1**	*xs:int*	Index des Haltepunkts innerhalb der Auflistung der Zwischenhaltepunkte.
	PlaceRef	0:1	*→StopPoint*	Referenz auf den Haltepunkt, siehe 7.5.1.
	PlaceName	0:*	*+International Text*	Bezeichnung des Zwischenhaltepunktes.
	PlaceShortName	0:*	*+International Text*	Kurzbezeichnung des Zwischenhaltepunktes.
	ViaPointDisplayPriority	0:1	*xs:nonNegativeInteger*	Anzeigepriorität des Zwischenhaltepunktes.

Tabelle 236: Beschreibung der Struktur ***ViaPointStructure***

21.3.10. AnnouncementStructure

AnnouncementStructure			*+Structure*	**Informationen über die einzelne Haltestellenansage**
	AnnouncementRef	**1:1**	*xs:NMTOKEN*	Index der Ansage.
	AnnouncementText	0:*	*+International Text*	Ansagetext als für den Fahrgast lesbare Information.
	AnnouncementTTSText	0:*	*+International Text*	Ansagetext, für ein TextToSpeech-System.

Tabelle 237: Beschreibung der Struktur ***AnnouncementStructure***

21.3.11. ConnectionStructure

ConnectionStructure			*+Structure*	**Information über die einzelnen Anschlüsse an einer Haltestelle inkl. Nachbarhaltestellen.**
	ConnectionRef	**1:1**	*xs:NMTOKEN*	Haltepunktsbezogener Index über alle verfügbaren Anschlüsse.
	ConnectionType	**1:1**	*Interchange | ProtectedConnection*	Art des Anschlusses (gesicherter Anschluss oder einfache Umsteigebeziehung).
	ConnectionStop	0:1	*→StopPoint*	Referenz auf einen Nachbarhaltepunkt, siehe 7.5.1. Falls nicht vorhanden, findet der Anschluss am selben Haltepunkt statt.
	DisplayContent	**1:1**	*+DisplayContent*	Information über die Zusammensetzung des Zieltextinhalts, siehe 21.3.6.
	Platform	**1:1**	*xs:string*	Abfahrtsstelle des Abbringers.
	WalkDuration	0:1	*xs:duration*	Durchschnittliche Fußwegzeit zum Abfahrtshaltepunkt des Anschlusses.
	ConnectionState	**1:1**	*ConnectionOK | ConnectionBroken | NoInformationAvailable*	Information darüber, ob der Anschluss gehalten werden kann oder nicht.
	Transportmode	0:1	*+VehicleType*	Information über die Art des Abbringer-Fahrzeugs, siehe 21.3.12.
	ExpectedDepartureTime	0:1	*xs:dateTime*	erwartete Abfahrtszeit des Abbringers.

Tabelle 238: Beschreibung der Struktur ***ConnectionStructure***

21.3.12. VehicleTypeStructure

VehicleTypeStructure			*+Structure*	**Information über die Art eines Fahrzeugs.**
	VehicleTypeRef	**1:1**	*→VehicleType*	Informationen über den Fahrzeugtyp und damit der Ausstattung des Fahrzeugs, vgl. 7.4.1.
	VehicleName	0:*	*+International Text*	Bezeichner des Fahrzeugtyps.

Tabelle 239: Beschreibung der Struktur ***VehicleTypeStructure***

22. Dienst Haltewunsch an Fahrzeug

22.1. Beschreibung

Dieser Dienst dient der Übertragung eines Haltewunsches von einer mobilen Applikation eines Fahrgasts an das Fahrzeug.

22.2. Anfragestrukturen

22.2.1. StopRequestRequestStructure

StopRequestRequestStructure			*+Structure*	**Fasst die Informationen über die Anfragestruktur bei einem Haltewunsch an das Fahrzeug zusammen.**
	StopRef	**1:1**	*→StopPoint*	Referenz auf die gewünschte Ausstiegshaltestelle, vgl. 7.5.1.
	StopName	0:*	*+International Text*	Bezeichner der Ausstiegshaltestelle.
	DisabledPerson	0:1	*xs:boolean*	Information darüber, ob die anfragende Person mobilitätseingeschränkt ist.

Tabelle 240: Beschreibung der Struktur ***StopRequestRequestStructure***

22.3. Antwortstrukturen

22.3.1. StopRequestResponseStructure

StopRequestResponseStructure			*+Structure*	**Fasst die Informationen über die Antwortstruktur bei einem Haltewunsch an das Fahrzeug zusammen.**
	StopRequestReceived	**1:1**	*xs:boolean*	Information darüber, dass der Haltewunsch angekommen ist.
	RequestedStop	0:1	*xs:NMTOKEN*	Referenz auf die gewünschte Ausstiegshaltestelle, vgl. 7.5.1.
	EstimatedArrivalTime	0:1	*xs:dateTime*	Prognostizierte Ankunftszeit an der Ausstiegshaltestelle.

Tabelle 241: Beschreibung der Struktur ***StopRequestResponseStructure***

23. Dienst Diensteregister

23.1. Beschreibung

Der Diensteregisterdienst führt Buch über verfügbare TRIAS-Dienste.

23.2. Einfache Typen

Folgende einfache Typen (simple types) werden definiert:

TriasServiceIdType	*xs:NMTOKEN*	**ID einer Instanz eines TRIAS-Dienstes.**
InterfaceVersionType	*xs:NMTOKEN*	Versionsnummer einer Dienstschnittstellendefinition.
TriasServiceTypeEnumeration	*Alerts* | *BookingInfo* | *ConnectionDemand* | *Facilities* | *Fares* | *IndividualTrips* | *Locations* | *Maps* | *Positioning* | *ServiceRegister* | *StopEvents* | *TripInfo* | *Trips*	Art des Diensts.
ServiceAddressType	*xs:anyURI*	Adresse (URL) eines Online-Dienstes.
ServiceUsageEnumeration	*Consumer* | *Provider*	Verwendung des Dienstes als Anbieter oder Client.

Tabelle 242: Beschreibung von einfachen Typen

23.3. Anfragestrukturen

Eine Anfrage an das Diensteregister wird mittels eines Elements ***ServiceRegisterRequest*** vom Typ *ServiceRegisterRequestStructure* gestellt.

23.3.1. ServiceRegisterRequestStructure

Mit einer Anfrage vom Typ *ServiceRegisterRequestStructure* kann ein TRIAS-Dienst im Dienstregister wahlweise aufgenommen, gelöscht oder aktualisiert werden, oder es können alle registrierten Dienste gesucht werden, die auf die angegebenen Filterkriterien passen.

ServiceRegisterRequestStructure				*+Structure*	**Fasst die Anfragedaten an das Diensteregister zusammen.**
	a	***RegisterRequest***	**-1:1**	*+ServiceRegisterRegis-terRequest*	Anfrage, um eine Dienstinstanz im Dienstregister anzumelden. Vgl. 23.3.3.
	b	***UpdateRequest***		*+ServiceRegisterUpdateRequest*	Anfrage, um eine Dienstinstanz im Dienstregisterzu aktualisieren. Vgl. 23.3.4.
	c	***LookupRequest***		*+ServiceRegisterLookupRe-quest*	Anfrage, um nach passenden Diensten im Dienstregister zu suchen. 23.3.5.
	d	***UnregisterRequest***		*+ServiceRegisterUnregis-terRequest*	Anfrage, um eine Dienstinstanz im Dienstregister zu löschen. Vgl. 23.3.6.
	Params		0:1	*+ServiceRegisterParam*	Anfrageparameter. Vgl. 23.3.2.

Tabelle 243: Beschreibung der Struktur ***ServiceRegisterRequestStructure***.

23.3.2. ServiceRegisterParamStructure

ServiceRegisterParamStructure			*+Structure*	**Fasst die Parameter für eine Anfrage an das Diensteregister zusammen.**
	Extension	0:1	*xs:anyType*	Erweiterungen.

Tabelle 244: Beschreibung der Struktur ***ServiceRegisterParamStructure***.

23.3.3. ServiceRegisterRegisterRequestStructure

ServiceRegisterRegisterRequestStructure			*+Structure*	**Anfrage an das Diensteregister, um einen Dienst anzumelden.**
TriasServicePropertiES	*ServiceType*	0:1	*TriasServiceTypeEnumeration*	Art des Dienstes.
	Version	0:1	*InterfaceVersion*	Versionsnummer der Dienstschnittstellendefinition.
	ServiceAddress	0:1	*ServiceAddress*	Adresse (URL) eines Online-Dienstes.
	ParticipantRef	0:1	*→ParticipantCode*	ID eines Kommunikationspartners. Vgl. 7.4.1.
	ServiceUsage	0:1	*Consumer | Provider*	Verwendung des Dienstes als Anbieter oder Client.

Tabelle 245: Beschreibung der Struktur ***ServiceRegisterRegisterRequestStructure***.

23.3.4. ServiceRegisterUpdateRequestStructure

ServiceRegisterUpdateRequestStructure			*+Structure*	**Anfrage an das Diensteregister, um die Eintragungen zu einem Dienst zu aktualisieren.**
	ServiceId	**1:1**	*TriasServiceId*	ID des Diensts, der aktualisiert werden soll.
TriasServicePropertiES	*ServiceType*	0:1	*TriasServiceTypeEnumeration*	Art des Dienstes.
	Version	0:1	*InterfaceVersion*	Versionsnummer der Dienstschnittstellendefinition.
	ServiceAddress	0:1	*ServiceAddress*	Adresse (URL) eines Online-Dienstes.
	ParticipantRef	0:1	*→ParticipantCode*	ID eines Kommunikationspartners. Vgl. 7.4.1.
	ServiceUsage	0:1	*Consumer | Provider*	Verwendung des Dienstes als Anbieter oder Client.

Tabelle 246: Beschreibung der Struktur ***ServiceRegisterUpdateRequestStructure***.

23.3.5. ServiceRegisterLookupRequestStructure

ServiceRegisterLookupRequestStructure			*+Structure*	**Anfrage an das Diensteregister, um passende Dienste zu finden.**
	ServiceId	0:1	*TriasServiceId*	ID des Diensts, nach dem gesucht wird.
TriasServiceProperties	*ServiceType*	0:1	*TriasServiceTypeEnumeration*	Art des Dienstes.
	Version	0:1	*InterfaceVersion*	Versionsnummer der Dienstschnittstellendefinition.
	ServiceAddress	0:1	*ServiceAddress*	Adresse (URL) eines Online-Dienstes.
	ParticipantRef	0:1	*→ParticipantCode*	ID eines Kommunikationspartners. Vgl. 7.4.1.
	ServiceUsage	0:1	*Consumer | Provider*	Verwendung des Dienstes als Anbieter oder Client.

Tabelle 247: Beschreibung der Struktur ***ServiceRegisterLookupRequestStructure***.

23.3.6. ServiceRegisterUnregisterRequestStructure

ServiceRegisterUnregisterRequestStructure			*+Structure*	Anfrage an das Diensteregister, um einen Dienst zu löschen.
	ServiceId	0:1	*TriasServiceId*	ID des Diensts, der gelöscht werden soll.

Tabelle 248: Beschreibung der Struktur ***ServiceRegisterUnregisterRequestStructure***.

23.4. Antwortstrukturen

Das Ergebnis einer Dienstregisteranfrage wird mittels eines Elements ***ServiceRegisterResponse*** vom Typ *ServiceRegisterResponseStructure* übertragen.

23.4.1. ServiceRegisterResponseStructure

ServiceRegisterResponseStructure			*+Structure*	**Fasst die Ergebnisdaten für eine Dienstregisteranfrage zusammen.**
	ErrorMessage	0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Gesamtbeantwortung der Anfrage. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	ServiceRegisterResult	0:1	*+ServiceRegisterResult*	Struktur für ein Dienstregisterresultat. Vgl. 23.4.2.

Tabelle 249: Beschreibung der Struktur ***ServiceRegisterResponseStructure***.

In *ErrorMessage* können folgende Fehlerzustände auftreten:

SERVICEREGISTER_SERVICEIDUNKNOWN	Die Anfrage an das Dienstregister enthält eine unbekannte Service-ID.
SERVICEREGISTER_NOMATCH	Die Suchanfrage an das Dienstregister liefert keine Übereinstimmung.
SERVICEREGISTER_TOOMANYMATCHES	Die Suchanfrage an das Dienstregister liefert zu viele Übereinstimmungen

Tabelle 250: Liste der Fehlerzustände in ***ServiceRegisterResponse***.

23.4.2. ServiceRegisterResultStructure

ServiceRegisterResultStructure				*+Structure*	**Resultatstruktur für die Diensteregisteranfrage.**
	ResultId		**1:1**	*xs:NMTOKEN*	ID des Resultats für spätere Referenzierung.
	ErrorMessage		0:*	*+ErrorMessage*	Fehlermeldungen bezogen auf die Anfrage an das Diensteregister. Siehe die nachstehende Tabelle für mögliche Werte. Vgl. 7.4.2.
	a	***RegisterResponse***	**-1:1**	*+ServiceRegisterRegister-Response*	Antwort zur Dienstregistrierung. Vgl. 23.4.3.
	b	***UpdateResponse***		*+ServiceRegisterUpdate-Response*	Antwort zur Dienstaktualisierung. Vgl. 23.4.4.
	c	***LookupResponse***		*+ServiceRegisterLooku-pResponse*	Antwort zur Suche nach passenden Diensten. Vgl. 23.4.5.
	d	***UnregisterRespon-se***		*+ServiceRegisterUnregister-Response*	Antwort zur Dienstlöschung. Vgl. 23.4.6.

Tabelle 251: Beschreibung der Struktur ***ServiceRegisterResultStructure***.

Die Verwendung von *ErrorMessage* in *ServiceRegisterResultStructure* ist für spätere Erweiterungen reserviert.

23.4.3. ServiceRegisterRegisterResponseStructure

ServiceRegisterRegisterResponseStruc-ture			*+Structure*	**Antwort auf die Dienstregistrierung.**
	ServiceId	**1:1**	*TriasServiceId*	ID des Diensts, wie er im Register geführt wird. Diese ID muss bei Folgeanfragen an das Diensteregister verwendet werden.

Tabelle 252: Beschreibung der Struktur ***ServiceRegisterRegisterResponseStructure***.

23.4.4. ServiceRegisterUpdateResponseStructure

ServiceRegisteUpdateResponseStructure			*+Structure*	**Antwort auf die Dienstaktualisierung.**
	ServiceId	**1:1**	*TriasServiceId*	ID des Diensts, wie er im Register geführt wird. Diese ID muss bei Folgeanfragen an das Diensteregister verwendet werden.

Tabelle 253: Beschreibung der Struktur ***ServiceRegisterUpdateResponseStructure***.

23.4.5. ServiceRegisterLookupResponseStructure

ServiceRegisteLookupResponseStructure			*+Structure*	**Antwort auf eine Dienstsuche.**
	Service	**1:***	*TriasService*	Ein oder mehrere Dienste, die auf die Anfragekriterien passen. Vgl. 23.4.7.

Tabelle 254: Beschreibung der Struktur ***ServiceRegisterLookupResponseStructure***.

23.4.6. ServiceRegisterUnregisterResponseStructure

ServiceRegisteUnregisterResponseStruc-ture			*+Structure*	**Antwort auf eine Dienstlöschung aus dem Register.**
				Das Antwortelement bleibt bis auf weiteres leer.

Tabelle 255: Beschreibung der Struktur ***ServiceRegisterUnregisterResponseStructure***.

23.4.7. TriasServiceStructure

TriasServiceStructure			*+Structure*	**Definition einer Instanz eines TRIAS-Dienstes.**
	ServiceId	**1:1**	*TriasServiceId*	ID der Instanz.
TriasServicePropertiES	*ServiceType*	0:1	*TriasServiceTypeEnumeration*	Art des Dienstes.
	Version	0:1	*InterfaceVersion*	Versionsnummer der Dienstschnittstellendefinition.
	ServiceAddress	0:1	*ServiceAddress*	Adresse (URL) eines Online-Dienstes.
	ParticipantRef	0:1	*→ParticipantCode*	ID eines Kommunikationspartners. Vgl. 7.4.1.
	ServiceUsage	0:1	*Consumer | Provider*	Verwendung des Dienstes als Anbieter oder Client.

Tabelle 256: Beschreibung der Struktur ***TriasServiceStructure***.

24. Dienst Authentifizierung

Einige Teile der TRIAS-Schnittstelle werden zum Austausch von wichtigen betrieblichen Daten verwendet. Um die Unverfälschtheit dieser Daten sicherzustellen, sind Mechanismen zur Authentifizierung und zur Autorisierung der Schnittstellenpartner notwendig. In diesem Kapitel soll daher ein Mechanismus beschrieben werden, um die Authentizität von Schnittstellenpartnern überprüfen zu können.

Zur Authentifizierung der Schnittstellenpartner kommt ein PKI-Verfahren zum Einsatz. Es wird die Verwendung des DSA-Verfahrens[8] empfohlen. Jedoch können auch andere Verfahren zum Einsatz kommen. Die Kommunikationspartner müssen sich dazu bilateral abstimmen.

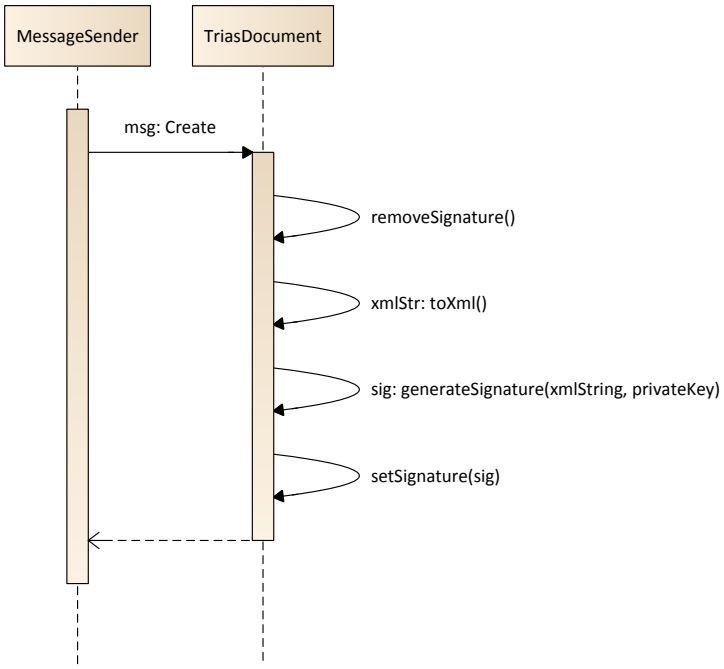

Abbildung 13: Erzeugung einer Nachrichtensignatur

[8] Digital Signature Standard: http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf

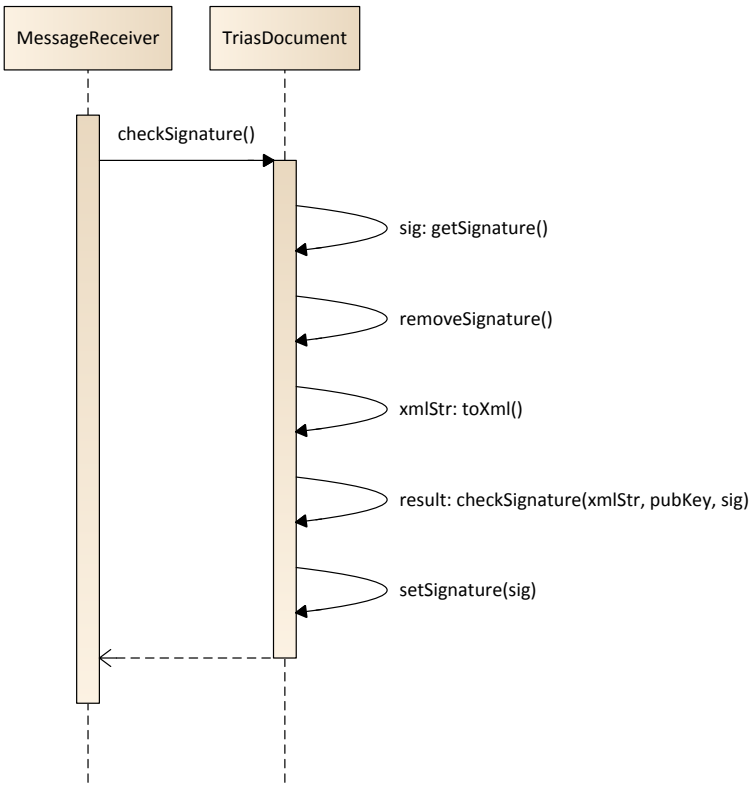

Abbildung 14: Überprüfung einer Signatur

Der Authentifizierungsdienst des Empfängers überprüft die CertificateId und die SignatureId aus dem ServiceRequest (vgl. die Definition der Nachrichteneigenschaften in 7.9.2). Das Feld CertificateId enthält eine Referenz auf diesen Schlüssel. Der Schlüssel muss also vorab ausgetauscht werden. Das Feld SignatureId enthält die Signatur der versendeten TRIAS-Nachricht. Zur Berechnung der Signatur wird die XML-Nachricht in der kanonischen Normalform[9] verwendet. Bei der Berechnung der Signatur darf das Feld SignatureId in der XML-Nachricht nicht vorhanden sein, es muss also vom Empfänger wieder entfernt werden. Dabei muss die kanonische Normalform erhalten bleiben.

Der Ablauf der Signierung einer Nachricht ist schematisch in Abbildung 13 dargestellt. Zunächst wird das Feld SignatureId komplett aus der Nachricht entfernt. Dann wird die Nachricht in ihre XML-Repräsentation in der kanonischen Normalform exportiert. Mithilfe der XML-Repräsentation und dem privaten Schlüssel kann nun die korrekte Signatur der Nachricht berechnet und im Feld SignatureId gespeichert werden. Das Ergebnis ist die korrekt signierte, und vor Modifikation durch Dritte geschützte Nachricht.

Der schematische Ablauf der Signaturprüfung ist in Abbildung 14 dargestellt. Zunächst wird das Feld SignatureId aus der Nachricht entfernt und die Nachricht in ihre kanonische XML-Darstellung überführt. Anschließend kann mithilfe des öffentlichen Schlüssels des Nachrichtensenders, der Nachricht in XML-Darstellung und der Signatur überprüft werden, ob diese Signatur gültig ist. Anschließend muss, um die Nachricht nicht zu verfälschen, die Signatur wieder auf ihren ursprünglichen Wert gesetzt werden.

[9] RFC 3076: http://www.ietf.org/rfc/rfc3076.txt

25. Normative Referenzen

- CEN-IFOPT (CEN, EN 28701:2012, 2012)
- ISO 8601 Zeitformate (ISO 8601:2004, 2004)

26. Begriffe

Es gelten die in VDV-Schrift 430 Teil 1, 2013 und in VDV-Schrift 431, Teil 1, 2013 festgelegten Begriffsbestimmungen.

27. Versionshistorie

27.1. Version 1.1 (Schrift und Schema)

27.1.1. Funktionale Erweiterungen

- Schema-Version im Wurzelelement auf „1.1“ erhöht.
- Neue Struktur *WebLinkStructure* (Link + Beschreibung) definiert als Ersatz für Elemente vom Typ *xs:anyUri*, die nur einen Link ohne Beschreibung darstellen.
- In Trias_FaresSupport.xsd eine neue Struktur *FareZoneStructure* aufgenommen, um zu den Referenzen auf Tarifzonenobjekte auch Tarifzonenbezeichnungen angeben zu können. Wird z.B. in *PassedZones* oder *ZonesAlreadyPaid* verwendet.
- Dem Element *FaresAuthorityRef* wird ein neues Element *FaresAuthorityText* beiseite gestellt, um Tarifverantwortliche mit Bezeichnungen benennen zu können.
- In *TicketStructure* weitere Elemente *ValidityDuration*, *ValidityDurationText*, *ValidityFareZones* und *ValidityAreaText*.
- *StaticFaresRequestStructure* bietet nun die Möglichkeit, eine TicketID anzugeben, um Informationen speziell zu diesem Ticket abzurufen.
- *FaresPassengerStructure* um *ZonesAlreadyPaid* und *OwnedTickets* erweitert, um die bereits vorhandenen Fahrberechtigungen modellieren zu können.
- POI-Kategorien (*POICategory*) als Liste von Key-Value-Paaren (Attribute aus Open Street Map) eingeführt. Ein POI kann einer oder mehreren POI-Kategorien zugeordnet werden. Die Suche nach POIs kann nach diesen POI-Kategorien gefiltert werden (in *LocationParamStructure*).
- Neues Element *OperatorFilter* in *LocationParamStructure* aufgenommen.
- Neues Element *SharingService* zur Modellierung von Fahrzeugverleihanbietern (Car-Sharing, Bike-Sharing) in *ContinuousServiceStructure* aufgenommen.
- In der *TripRequestStructure* werden die Elemente *Origin* und *Destination* nun auch mehrfach zugelassen.

27.1.2. Technische Ergänzungen/Korrekturen

- Versionsnummers aus den Dateinamen der Schemadateien entfernt.
- Mehrfache import-Anweisungen aus dem siri-Namensraum pro Schemadatei entfernt. Stattdessen Import der SIRI-Haupt-Schemadatei.
- Einheitliche Vergabe des TRIAS-Namensraums „www.vdv.de/trias“.
- In *NavigationSectionStructure* Korrektur der Schreibung von *Manoeuvre*.
- Im Benachrichtigungsdienst (*TripMonitoringPolicyGroup*) jeweils den Defaultwert *false* für die Elemente *AcceptThirdPartyInformation* und *IncludeAlternatives* aufgenommen.
- Die beiden unbenutzten Schemadateien Trias_Authorisation.xsd und Trias_PushToDevice.xsd entfernt. Das waren Relikte aus dem Forschungsprojekt IP-KOM-ÖV, die nicht mehr benötigt werden.
- Tippfehler im Element *IndvidualTransportOptions* (sic!) in der *LocationContextStructure* korrigiert.
- In *DatedJourneyStructure* und ContinuousServiceStructure sind *OriginStopPointRef* bzw. *DestinationStopPointRef* nun optional, wenn man die *ServiceOriginGroup* bzw. *ServiceDestinationGroup* verwenden möchte.

- Weitere Fehlercodes *TRIASGENERIC_ERROR, TRIASGENERIC_SERVICENOTSUPPORTED*, *TRIASGENERIC_REQUESTNOTSUPPORTED* und *TRIASGENERIC_FEATURENOTSUPPORTED definiert.*
- Trias_JourneySupport.xsd:DatedCallAtLocationStructure->ServiceDeparture steht <xs:documentation>Arrival times of the service at this stop.</xs:documentation> Da es das ServiceDeparture-Element ist, steht nun stattdessen "Departure times".

27.1.3. **Dokumentarische Korrekturen**

- In *TicketStructure* die Dokumentation der Elemente *InfoUrl* und *SaleUrl* ergänzt.
- In *AddressStructure* das Element *PrivateCode* an die richtige Position verschoben.
- In *AddressStructure* das Element *CityName* in die korrekte Benennung *LocalityName* umbenannt.
- In *AddressStructure* die Dokumentation des Elements *LocalityRef* ergänzt.
- In *LocationStructure* die Referenzen auf die Ortsobjekte in die im Schema vorhandenen Kindelemente gewandelt.
- Diverse Tippfehler, Formatierungsfehler und falsche/fehlende Querverweise korrigiert.
- Erläuterung zum Verkehrstag und Betriebstag-Code in Kapitel 5.9 ergänzt
- Kapitel 5.3 Ergänzung mit Verweis auf OSM
- Ergänzung 7.5.5. PointOfInterestCategoryStructure und 7.5.6. OsmTagStructure
- Ergänzungen und Klärungen im Kapitel 19 zum Ablauf (*DataReceivedAcknowledgement*)

28. Abkürzungen

Es gelten die in VDV-Schrift 430 Teil 1, 2013 und in VDV-Schrift 431, Teil 1, 2013 angegebenen Abkürzungen. Darüber hinaus werden die in Tabelle 257 aufgeführten Abkürzungen verwendet.

Begriff	Beschreibung
HTTP	Hypertext Transfer Protocol. Übertragungsprotokoll basierend auf TCP/IP, wird vor allem im Internet zum Austausch von Informationen benutzt.
IFOPT	Identification of Fixed Objects in Public Transport (CEN, EN 28701:2012, 2012)
JourneyWeb	Britischer Standard zur Verknüpfung von regionalen Fahrplanauskunftssystemen für eine Großbritannienweite nationale Auskunft (Department for Transport, 2012)
Transmodel	Reference Data Model for Public Transport (CEN, EN 12896:2006, 2006)
TRIAS	Travellers‘ Realtime Information and Advisory Standard. Mit diesem Akronym wird die Familie der in diesem Dokument definierten Schnittstellendienste bezeichnet.
UML	Unified Modeling Language Standardisierte, grafische Modellierungssprache zur Spezifikation von Software.

Tabelle 257: Liste der Abkürzungen.

29. Abbildungsverzeichnis

Abbildung 1: Umfeld und Schwerpunkte im Projekt IP-KOM-ÖV 17

Abbildung 2: Anfrage mit synchroner Antwort (Abbildung entnommen aus SIRI, (CEN, TS 15531, Part 2)). 22

Abbildung 3: Abonnement-Verfahren mit asynchronen Benachrichtigungen (Abbildung entnommen aus SIRI, (CEN, TS 15531, Part 2)).....23

Abbildung 4: Gesamtübersicht über die Hierarchie der Schemadateien. 35

Abbildung 5: Linker Teil der Schema-Hierarchie.....36

Abbildung 6: Rechter Teil der Schema-Hierarchie.....36

Abbildung 7: Ablauf der Anschlussdienste mit aktiver Benachrichtigung bei Statusänderung 102

Abbildung 8: Ablauf der Anschlussdienste mit passiver Benachrichtigung bei Statusänderung 102

Abbildung 9: Funktionsweise des Benachrichtigungsdienstes.....134

Abbildung 10: Sequenzdiagramm Lebenszyklus eines Wertes.....138

Abbildung 11: Sequenzdiagramm Werteliste ermitteln 139

Abbildung 12: Sequenzdiagramm Personalisierung 140

Abbildung 13: Erzeugung einer Nachrichtensignatur.....154

Abbildung 14: Überprüfung einer Signatur.....155

30. Tabellenverzeichnis

Tabelle 1: Beispiel (aus einem späteren Abschnitt) für die tabellarische Notation einer XML-Struktur.....20

Tabelle 2: Liste der TRIAS-Dienste und ihrer Anfrageelemente. 33

Tabelle 3: Liste der Fehlercodes, wie sie in SIRI für das Nachrichtenübermittlungsverfahren definiert werden. 38

Tabelle 4: Generische TRIAS-Fehlermeldungen, die in allen Nachrichten auftreten können.....39

Tabelle 5: Beschreibung der Struktur ***ServiceRequestStructure***. 40

Tabelle 6: Beschreibung der Struktur ***SubscriptionRequestStructure***. 40

Tabelle 7: Beschreibung der Struktur ***RequestPayloadStructure***. 41

Tabelle 8: Beschreibung der Struktur ***ServiceDeliveryStructure***. 42

Tabelle 9: Beschreibung der Struktur ***DeliveryPayloadStructure***. 43

Tabelle 10: Liste der einfachen Typdefinitionen in Trias_Utility.xsd. 43

Tabelle 11: Beschreibung der Struktur ***InternationalTextStructure***. 44

Tabelle 12: Beschreibung der Struktur ***GeoPositionStructure***.....44
Tabelle 13: Beschreibung der Struktur ***WebLinkStructure***.....44
Tabelle 14: Liste der einfachen Typdefinitionen in Trias_ModesSupport.xsd. 46
Tabelle 15: Beschreibung der Struktur ***IndividualTransportOptionsStructure***. 46
Tabelle 16: Beschreibung der Gruppe ***PtSubmodeChoiceGroup***.....47
Tabelle 17: Beschreibung der Struktur ***ModeStructure***. 47
Tabelle 18: Beschreibung der Struktur ***PtModeFilterStructure***. 48
Tabelle 19: Liste der einfachen Typdefinitionen in Trias_Common.xsd. 48
Tabelle 20: Beschreibung der Struktur ***ErrorMessageStructure***.....48
Tabelle 21: Beschreibung der Struktur ***PrivateCodeStructure***.....49
Tabelle 22: Beschreibung der Struktur ***OperatorFilterStructure***.....49
Tabelle 23: Beschreibung der Struktur ***LineDirectionStructure***.....49
Tabelle 24: Beschreibung der Struktur ***LineDirectionFilterStructure***. 49
Tabelle 25: Beschreibung der Struktur ***SharingServiceStructure***.....50
Tabelle 26: Beschreibung der Struktur ***OperatingDaysStructure***. 50
Tabelle 27: Beschreibung der Struktur ***WeekdayTimePeriodStructure***.....50
Tabelle 28: Beschreibung der Struktur ***GeneralAttributeStructure***.....51
Tabelle 29: Liste der einfachen Typdefinitionen in Trias_LocationSupport.xsd.....51
Tabelle 30: Beschreibung der Struktur ***StopPointStructure***.....52
Tabelle 31: Beschreibung der Struktur ***StopPlaceStructure***. 52
Tabelle 32: Beschreibung der Struktur ***LocalityStructure***. 53
Tabelle 33: Beschreibung der Struktur ***PointOfInterestStructure***.....53
Tabelle 34: Beschreibung der Struktur ***PointOfInterestCategoryStructure***.....53
Tabelle 35: Beschreibung der Struktur ***OsmTagStructure***. 54
Tabelle 36: Beschreibung der Struktur ***PointOfInterestFilterStructure***. 54
Tabelle 37: Beschreibung der Struktur ***AddressStructure***. 54
Tabelle 38: Beschreibung der Struktur ***LocationStructure***.....55
Tabelle 39: Beschreibung der Struktur ***LocationRefStructure***. 55
Tabelle 40: Beschreibung der Struktur ***ServiceViaPointStructure***.....56
Tabelle 41: Beschreibung der Struktur ***ViaStructure***. 56
Tabelle 42: Beschreibung der Gruppe ***DatedServiceGroup***.....57
Tabelle 43: Beschreibung der Struktur ***DatedJourneyStructure***.....57
Tabelle 44: Beschreibung der Struktur ***TripLocationStructure***. 58
Tabelle 45: Beschreibung der Struktur ***CallAtStopStructure***. 58
Tabelle 46: Beschreibung der Struktur ***DatedCallAtLocationStructure***.....59

Tabelle 47: Beschreibung der Struktur ***ContinuousServiceStructure***. 60
Tabelle 48: Beschreibung der Struktur ***VehiclePositionStructure***. 60
Tabelle 49: Beschreibung der Struktur ***ProgressBetweenStopsStructure***. 60
Tabelle 50: Beschreibung der Struktur ***LocationContextStructure***.....61
Tabelle 51: Beschreibung der Struktur ***AbstractResponseContextStructure***. 61
Tabelle 52: Beschreibung der Struktur ***LegAttributeStructure***.....61
Tabelle 53: Beschreibung der Gruppe ***siri:CommonFacilityGroup***. 64
Tabelle 54: Beschreibung der Gruppe ***siri:StopFacilityGroup***.....65
Tabelle 55: Beschreibung der Gruppe ***siri:ServiceFacilityGroup***. 65
Tabelle 56: Beschreibung der Gruppe ***siri:AllFacilitiesGroup***. 66
Tabelle 57: Beschreibung der Struktur ***SituationsStructure***. 66
Tabelle 58: Beschreibung der Struktur ***SituationFullRefStructure***. 67
Tabelle 59: Liste der einfachen Typdefinitionen in Trias_RequestSupport.xsd. 68
Tabelle 60: Beschreibung der Struktur ***AbstractTriasServiceRequestStructure***. 68
Tabelle 61: Beschreibung der Struktur ***AbstractTriasSubscriptionRequestStructure***. 69
Tabelle 62: Beschreibung der Struktur ***AbstractTriasResponseStructure***. 70
Tabelle 63: Liste der einfachen Typdefinitionen in Trias_FaresSupport.xsd. 71
Tabelle 64: Beschreibung der Struktur ***FareZoneStructure***.....71
Tabelle 65: Beschreibung der Struktur ***FareZoneListInAreaStructure***. 71
Tabelle 66: Beschreibung der Struktur ***BookingInfoStructure***. 72
Tabelle 67: Beschreibung der Struktur ***TicketStructure***. 72
Tabelle 68: Beschreibung der Struktur ***TripFaresResultStructure***.....73
Tabelle 69: Liste der Fehlerzustände in ***TripFaresResult***.....73
Tabelle 70: Beschreibung der Struktur ***FaresPassengerStructure***. 73
Tabelle 71: Beschreibung der Struktur ***FaresParamStructure***. 74
Tabelle 72: Liste der einfachen Typdefinitionen in Trias_Locations.xsd. 75
Tabelle 73: Beschreibung der Struktur ***LocationInformationRequestStructure***. 76
Tabelle 74: Beschreibung der Struktur ***InitialLocationInputStructure***. 76
Tabelle 75: Beschreibung der Struktur ***GeoRestrictionsStructure***. 76
Tabelle 76: Beschreibung der Struktur ***GeoCircleStructure***. 77
Tabelle 77: Beschreibung der Struktur ***GeoRectangleStructure***. 77
Tabelle 78: Beschreibung der Struktur ***GeoAreaStructure***.....77
Tabelle 79: Beschreibung der Struktur ***LocationParamStructure***. 78

Tabelle 80: Beschreibung der Struktur ***LocationInformationResponseStructure***. 79
Tabelle 81: Liste der Fehlerzustände in ***LocationInformationResponse***.....79
Tabelle 82: Beschreibung der Struktur ***LocationResultStructure***. 79
Tabelle 83: Beschreibung der Struktur ***TripRequestStructure***. 80
Tabelle 84: Beschreibung der Struktur ***TripParamStructure***. 82
Tabelle 85: Beschreibung der Gruppe ***NumberOfResultsGroup***. 82
Tabelle 86: Beschreibung der Struktur ***NotViaStructure***. 82
Tabelle 87: Beschreibung der Struktur ***NoChangeAtStructure***. 82
Tabelle 88: Beschreibung der Struktur ***TripResponseStructure***. 83
Tabelle 89: Liste der Fehlerzustände in ***TripResponse***. 83
Tabelle 90: Beschreibung der Struktur ***TripResponseContextStructure***. 83
Tabelle 91: Beschreibung der Struktur ***TripResultStructure***. 84
Tabelle 92: Liste der Fehlerzustände in ***TripResult***. 84
Tabelle 93: Beschreibung der Struktur ***TripStructure***.....85
Tabelle 94: Beschreibung der Struktur ***TripLegStructure***.....85
Tabelle 95: Beschreibung der Struktur ***TimedLegStructure***. 86
Tabelle 96: Beschreibung der Struktur ***InterchangeLegStructure***. 87
Tabelle 97: Beschreibung der Struktur ***ContinuousLegStructure***.....88
Tabelle 98: Beschreibung der Struktur ***LegBoardStructure***. 89
Tabelle 99: Beschreibung der Struktur ***LegAlightStructure***. 90
Tabelle 100: Beschreibung der Struktur ***LegIntermediateStructure***. 91
Tabelle 101: Beschreibung der Struktur ***LegTrackStructure***. 91
Tabelle 102: Beschreibung der Struktur ***TrackSectionStructure***. 92
Tabelle 103: Beschreibung der Struktur ***NavigationPathStructure***. 92
Tabelle 104: Beschreibung der Struktur ***NavigationSectionStructure***. 92
Tabelle 105: Beschreibung der Struktur ***AccessPathStructure***.....93
Tabelle 106: Beschreibung der Struktur ***StopEventRequestStructure***. 94
Tabelle 107: Beschreibung der Struktur ***StopEventParamStructure***. 94
Tabelle 108: Beschreibung der Struktur ***StopEventResponseStructure***. 95
Tabelle 109: Liste der Fehlerzustände in ***StopEventResponse.*** 95
Tabelle 110: Beschreibung der Struktur ***StopEventResponseContextStructure***. 95
Tabelle 111: Beschreibung der Struktur ***StopEventResultStructure***. 95
Tabelle 112: Liste der Fehlerzustände in ***StopEventResult.*** 96
Tabelle 113: Beschreibung der Struktur ***StopEventStructure***. 96

Tabelle 114: Beschreibung der Struktur ***CallAtNearStopStructure***.....96
Tabelle 115: Beschreibung der Struktur ***PositioningRequestStructure***. 97
Tabelle 116: Beschreibung der Struktur ***PositioningParamStructure***. 97
Tabelle 117: Beschreibung der Struktur ***TimedPositionStructure***. 98
Tabelle 118: Beschreibung der Struktur ***TimedStopStructure***. 98
Tabelle 119: Beschreibung der Struktur ***PositioningResponseStructure***. 98
Tabelle 120: Liste der Fehlerzustände in ***PositioningResponse.***.....99
Tabelle 121: Beschreibung der Struktur ***PositioningResultStructure***. 99
Tabelle 122: Beschreibung der Struktur ***PositioningStructure***. 99
Tabelle 123: Beschreibung der Struktur ***RankedPositionStructure***.....99
Tabelle 124: Beschreibung der Struktur ***TripInfoRequestStructure***. 100
Tabelle 125: Beschreibung der Struktur ***TripInfoParamStructure***.....100
Tabelle 126: Beschreibung der Struktur ***TripInfoResponseStructure***. 101
Tabelle 127: Liste der Fehlerzustände in ***TripInfoResponse***.....101
Tabelle 128: Beschreibung der Struktur ***TripInfoResponseContextStructure***. 101
Tabelle 129: Beschreibung der Struktur ***TripInfoResultStructure***.....101
Tabelle 130: Beschreibung der einfachen Typen 104
Tabelle 131: Beschreibung der Struktur ***DatedConnectionStructure*** 105
Tabelle 132: Beschreibung der Struktur ***FeederDistributorStructure***.....105
Tabelle 133: Beschreibung des Typs ***GeneralizedConnectionStructure*** 105
Tabelle 134: Beschreibung der Struktur ***ConnectionStatusStructure*** 106
Tabelle 135: Beschreibung der Struktur ***RecommendationStructure***.....106
Tabelle 136: Beschreibung der Struktur ***ConnectionDemandRequestStructure***. 106
Tabelle 137: Beschreibung der Struktur ***ConnectionDemandDeleteRequestStructure***.....107
Tabelle 138: Beschreibung der Struktur ***ConnectionDemandResponseStructure.***107
Tabelle 139: Liste der Fehlerzustände in ***ConnectionDemandResponse***. 107
Tabelle 140: Beschreibung der Struktur ***ConnectionDemandDeleteResponseStructure.***.....107
Tabelle 141: Liste der Fehlerzustände in ConnectionDemandDeleteResponse. 107
Tabelle 142: Beschreibung der Struktur ***ConnectionStatusRequestStructure*** 108
Tabelle 143: Beschreibung der Struktur ***ConnectionStatusNotificationStructure***.....108
Tabelle 144: Beschreibung der Struktur ***ConnectionStatusResponseStructure*** 109
Tabelle 145: Liste der Fehlerzustände in ConnectionStatusResponseStructure.....109

Tabelle 146: Beschreibung der Struktur ***ConnectionReportRequestStructure***.....109
Tabelle 147: Beschreibung der Struktur ***ConnectionReportResponseStructure***.....110
Tabelle 148: Liste der Fehlerzustände in ConnectionReportResponse. 110
Tabelle 149: Beschreibung der Struktur ***FaresRequestStructure***. 110
Tabelle 150: Beschreibung der Struktur ***StopFaresRequestStructure***. 111
Tabelle 151: Beschreibung der Struktur ***StaticFaresRequestStructure***.....111
Tabelle 152: Beschreibung der Struktur ***TripFaresRequestStructure***. 111
Tabelle 153: Beschreibung der Struktur ***MultiTripFaresRequestStructure***. 111
Tabelle 154: Beschreibung der Struktur ***FaresResponseStructure***. 112
Tabelle 155: Liste der Fehlerzustände in ***FaresResponse***. 112
Tabelle 156: Beschreibung der Struktur ***FaresResultStructure***. 112
Tabelle 157: Beschreibung der Struktur ***StopFaresResultStructure***. 112
Tabelle 158: Beschreibung der Struktur ***StaticFaresResultStructure***.....113
Tabelle 159: Beschreibung der Struktur ***TripTicketReferenceStructure***.....113
Tabelle 160: Beschreibung der Struktur ***MultiTripFaresResultStructure.*** 113
Tabelle 161: Liste der Fehlerzustände in ***MultiTripFaresResultStructure***.....114
Tabelle 162: Beschreibung der Struktur ***BookingInfoRequestStructure***. 115
Tabelle 163: Beschreibung der Struktur ***BookingInfoResponseStructure***. 115
Tabelle 164: Liste der Fehlerzustände in ***BookingInfoResponse***. 116
Tabelle 165: Beschreibung der Struktur ***BookingInfoResultStructure***. 116
Tabelle 166: Liste der einfachen Typdefinitionen in Trias_IndividualTrips.xsd.....117
Tabelle 167: Beschreibung der Struktur ***IndividualRouteRequestStructure***. 117
Tabelle 168: Beschreibung der Struktur ***IndividualTripParamStructure***.....118
Tabelle 169: Beschreibung der Struktur ***IndividualRouteContextStructure***. 119
Tabelle 170: Beschreibung der Struktur ***IndividualRouteResponseStructure***. 119
Tabelle 171: Liste der Fehlerzustände in ***IndividualRouteResponse***. 120
Tabelle 172: Beschreibung der Struktur ***RouteResultStructure***.....120
Tabelle 173: Liste der Fehlerzustände in ***RouteResult***.....120
Tabelle 174: Beschreibung der Struktur ***RouteStructure*** 121
Tabelle 175: Liste der einfachen Typdefinitionen in Trias_Maps.xsd.....122
Tabelle 176: Beschreibung der Struktur ***MapServiceRequestStructure***. 122
Tabelle 177: Beschreibung der Struktur ***ImageCoordinatesRequestStructure***.....123
Tabelle 178: Beschreibung der Struktur ***GeoCoordinatesRequestStructure***. 123
Tabelle 179: Beschreibung der Struktur ***MapAspectStructure***.....123
Tabelle 180: Beschreibung der Struktur ***MapSizeStructure***. 124

Tabelle 181: Beschreibung der Struktur ***MapCoordinateStructure***. 124
Tabelle 182: Beschreibung der Struktur ***MapServiceResponseStructure***. 124
Tabelle 183: Liste der Fehlerzustände in ***MapServiceResponse***. 124
Tabelle 184: Beschreibung der Struktur ***ImageCoordinatesResponseStructure***. 125
Tabelle 185: Liste der Fehlerzustände in ***ImageCoordinatesResponse***. 125
Tabelle 186: Beschreibung der Struktur ***GeoCoordinatesResponseStructure***. 125
Tabelle 187: Liste der Fehlerzustände in ***GeoCoordinatesResponse***. 125
Tabelle 188: Beschreibung der Struktur ***MapResultStructure***. 126
Tabelle 189: Beschreibung der Struktur ***ImagePointResultStructure***. 126
Tabelle 190: Liste der Fehlerzustände in ***ImagePointResultStructure***. 126
Tabelle 191: Beschreibung der Struktur ***GeoCoordinateResultStructure***. 127
Tabelle 192: Liste der Fehlerzustände in ***GeoCoordinateResultStructure***. 127
Tabelle 193: Beschreibung der Struktur einfacher Typen 128
Tabelle 194: Beschreibung der Struktur ***FacilityStructure***. 128
Tabelle 195: Beschreibung der Struktur ***VehicleFacilityStructure***. 128
Tabelle 196: Beschreibung der Struktur ***InfrastructureFacilityStructure***. 129
Tabelle 197: Beschreibung der Gruppe ***VehicleFacilityRefGroup***. 129
Tabelle 198: Beschreibung der Gruppe ***InfrastructureFacilityRefGroup***. 129
Tabelle 199: Beschreibung der Gruppe ***FacilityDataFilterGroup***. 130
Tabelle 200: Beschreibung der Struktur ***FacilityStatusStructure***. 130
Tabelle 201: Beschreibung der Struktur ***FacilityStatusReportStructure***. 130
Tabelle 202: Beschreibung der Struktur ***FacilityStatusReportResponseStructure***. 131
Tabelle 203: Liste der Fehlerzustände in ***FacilityStatusReportResponse***. 131
Tabelle 204: Beschreibung der Struktur ***FacilityRequestStructure***. 131
Tabelle 205: Beschreibung der Struktur ***FacilityResponseStructure***. 132
Tabelle 206: Liste der Fehlerzustände in ***FacilityResponse***. 132
Tabelle 207: Beschreibung der Struktur ***FacilityResultStructure***. 132
Tabelle 208: Liste der Fehlerzustände in ***FacilityResultStructure***. 132
Tabelle 209: Beschreibung der Struktur ***TripMonitoringParamStructure***. 134
Tabelle 210: Beschreibung der Struktur ***TripMonitoringSubscriptionRequestStructure*** 135
Tabelle 211: Beschreibung der Struktur ***TripMonitoringDeliveryStructure***. 135
Tabelle 212: Liste der Fehlerzustände in ***TripMonitoringDeliveryStructure***. 136
Tabelle 213: Liste der einfachen Typdefinitionen in Trias_Personalisation.xsd. 141

Tabelle 214: Beschreibung der Struktur ***PersonalisationRequestStructure***. 141

Tabelle 215: Beschreibung der Struktur ***PersonalisationSaveValueRequestStructure***.....141

Tabelle 216: Beschreibung der Struktur ***PersonalisationRetrieveValueRequestStructure***. 141

Tabelle 217: Beschreibung der Struktur ***PersonalisationDeleteValueRequestStructure***. 142

Tabelle 218: Beschreibung der Struktur ***PersonalisationEnumerateValuesRequestStructure***.....142

Tabelle 219: Beschreibung der Struktur ***PersonalisationResponseStructure***. 142

Tabelle 220: Beschreibung der Struktur ***PersonalisationSaveValueResponseStructure***.....142

Tabelle 221: Liste der Fehlerzustände in ***PersonalisationSaveValueResponseStructure***.....143

Tabelle 222: Beschreibung der Struktur ***PersonalisationRetrieveValueResponseStructure***. 143

Tabelle 223: Liste der Fehlerzustände in ***PersonalisationRetrieveValueResponseStructure***. 143

Tabelle 224: Beschreibung der Struktur ***PersonalisationDeleteValueResponseStructure***. 143

Tabelle 225: Liste der Fehlerzustände in ***PersonalisationDeleteValueResponseStructure***. 143

Tabelle 226: Beschreibung der Struktur ***PersonalisationEnumerateValuesResponseStructure***.....143

Tabelle 227: Beschreibung der Struktur ***VehicleDataRequestStructure***.....144

Tabelle 228: Beschreibung der Struktur ***VehicleDataResponseStructure***.....144

Tabelle 229: Beschreibung der Struktur ***VehicleStatusStructure***. 144

Tabelle 230: Beschreibung der Struktur ***VehicleActivityStructure***. 145

Tabelle 231: Beschreibung der Struktur ***StopSequenceStructure***. 145

Tabelle 232: Beschreibung der Struktur ***StopInformationStructure***. 145

Tabelle 233: Beschreibung der Struktur ***DisplayContentStructure***.....146

Tabelle 234: Beschreibung der Struktur ***LineInformationStructure***.....146

Tabelle 235: Beschreibung der Struktur ***DestinationStructure*** 146

Tabelle 236: Beschreibung der Struktur ***ViaPointStructure*** 146

Tabelle 237: Beschreibung der Struktur ***AnnouncementStructure***.....147

Tabelle 238: Beschreibung der Struktur ***ConnectionStructure***.....147

Tabelle 239: Beschreibung der Struktur ***VehicleTypeStructure***.....147

Tabelle 240: Beschreibung der Struktur ***StopRequestRequestStructure*** 148

Tabelle 241: Beschreibung der Struktur ***StopRequestResponseStructure*** 148

Tabelle 242: Beschreibung von einfachen Typen 149

Tabelle 243: Beschreibung der Struktur ***ServiceRegisterRequestStructure*** 149

Tabelle 244: Beschreibung der Struktur ***ServiceRegisterParamStructure*** 150

Tabelle 245: Beschreibung der Struktur ***ServiceRegisterRegisterRequestStructure*** 150

Tabelle 246: Beschreibung der Struktur ***ServiceRegisterUpdateRequestStructure*** 150

Tabelle 247: Beschreibung der Struktur ***ServiceRegisterLookupRequestStructure*** 151

Tabelle 248: Beschreibung der Struktur ***ServiceRegisterUnregisterRequestStructure*** 151

Tabelle 249: Beschreibung der Struktur ***ServiceRegisterResponseStructure*** 151

Tabelle 250: Liste der Fehlerzustände in ***ServiceRegisterResponse*** 151

Tabelle 251: Beschreibung der Struktur ***ServiceRegisterResultStructure*** 152

Tabelle 252: Beschreibung der Struktur ***ServiceRegisterRegisterResponseStructure*** 152

Tabelle 253: Beschreibung der Struktur ***ServiceRegisterUpdateResponseStructure*** 152

Tabelle 254: Beschreibung der Struktur ***ServiceRegisterLookupResponseStructure*** 152

Tabelle 255: Beschreibung der Struktur ***ServiceRegisterUnregisterResponseStructure*** 152

Tabelle 256: Beschreibung der Struktur ***TriasServiceStructure*** 153

Tabelle 257: Liste der Abkürzungen 159

31. Literaturverzeichnis

CEN, EN 12896:2006. (2006). *Reference Data Model for Public Transport.* CEN - Europäisches Komitee für Normung.

CEN, EN 28701:2012. (2012). *Intelligent transport systems - Public transport - Identification of Fixed Objects in Public Transport (IFOPT, EN 28701:2012).* CEN - Europäisches Komitee für Normung.

CEN, TS 15531 Part 1. (kein Datum). *SIRI - Service Interface for Realtime Information, Part 1.* CEN - Europäisches Komitee für Normung.

CEN, TS 15531 Part 5. (kein Datum). *SIRI - Service Interface for Realtime Information, Part 5.* CEN - Europäisches Komitee für Normung.

CEN, TS 15531, Part 2. (kein Datum). *SIRI - Service Interface for Realtime Information, Part 2.* CEN - Europäisches Komitee für Normung.

Department for Transport. (19. 04 2012). *Dft - Journeyweb*. Abgerufen am 15. 05 2013 von http://www.dft.gov.uk/journeyweb/

ISO 8601:2004. (2004). *ISO 8601:2004, Data elements and interchange formats - Information interchange - Representation of dates and times.* ISO, International Organization for Standardization.

VDV-Schrift 430. (kein Datum). *Kundenschnittstelle - Architektur.* Köln: VDV - Die Verkehrsunternehmen.

VDV-Schrift 431-1. (kein Datum). *Systemarchitektur EKAP.* Köln: VDV - Die Verkehrsunternehmen.

Impressum

Verband Deutscher Verkehrsunternehmen e. V. (VDV)
Kamekestraße 37-39 · 50672 Köln
T 0221 57979-0 · F 0221 57979-8000
info@vdv.de · www.vdv.de

Ansprechpartner

Dipl.-Ing. Berthold Radermacher
T 0221 57979-141
F 0221 57979-8141
radermacher@vdv.de

Verband Deutscher Verkehrsunternehmen e. V. (VDV)
Kamekestraße 37-39 · 50672 Köln
T 0221 57979-0 · F 0221 57979-8000
info@vdv.de · www.vdv.de