1 Beschreibung der TRIAS-Dienste und Datenstrukturen

1.1 Gemeinsam genutzte Elemente

In diesem Kapitel werden zunächst die XML-Strukturen vorgestellt und erläutert, die dienstübergreifend verwendet werden, die also nicht anfrage- oder dienstspezifisch sind. Dabei werden nur jene Strukturen erläutert, die in den jeweiligen Diensten und übergreifend unterstützt werden. Im Anschluss werden die drei Dienste, die über die EFAMiddleware abgefragt werden können, beschrieben und in ihrer Struktur dargestellt. TRIAS-Elemente, die von EFA nicht unterstützt werden, sind in den Schema-Darstellungen der folgenden Kapitel eigens gekennzeichnet.

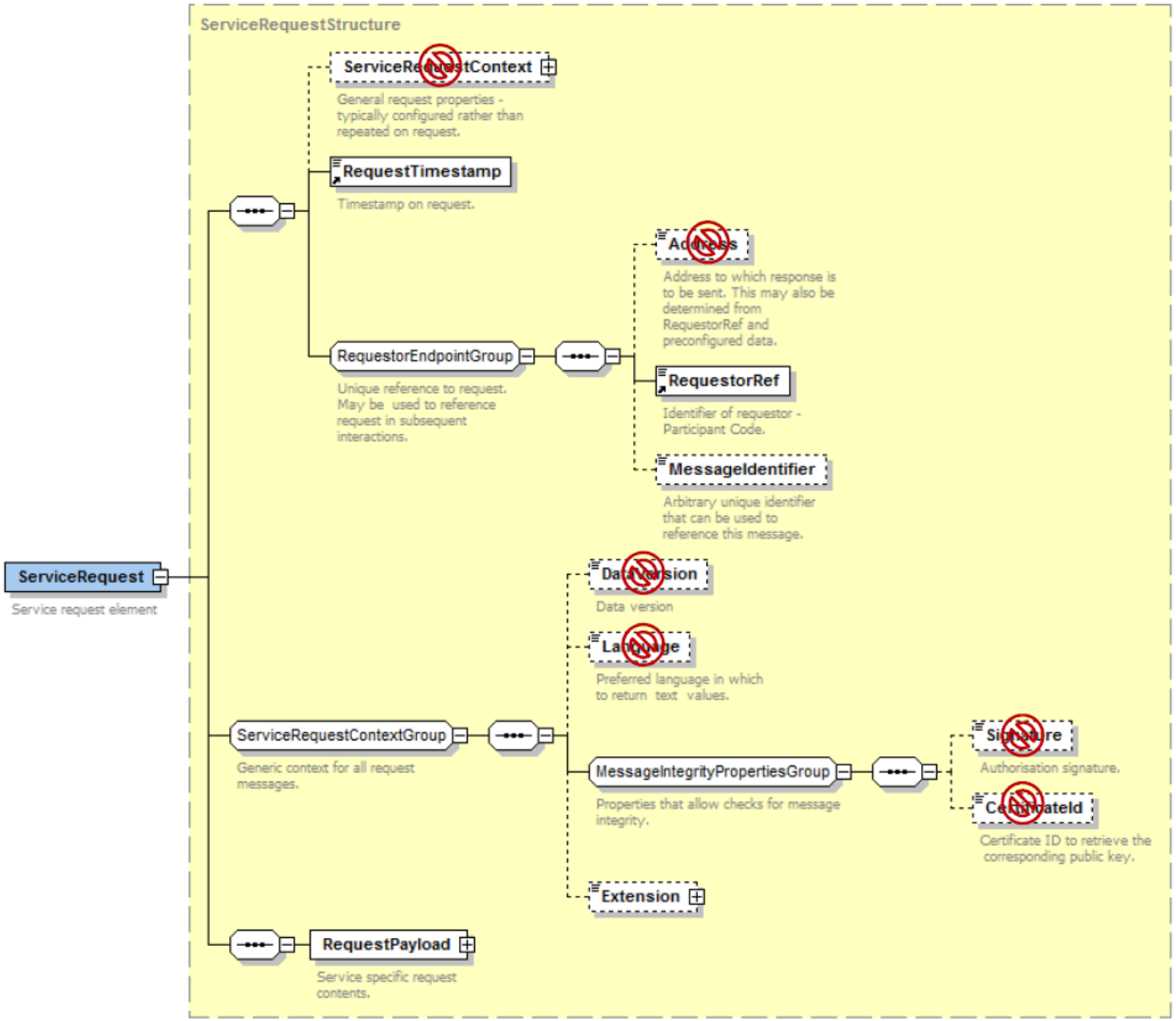
Die von TRIAS insgesamt zur Verfügung gestellten und von EFA unterstützen Anfrage- und Antwortstrukturen gehen aus nachfolgender Übersicht der *RequestPayloadStructure* (Anfragestrukturen) und *DeliveryPayloadStructure* (Antwortstrukturen) hervor. Die rot eingerahmten Dienste werden von der EFAMiddleware unterstützt. Die vollständige Dokumentation aller TRIAS-Dienste und ihrer Anfrage- bzw. Antwortstrukturen steht auf der Homepage des VDV zum Download zur Verfügung.

TRIAS-Anfragen bestehen aus XML-Dokumenten. Jedes XML-Dokument beginnt mit der XML-Deklaration, die eine Erkennungszeichenfolge im Prolog einer XML-Datei ist. Sie hat folgende Form:

<?xml version="1.0" encoding="UTF-8"?>

- *version="1.0"* definiert die Versionsnummer der zugrundeliegenden XML-Spezifikation. Die Angabe wird empfohlen, ist aber nicht zwingend erforderlich. Beim Weglassen wird von version="1.0" ausgegangen.
- *encoding="UTF-8"* bestimmt die Kodierung der XML-Datei. Wird dieser Parameter ausgelassen, wird UTF-8 angenommen.

Nach der XML-Deklaration folgt das sogenannte Wurzelelement, mit dem das Dokument beginnt und mit dem es endet. Bei TRIAS-Anfragen wird dafür das Element *Trias* verwendet, das als gemeinsame Basis für alle Nachrichten aller TRIAS-Dienste steht. Das gilt für die Anfrage gleichermaßen wie für die Antworten. Direkt im Anschluss wird bei Anfragen das Element *ServiceRequest* erwartet, das die Anfragegrundstruktur darstellt. Man spricht hier auch von der *ServiceRequestStructure*. Das Inhaltsmodell dieser Struktur wird anhand der folgenden Darstellungen erklärt.



Unmittelbar nach dem *ServiceRequest* kommen folgende Elemente:

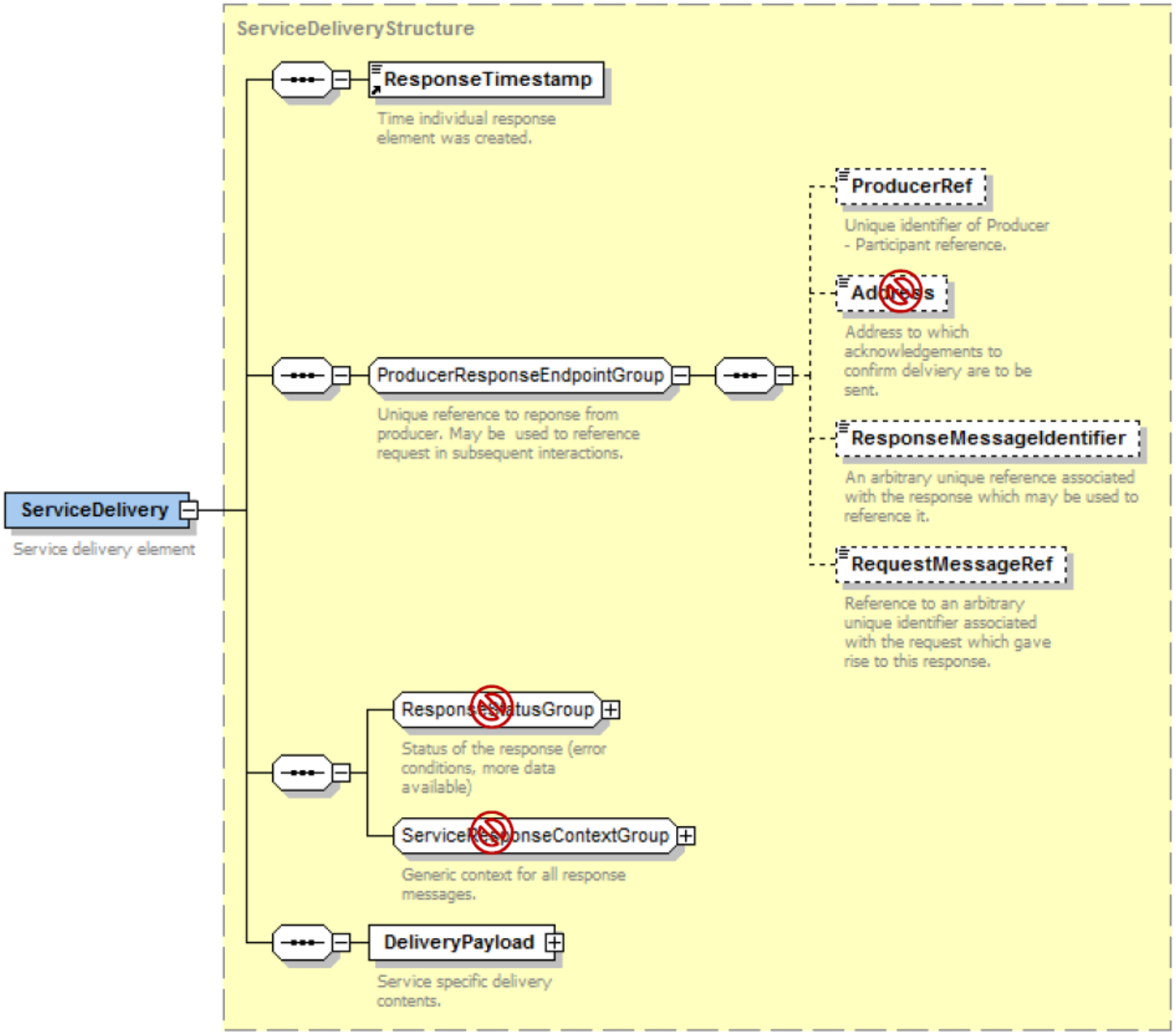
- *RequestTimestamp*: enthält den Zeitstempel der Anfrage
- *RequestorRef*: dient zur Identifizierung des anfragenden Dienstes und wird gegen die Einträge in der Datei requestor-refs.xml geprüft
- *RequestPayload*: Container für die eigentliche Anfrage
- *MessageIdentifier*: nicht obligatorisches Element, das – wenn es bei der Anfrage verwendet wird – eine eindeutige ID enthält, mit der die Anfrage referenziert werden kann

Die Antwortstruktur (*ServiceDeliveryStructure*) beginnt – nach der XML-Deklaration und dem *Trias*-Element – mit *ServiceDelivery* und hat folgenden Inhalt:

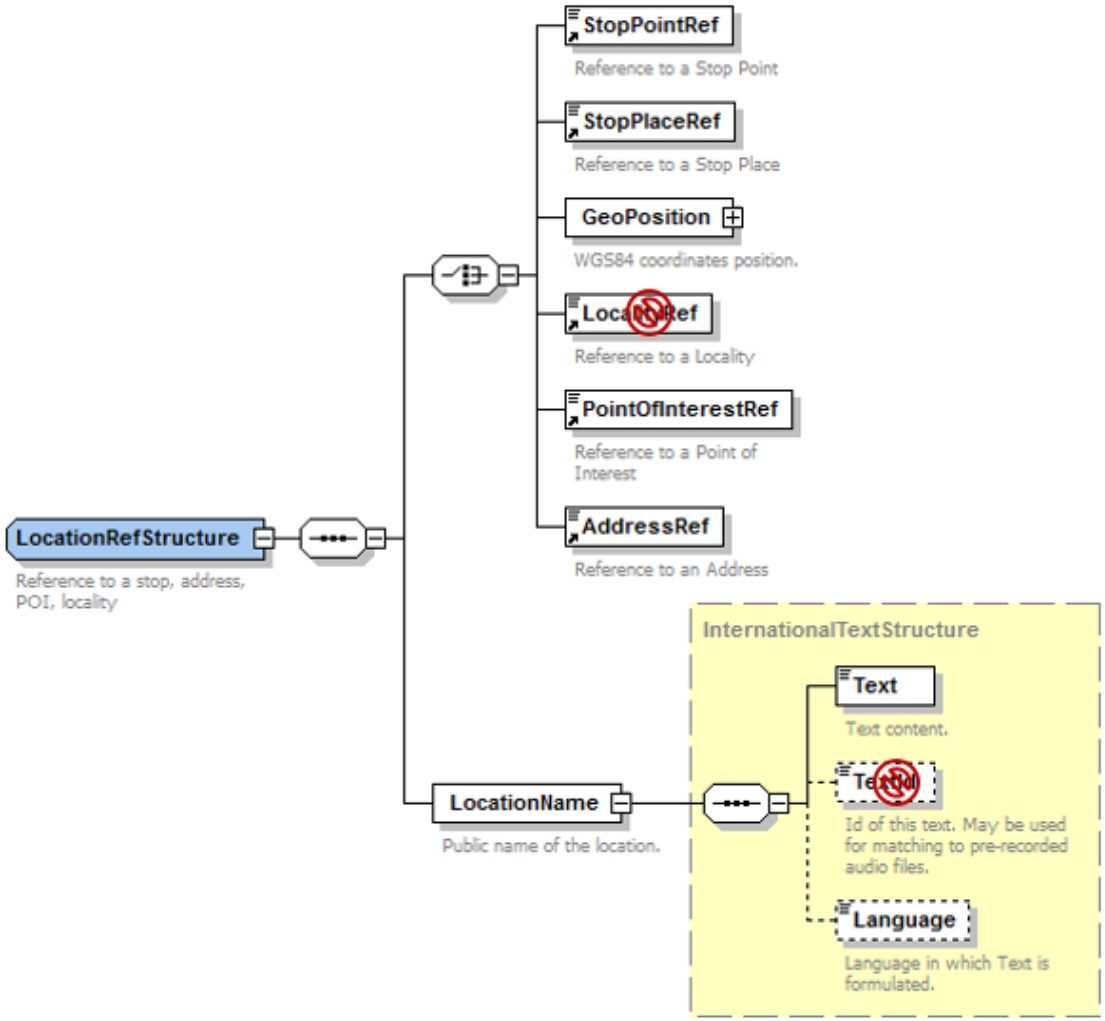
- *ResponseTimestamp*: Zeitpunkt, zu dem die XML-Antwort erstellt wurde
- *ProducerRef*: Wert der in der Mdv.Efa.Trias.dll.config festgelegten Einstellung

```
<trias producerRef="VVO" defaultEndpoint="v10">
```

- *ResponseMessageIdentifier*: automatisch erzeugte ID zur Identifikation der Antwort; ist gleichzeitig der Dateiname der erzeugten und im *Request*-Verzeichnis abgelegten Log-Datei
- *RequestMessageRef*: Enthält den Inhalt von *MessageIdentifier* aus der Anfrage
- *DeliveryPayload*: Enthält die eigentliche Antwort

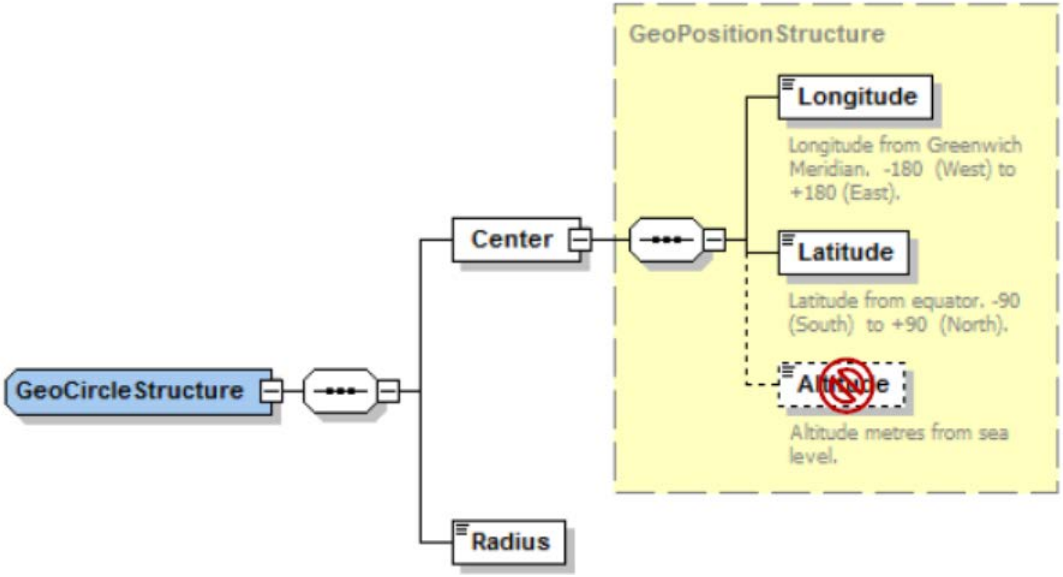


Die *LocationRefStructure* stellt die Referenz auf einen allgemeinen Ortspunkt dar. Es kann sich dabei um eine Haltestelle, einen Haltepunkt, eine Koordinatenposition, einen POI oder eine Ortschaft handeln. Sie wird wie folgt definiert:

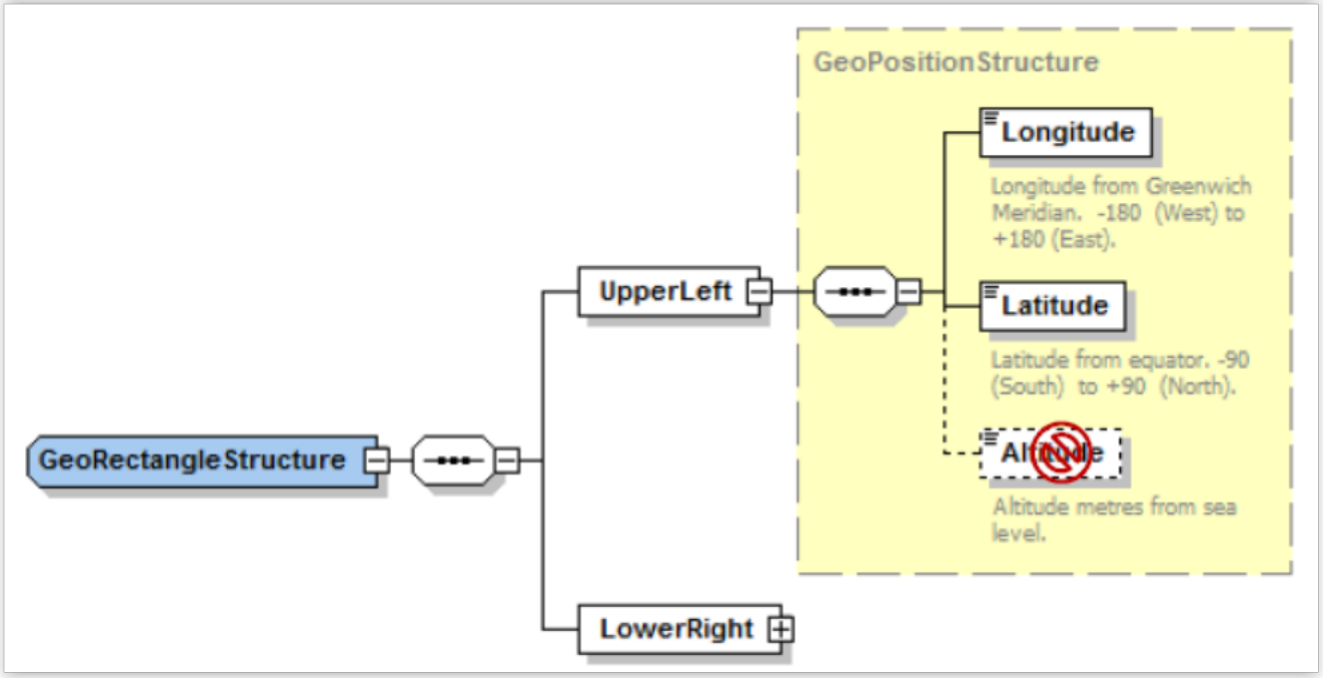


- *StopPlaceRef*[1]: ID der Haltestelle (Gobale ID)
- *GeoPosition*: Ortskoordinate (*Longitude* und *Latitude*)
- *PointOfInterestRef*: ID eines wichtigen Punktes (POI)
- *AdressRef*: ID für das eindeutige Auflösen einer Adresse
- *LocationName*: Im Feld *Text* steht der Name der zugehörigen Gemeinde

Die *GeoCircleStructure* und die *GeoRectangleStructure* bieten die Möglichkeit, Flächenstrukturen anzufragen, wobei die *GeoCircleStructure* mit *Center* und *Radius* die Koordinate des Kreismittelpunktes sowie dessen Radius, und die *GeoRectangleStructure* mit *UpperLeft* und *UpperRight* die jeweils linke obere und rechte untere Koordinate des angefragten Rechtecks als Parameter benötigen.

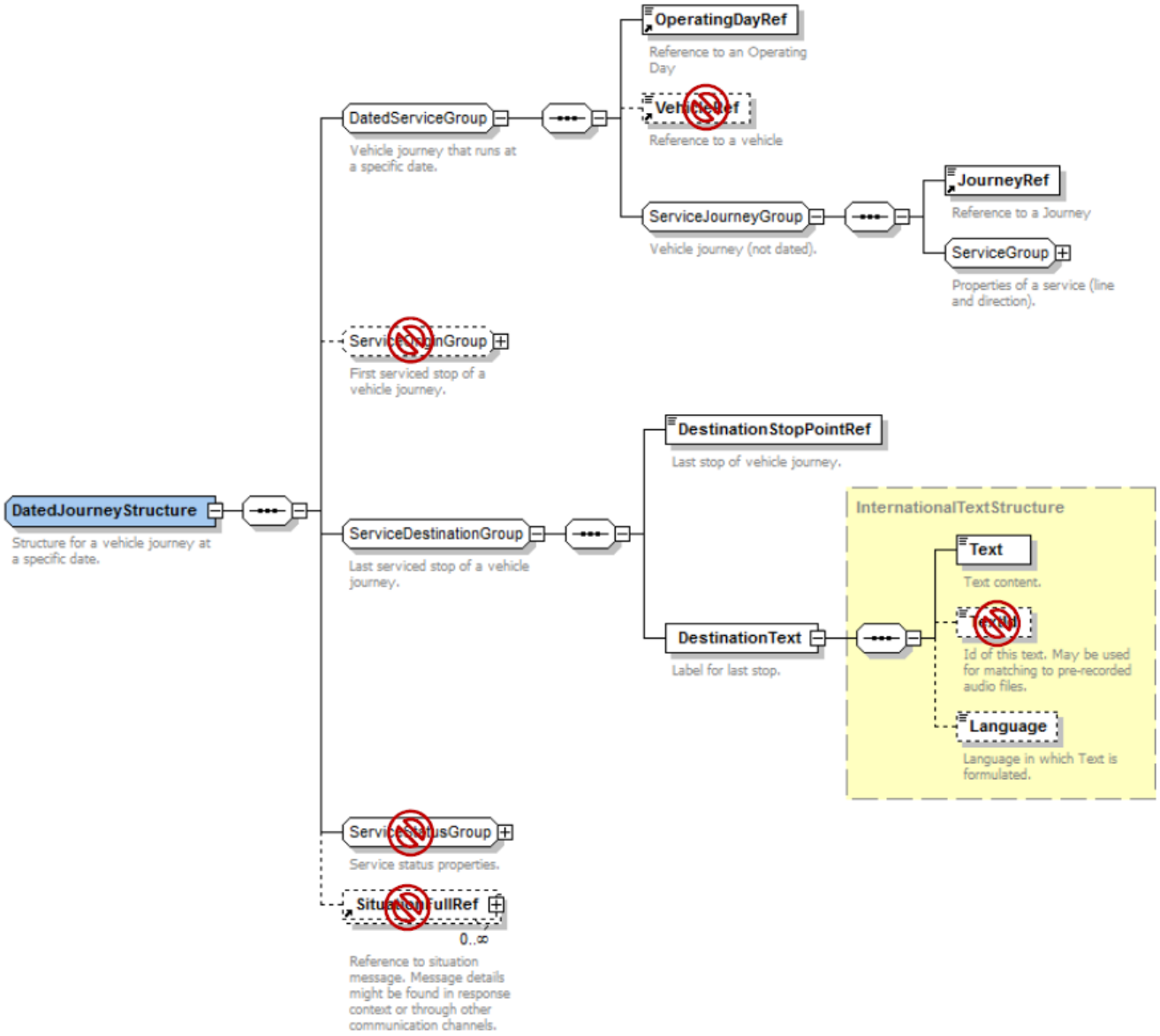


[1] Die Version 1.0 der TRIAS-Schnittstelle erlaubt die Eingabe der Haltestellen-ID auch in *StopPointRef*, das normalerweise die ID des Haltepunkts (Steig) beinhaltet.

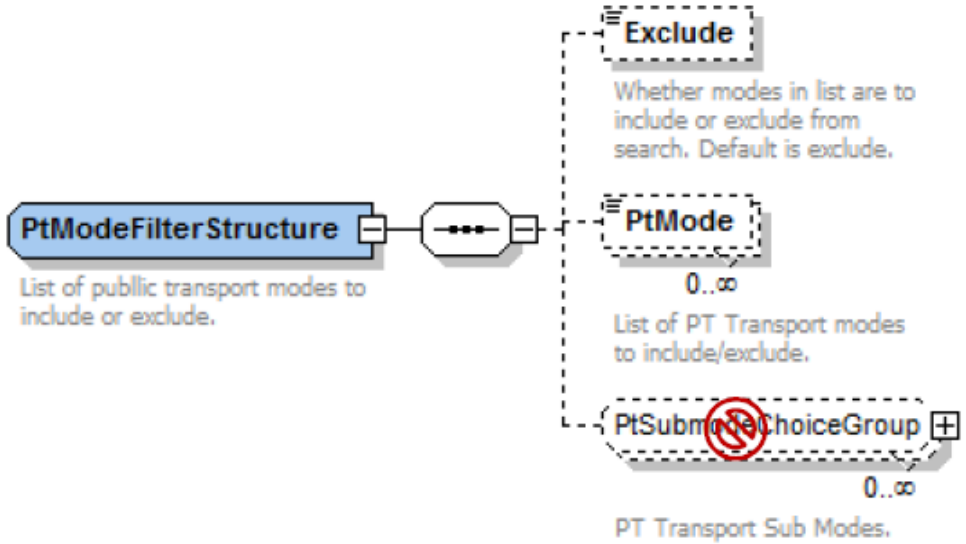


Die *DatedJourneyStucture* steht für die nähere Fahrtbeschreibung und gibt Auskunft über die Fahrplanfahrt an einem bestimmten Tag. Sie enthält folgende Elemente:

- *OperatingDayRef*: Datum und Uhrzeit der Fahrt
- *JourneyRef*: ID der Fahrt
- *DestinationStopPointRef*: ID der Zielhaltestelle
- *DestinationText*: Text, der am Verkehrsmittel als Ziel angezeigt wird



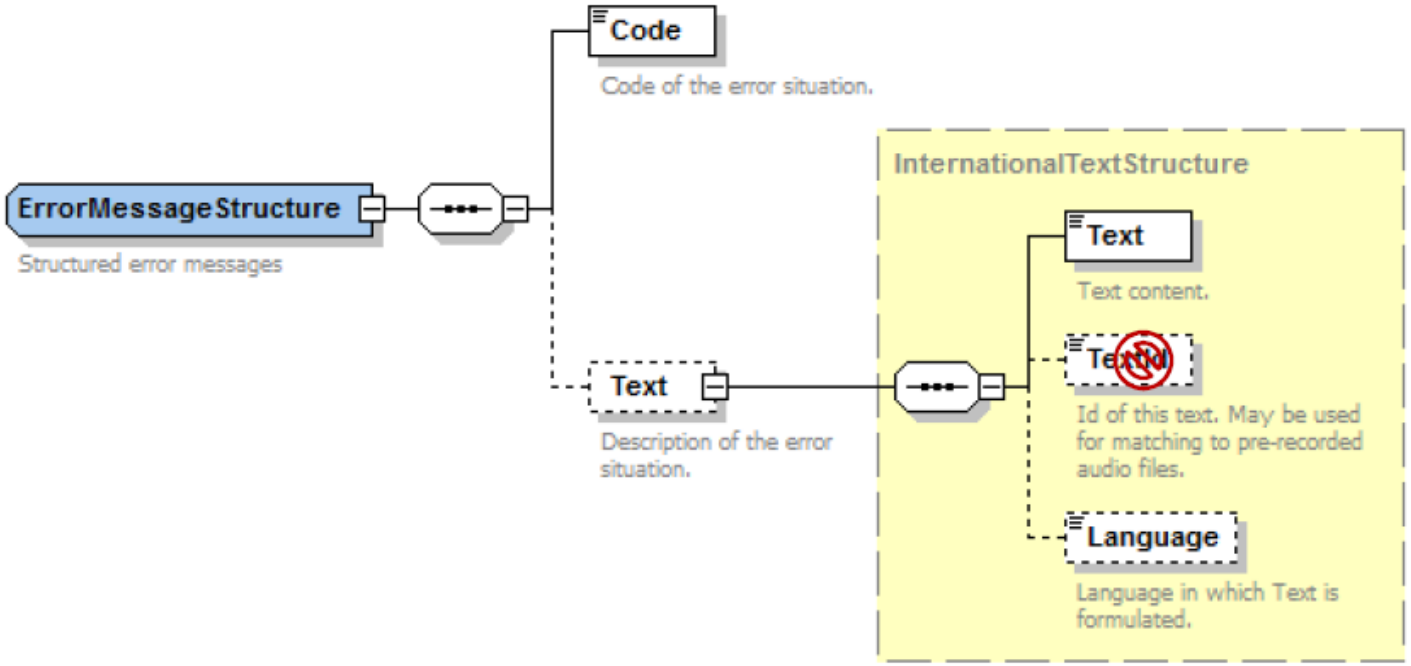
Die *PtModeFilterStructure* erlaubt das Filtern nach Verkehrsmitteltypen und hat folgendes Inhaltsmodell:



- *Exclude*: Gibt an, ob die in der Liste angegebenen Verkehrsmittel ausgeschlossen (Wert = *true*) oder als einzige verwendet werden sollten (Wert = *false*). Die Voreinstellung ist *true*
- *PtMode*: Liste der ausgeschlossenen oder verwendeten ÖV-Verkehrsmittel (*bus, tram, coach, rail, intercityRail, urbanRail, metro*)

Die *ErrorMessageStructure* spiegelt die Struktur zur Übertragung von Fehlerzuständen wieder. Die Fehlerzustände sind dienstspezifisch. Je nachdem, um was für einen Dienst es sich handelt, beginnt der Wert der Fehlermeldung mit *LOCATION_*, *TRIP_* oder *STOPEVENT_*.

Die Struktur wird wie folgt dargestellt:



1.2 Beschreibung der TRIAS-Dienste

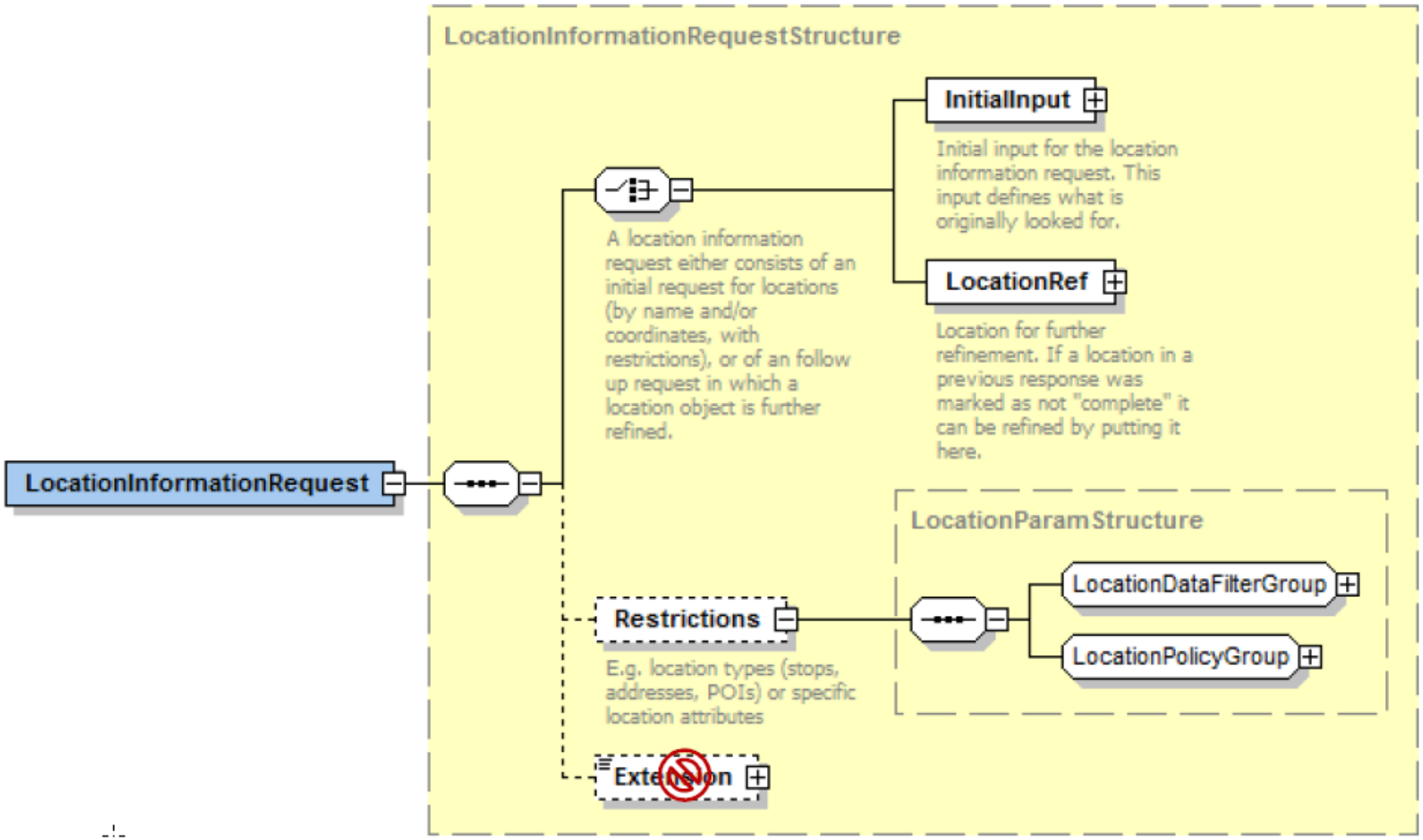
1.2.1 LocationInformationRequest (Ortsinformationsdienst) – Beschreibung

Der Ortsinformationsdienst umfasst diese Funktionalitäten:

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

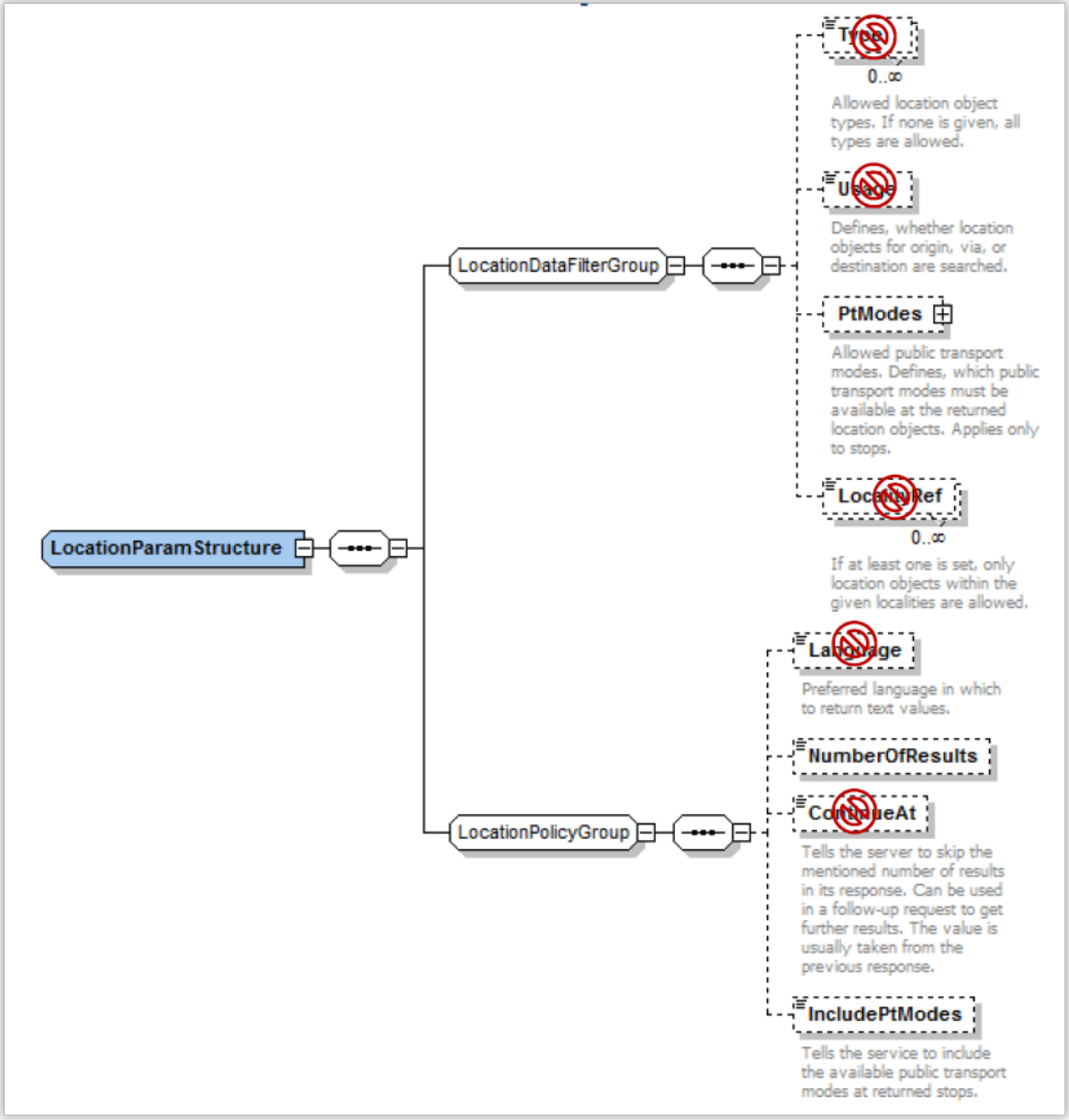
1.2.1.1 Ortsinformationsdienst – Anfragestrukturen

Ortsobjekte werden mit dem Element *LocationInformationeRequest* angefordert. In der EFA stehen hierfür zwei unterschiedliche Requests zur Verfügung: der XML_STOPFINDER_REQUEST und der XML_COORD_REQUEST, je nachdem, ob eine Haltestelle oder ein anderes Objekt angefragt wird. Beim Ortsinformationsdienst wird zwischen zwei Anfragestrukturen unterschieden: der *InitialInputStructure*, mit der die üblichen Ortsinformationen abgefragt werden können, und der *LocationRefStructure*, die eine genauere Anfrage über bereits verifizierte Ortsinformationen zulässt.



Die Anfrage kann um Parameter ergänzt werden. Dies geschieht innerhalb der *LocationParamStructure*. EFA unterstützt folgende Parameter:

- *PtModes*: Filterung der Haltestellen nach Verkehrsmittel (*PtModeFilterStructure*)
- *NumberOfResults*: Anzahl der gewünschten Ergebnisse
- *IncludePtModes*: Anzeige der verfügbaren Verkehrsmittel an einer Haltestelle

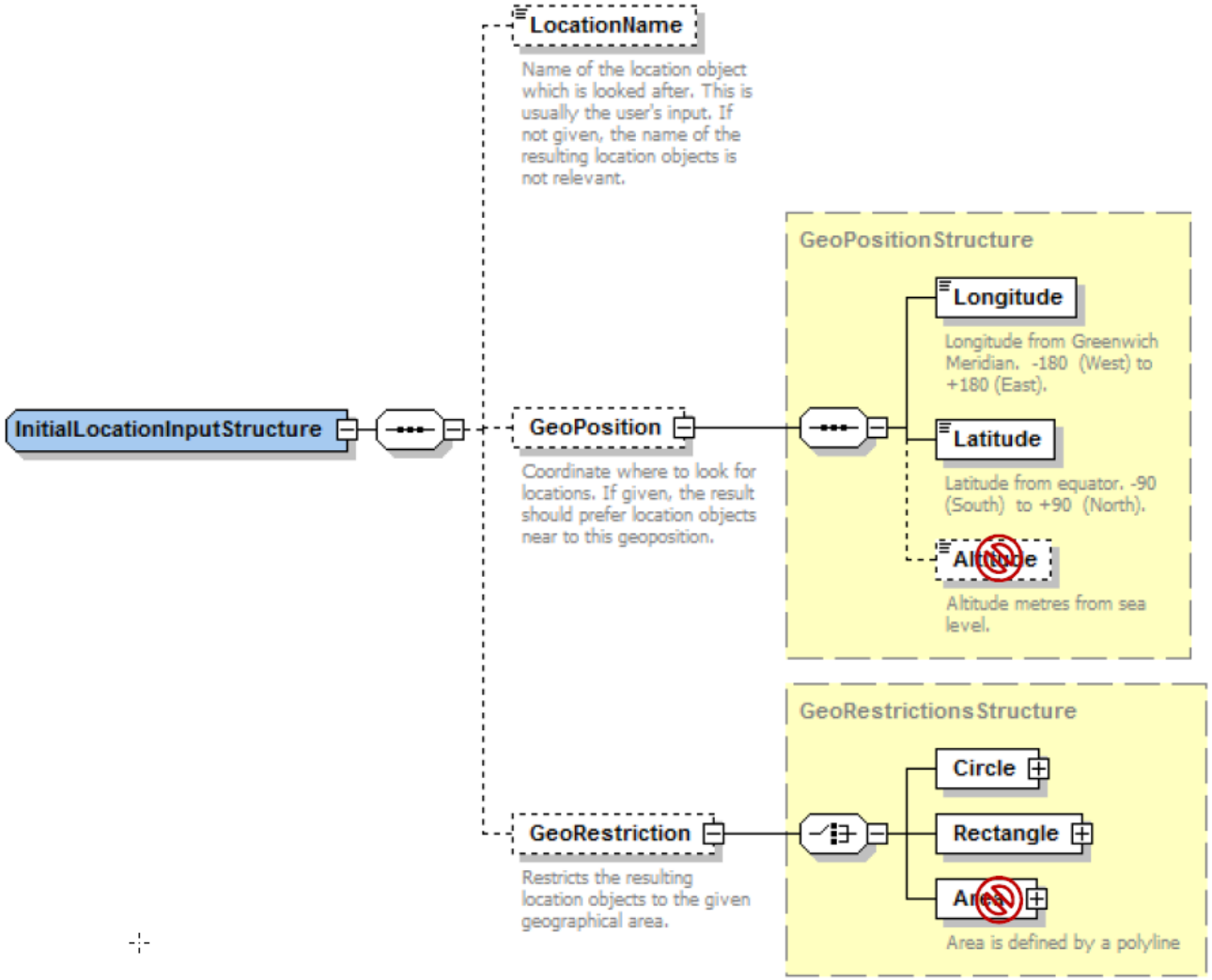


Beispiel eines *LocationInformationRequest,* mit dem „Dresden, Hauptbahnhof“ angefragt werden soll:

```
<?xml version="1.0" encoding="UTF-8"?>
<Trias xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" version="1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="trias file:///C:/Entwicklung/Data/IP-KOM/Schema/Trias.xsd">
  <ServiceRequest>
    <siri:RequestTimestamp>2016-03-30T13:36:00Z</siri:RequestTimestamp>
    <siri:RequestorRef>SEUS</siri:RequestorRef>
    <RequestPayload>
      <LocationInformationRequest>
        <InitialInput>
          <LocationName>haupt</LocationName>
        </InitialInput>
        <Restrictions>
          <Type>stop</Type>
          <Language>de</Language>
          <NumberOfResults>10</NumberOfResults>
        </Restrictions>
      </LocationInformationRequest>
    </RequestPayload>
  </ServiceRequest>
</Trias>
```

Beispiel 1: Anfrage eines *LocationInformationRequest* mit *InitialInput*

Die Anfrage erfolgt mit den Elementen *InitialInput* und *LocationName*, d.h. die Anfrage enthält die ursprüngliche, vom Anwender angegebene Eingabe, die in diesem Fall lediglich aus der Eingabe „haupt“ besteht. Im Abschnitt *Restrictions* werden der Anfrage weitere Parameter übergeben. Generell gilt, dass ein *LocationInformationRequest* in einen XML_STOPFINDER_REQUEST umgesetzt wird, außer es sind darin geografische Filter aus der *GeoRestrictionsStructure* enthalten. In diesem Fall wird ein XML_COOR_REQUEST daraus. Die schematische Darstellung der Anfragestruktur mittels der *InitialInputStructure* sieht daher wie folgt aus:



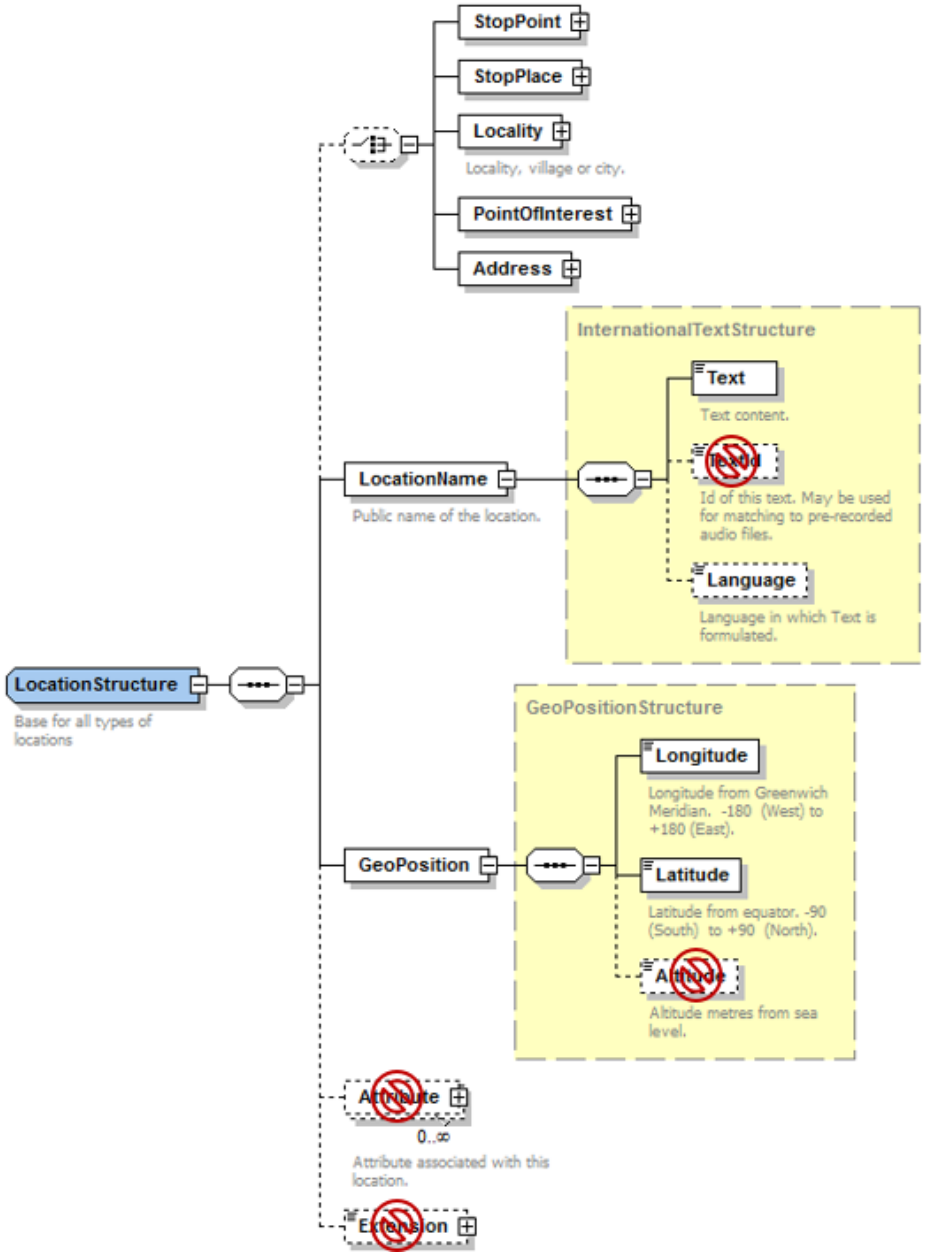
Alternativ ist die Verwendung von *LocationRef* möglich, die für bereits verifizierte Ortsinformationen angewendet wird. Dazu stehen innerhalb von *LocationRef* Substrukturen wie *StopPlace* (ID der Haltestelle) *StopPointRef* (ID des Abfahrtspunkts/Steigs), *LocalityRef* (ID der Ortschaft), *PointOfInterestRef* (ID eines POIs), *AddressRef* (ID einer Adresse), *GeoPosition* (Längen- und Breitengrad) sowie

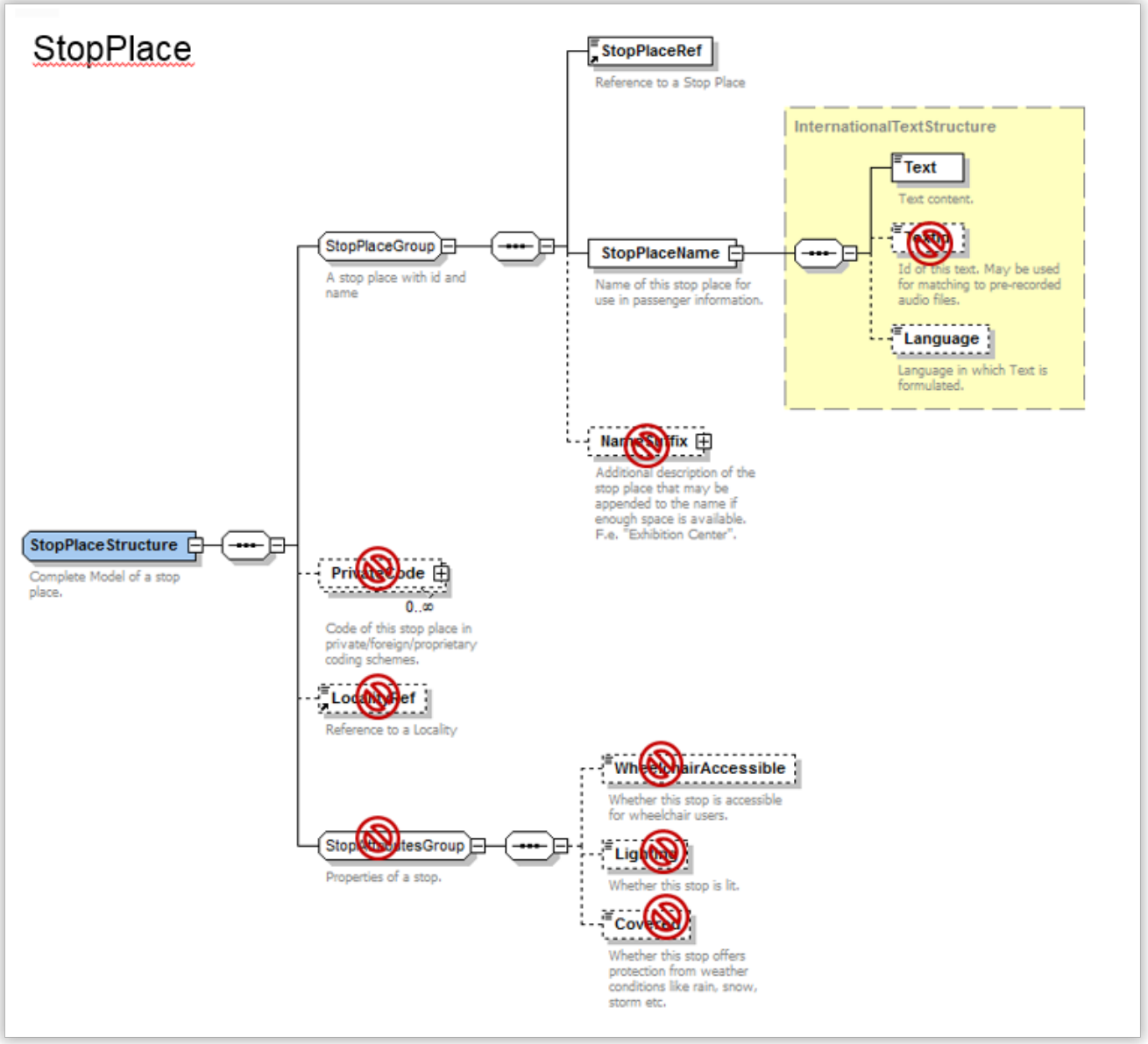
LocationName (mit *Text* und *Language*) zur Verfügung. Im Folgenden Beispiel erfolgt die Anfrage über Angabe der Globalen ID und des Haltestellenamens:

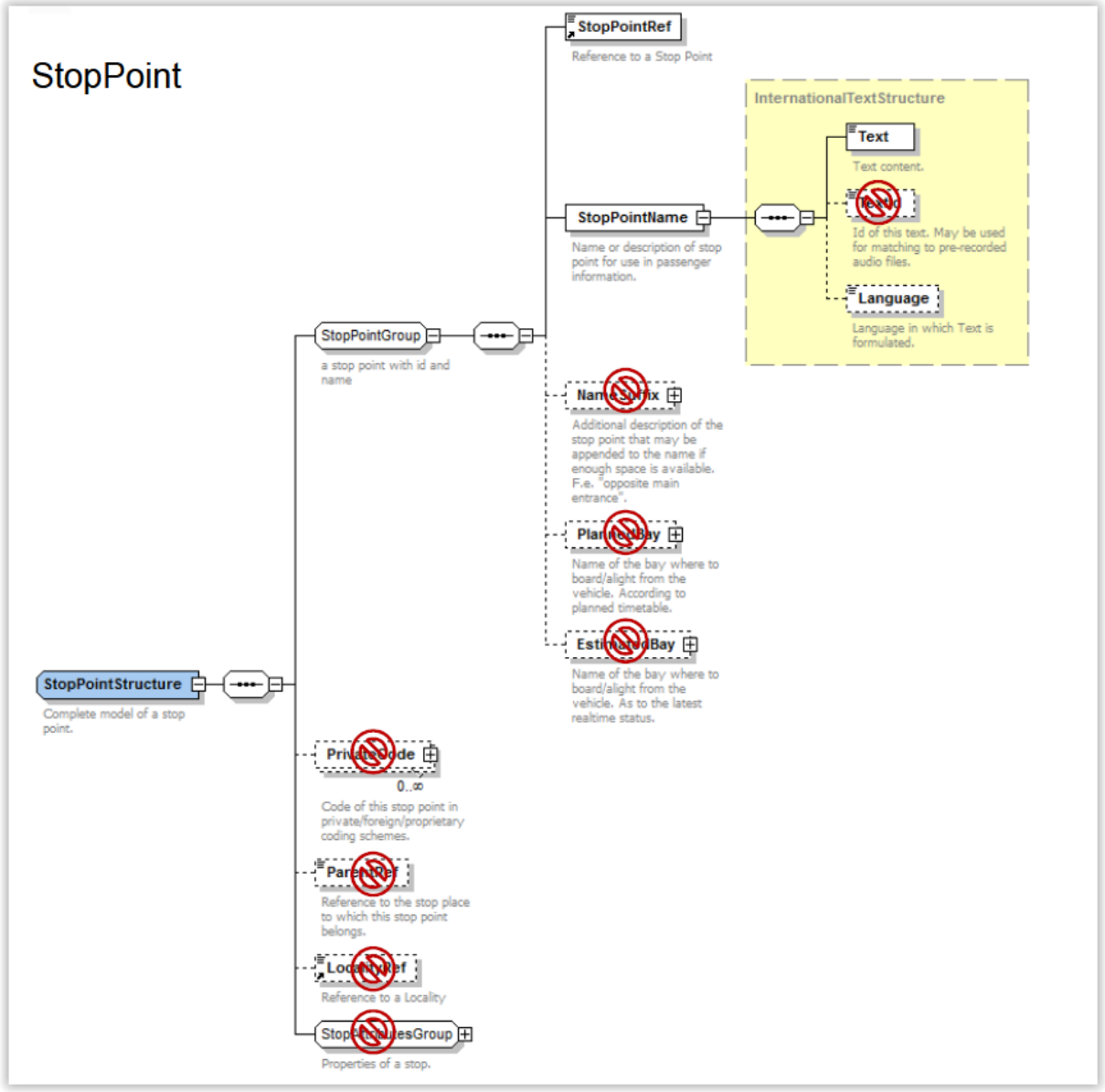
```
<?xml version="1.0" encoding="UTF-8"?>
<Trias version="1.0" xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="trias file:///C:/mentzdv/Trias/TRIASSchemas/Trias.xsd">
  <ServiceRequest>
    <siri:RequestTimestamp>2016-03-30T13:36:00</siri:RequestTimestamp>
    <siri:RequestorRef>SEUS</siri:RequestorRef>
    <RequestPayload>
      <LocationInformationRequest>
        <LocationRef>
          <StopPlaceRef>de:14612:28</StopPlaceRef>
          <StopPlaceName>
            <Text>Dresden, Hauptbahnhof</Text>
            <Language>de</Language>
          </StopPlaceName >
        </LocationRef>
        <Params />
      </LocationInformationRequest>
    </RequestPayload>
  </ServiceRequest>
</Trias>
```

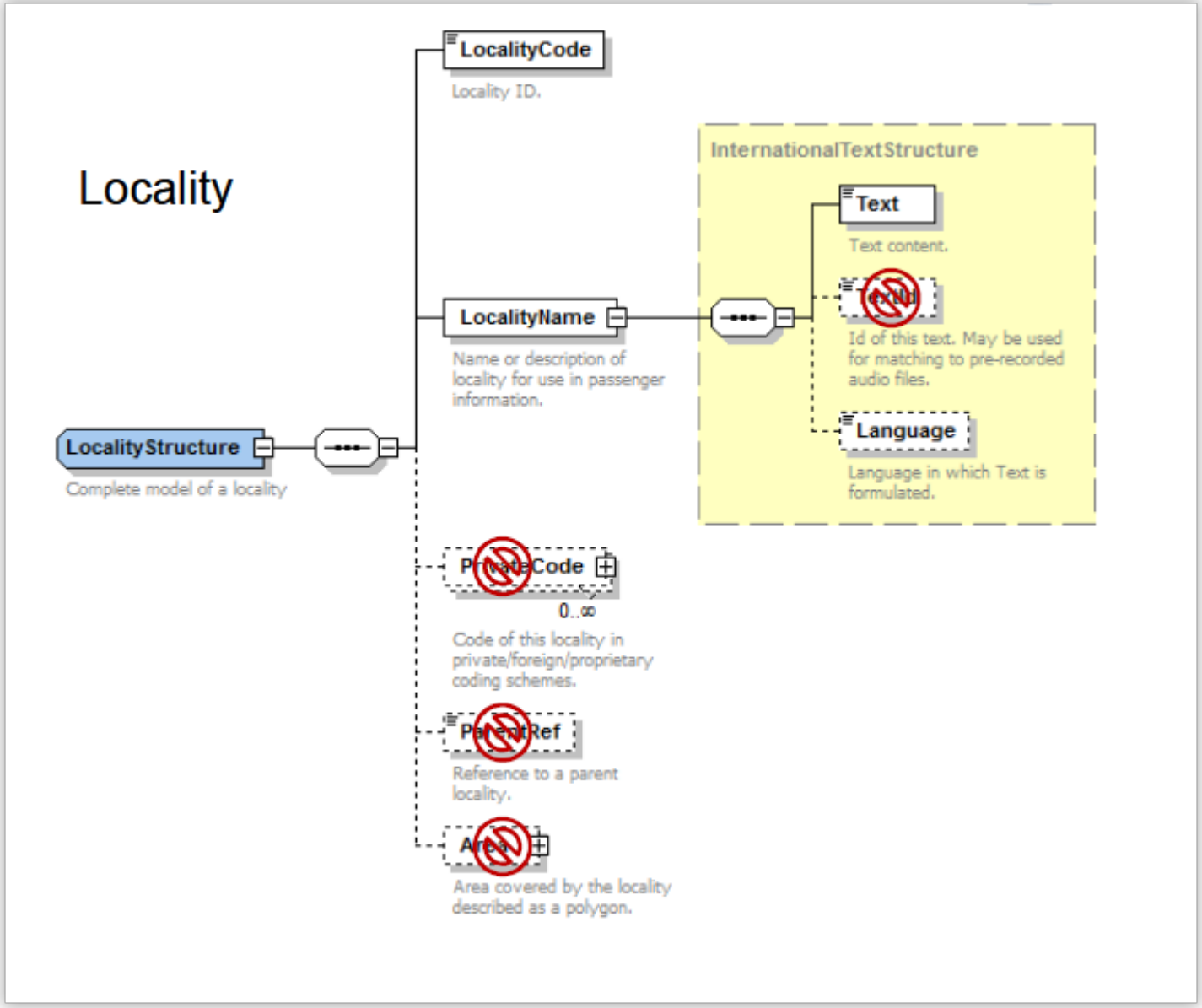
Beispiel 2: Anfrage eines *LocationInformationRequests* mit *LocationRef*

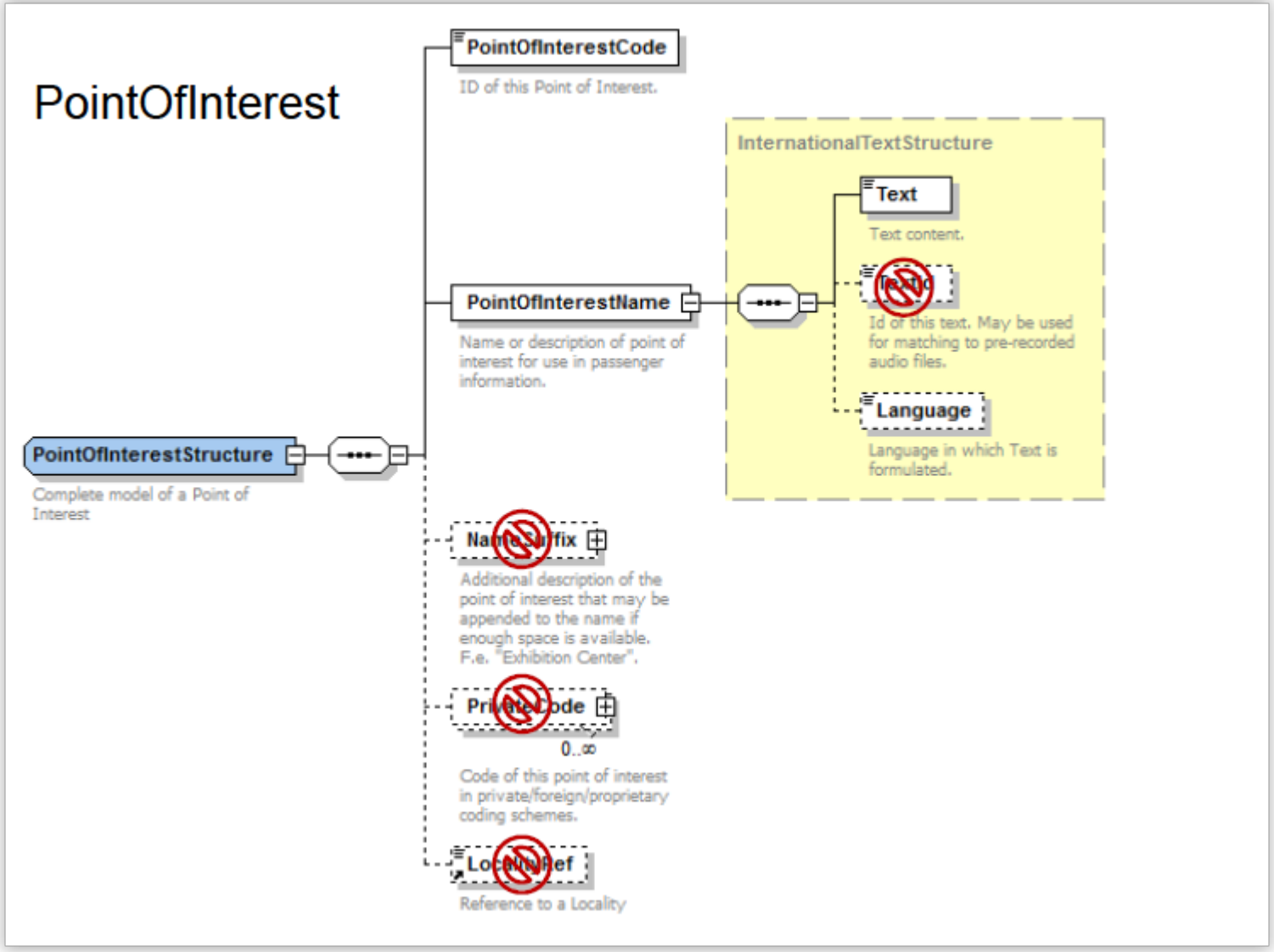
Die schematische Darstellung der *LocationRefStructure* und den untergeordneten Strukturen kann den folgenden Abbildungen entnommen werden. Neben der Suche einer Haltestelle (*StopPlace*) oder einem Steig (StopPoint) können auch Orte (*Locality*), Adressen (*Adress*) und POIs (*PointOfInterest*), Gemeindenamen (*LocationName*) und Ortskoordinaten (*GeoPosition*) mit der *LocationRefStructure* angefragt werden.

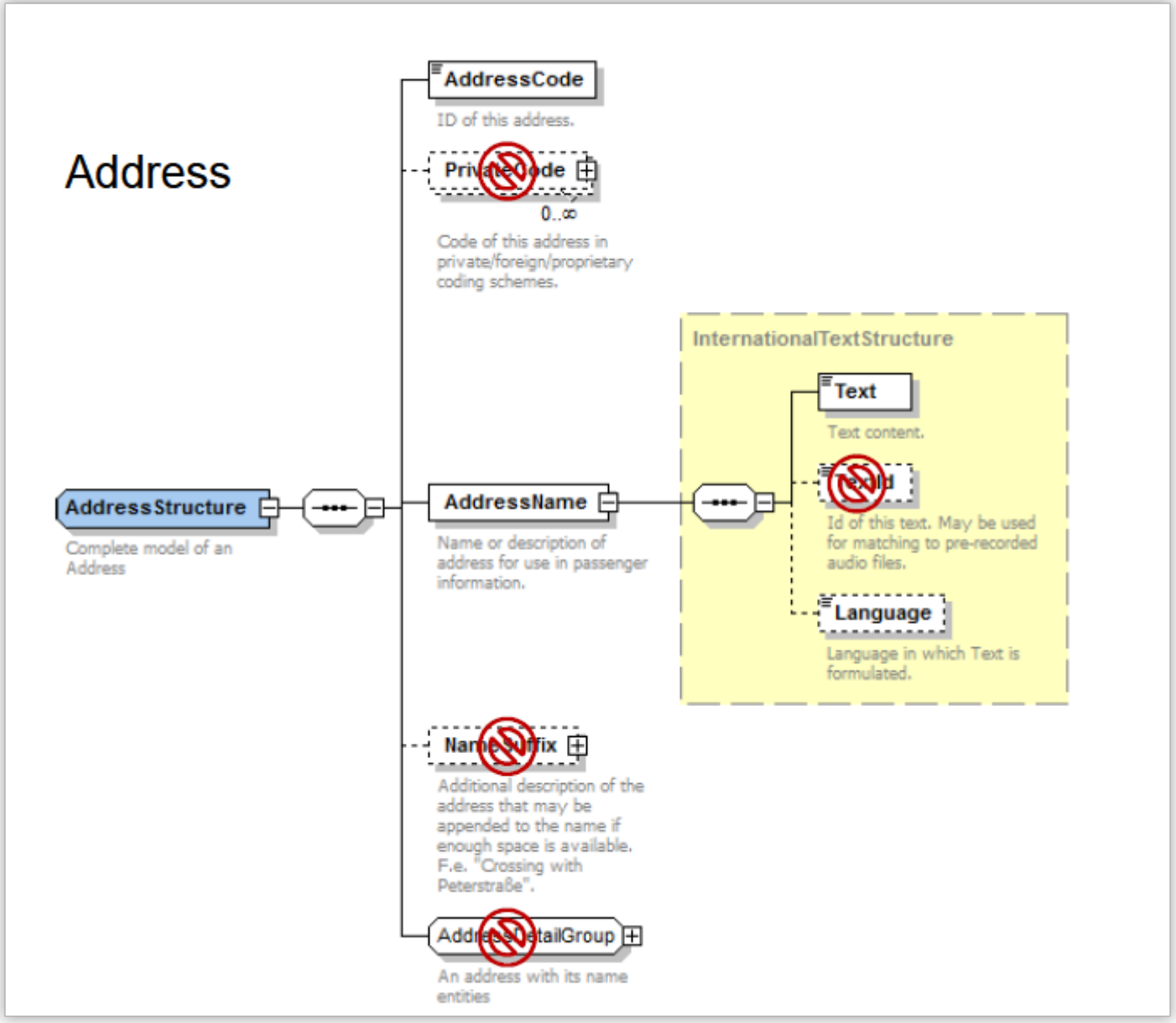












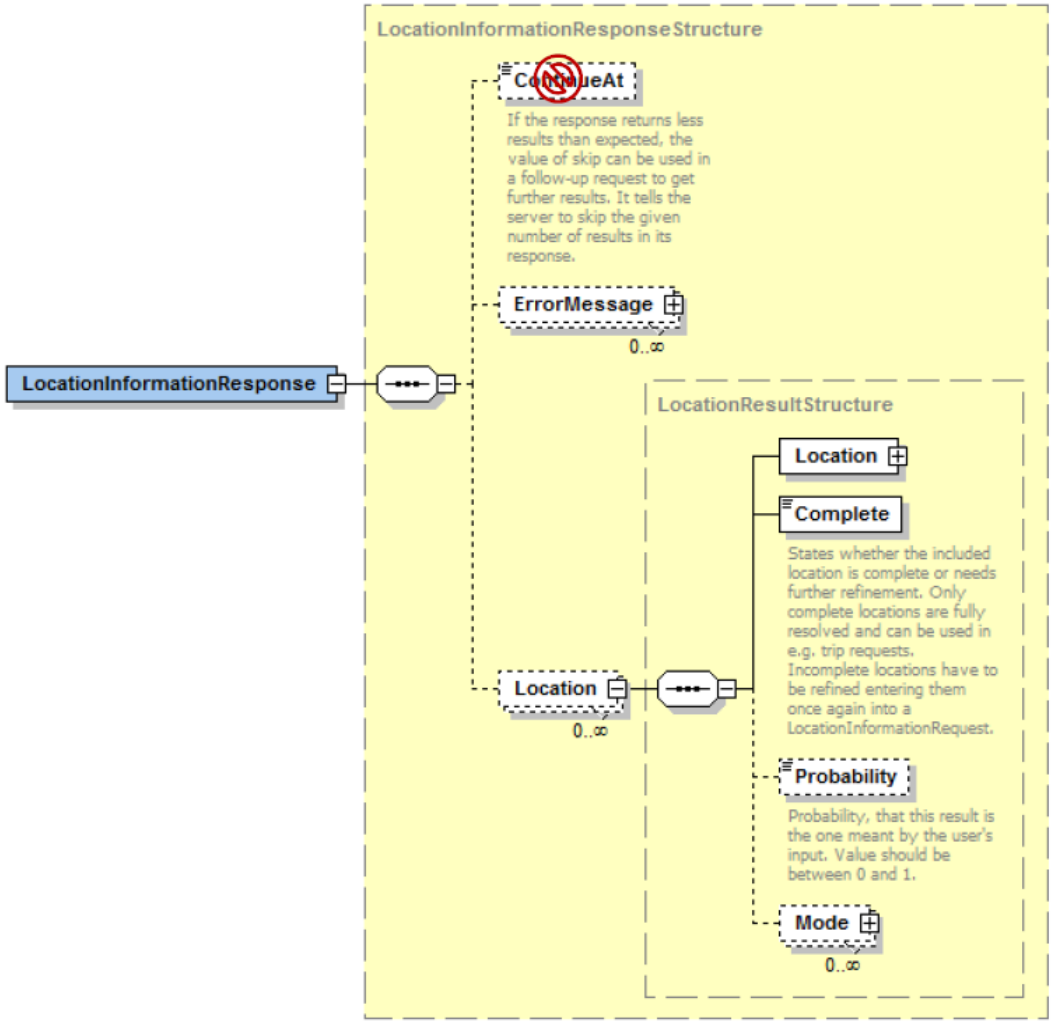
1.2.1.2 Ortsinformationsdienst – Antwortstrukturen

Das Ergebnis einer Objektinformationsanfrage wird im Element *LocationInformationResponse* übertragen. Die Antwort beider Anfragen von oben ist identisch und sieht so aus:

```
<?xml version="1.0" encoding="utf-8"?>
<Trias version="1.0" xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" xmlns:mw="http://services.mentzdv.de/2010/07/middleware">
  <ServiceDelivery>
    <siri:ResponseTimestamp>2016-03-30T16:11:00</siri:ResponseTimestamp>
    <siri:ProducerRef>VVO</siri:ProducerRef>
    <siri:ResponseMessageIdentifier>16-11-00-012</siri:ResponseMessageIdentifier>
    <DeliveryPayload>
      <LocationInformationResponse>
        <Location>
          <Location>
            <StopPlace>
              <StopPlaceRef>de:14612:28</StopPlaceRef >
              < StopPlaceName >
                <Text>Dresden, Hauptbahnhof</Text>
                <Language>germ</Language>
              </StopPlaceName>
              <LocalityRef>germ:14612000:1</LocalityRef>
            </StopPlace >
            <LocationName>
              <Text>Dresden</Text>
              <Language>germ</Language>
            </LocationName>
            <GeoPosition>
              <Longitude>13.73218</Longitude>
              <Latitude>51.04020</Latitude>
            </GeoPosition>
          </Location>
          <Complete>true</Complete>
        </Location>
      </LocationInformationResponse>
    </DeliveryPayload>
  </ServiceDelivery>
</Trias>
```

Beispiel 3: *LocationInformationResponse*

Die Struktur der *LocationInformationRequest* sieht wie folgt aus:



1.2.2 TripRequest (Verbindungsauskunft) – Beschreibung

Dieser Dienst berechnet intermodale Verbindungen von einem Startpunkt zu einem Zielpunkt unter Berücksichtigung unterschiedlicher benutzerspezifischer Präferenzen.

1.2.2.1 Verbindungsauskunft – Anfragestrukturen

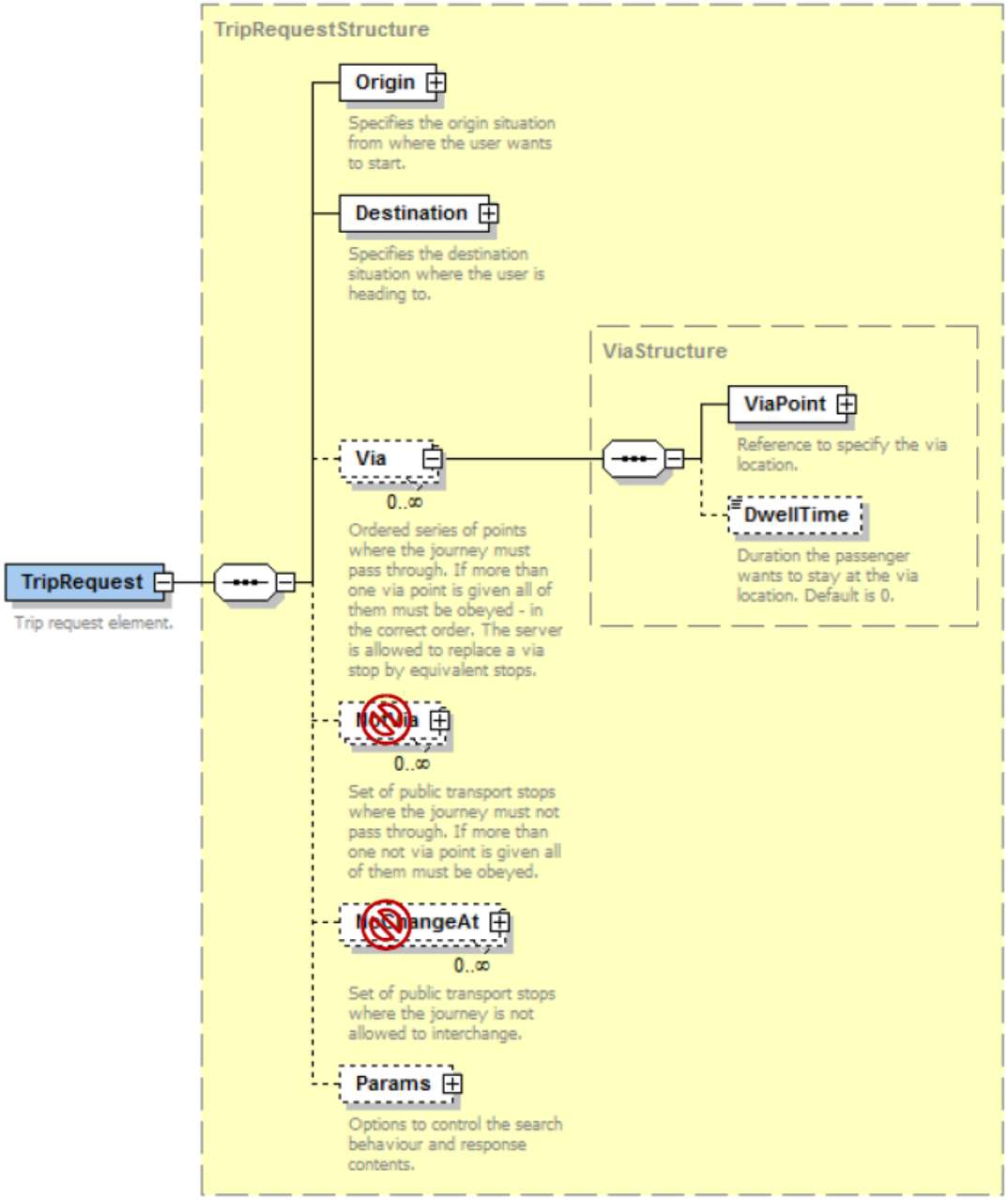
Die intermodale Verbindungsauskunft wird mit Hilfe eines Elements *TripRequest* angefordert. In der EFA erfolgt die Anfrage einer Verbindungsauskunft über den XML_TRIP_REQUEST2. Eine einfache Verbindungsanfrage, die nur Start, Ziel und Abfahrtszeit enthält, sieht demnach so aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<Trias version="1.0" xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="trias file:///C:/mentzdv/Trias/TRIASSchemas/Trias.xsd">
  <ServiceRequest>
    <siri:RequestTimestamp>2016-03-30T13:36:00</siri:RequestTimestamp>
    <siri:RequestorRef>SEUS</siri:RequestorRef>
    <RequestPayload>
      <TripRequest>
        <Origin>
          <LocationRef>
            <StopPlaceRef>de:14612:28</StopPlaceRef>
            <LocationName>
              <Text>
              </Text>
            </LocationName>
          </LocationRef>
          <DepArrTime>2016-03-30T160:39:00</DepArrTime>
        </Origin>
        <Destination>
          <LocationRef>
            < StopPlaceRef>de:14523:311</StopPlaceRef>
            <LocationName>
              <Text>
              </Text>
```

```
          </LocationName>
        </LocationRef>
      </Destination>
      <Params>
        <PtModeFilter>
        </PtModeFilter>
      </Params>
    </TripRequest>
  </RequestPayload>
 </ServiceRequest>
</Trias>
```

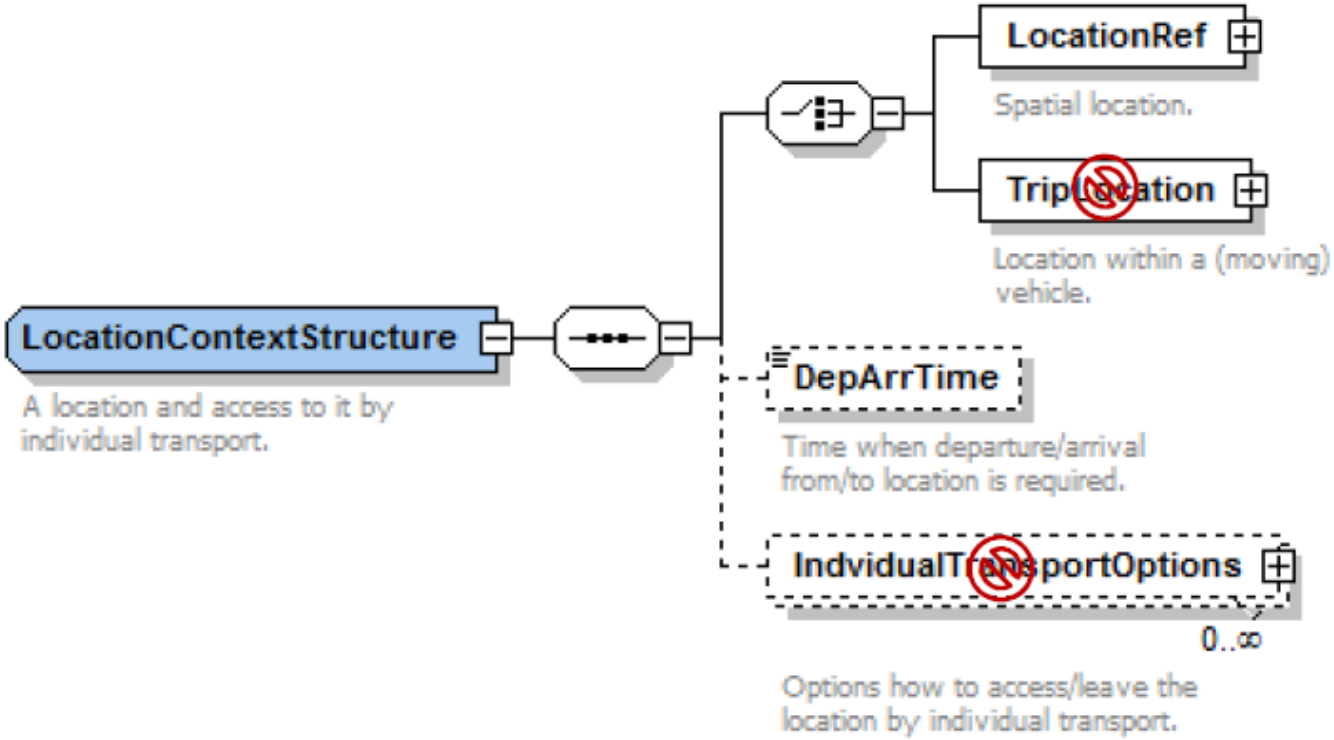
Beispiel 4: *TripRequest*

Die zugrunde liegende Struktur der TRIAS-Verbindungsauskunft lässt sich wie folgt darstellen:

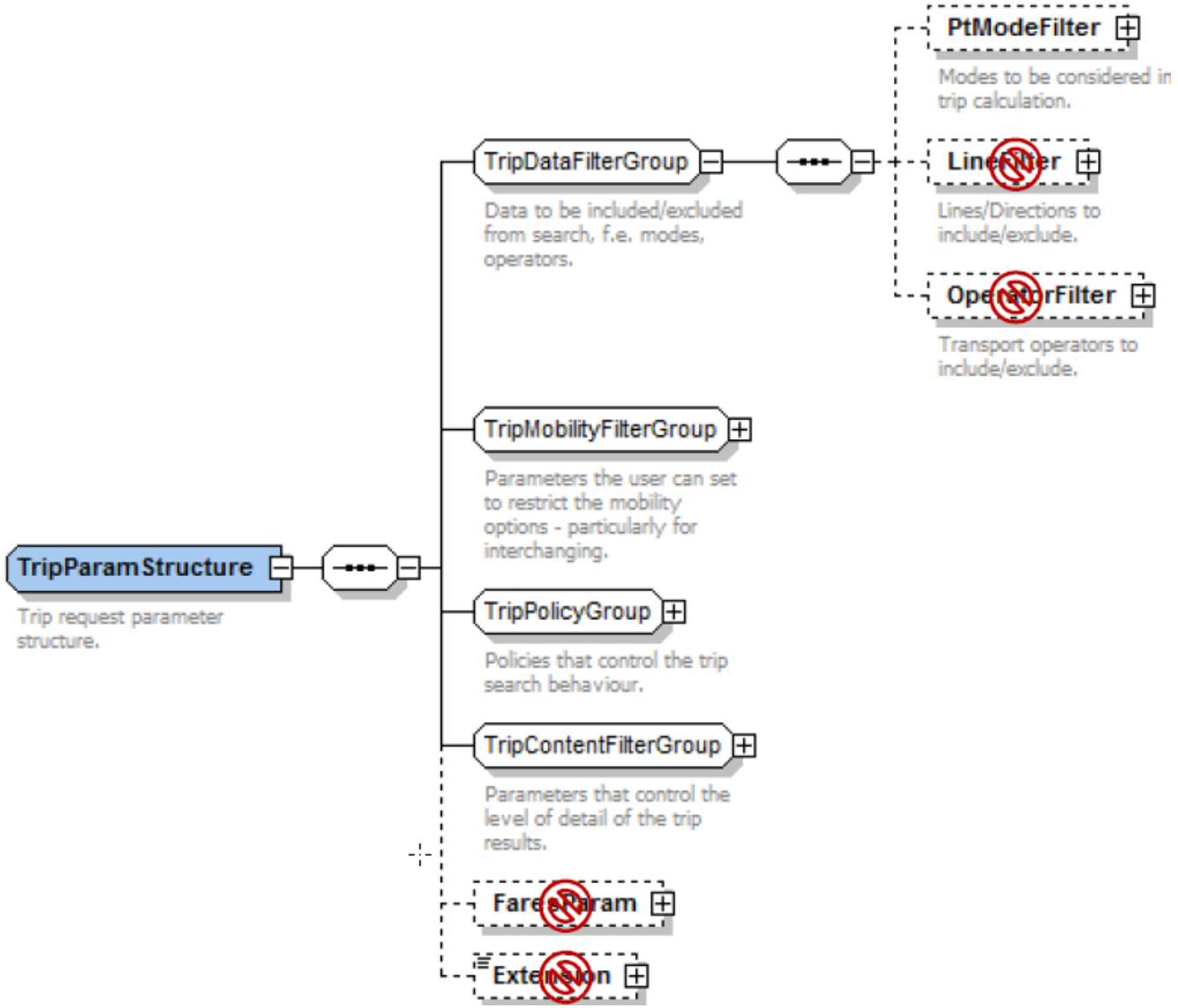


- *Origin*: enthält die *LocationContextStructure* des Startelements
- *Destination*: enthält die *LocationContextStructure* für das Ziel
- *Via*:
  - *ViaPoint*: enthält die *LocationRefStructure* für das *Via*-Element
  - *DwellTime*: enthält die Dauer des Aufenthalts an der Via-Haltestelle
- *Params:* Parameter, die die Suche und Rückgabewerte beeinflussen können

Das für *Origin* und *Destination* definierte Inhaltsmodell, die *LocationContextStructure,* stellt sich jeweils so dar wie in unten stehender Abbildung, wobei *DepArrTime* die Ankunfts- bzw. Abfahrtszeit und *LocationRef* die *LocationRefStructure* für Start oder Ziel enthält:

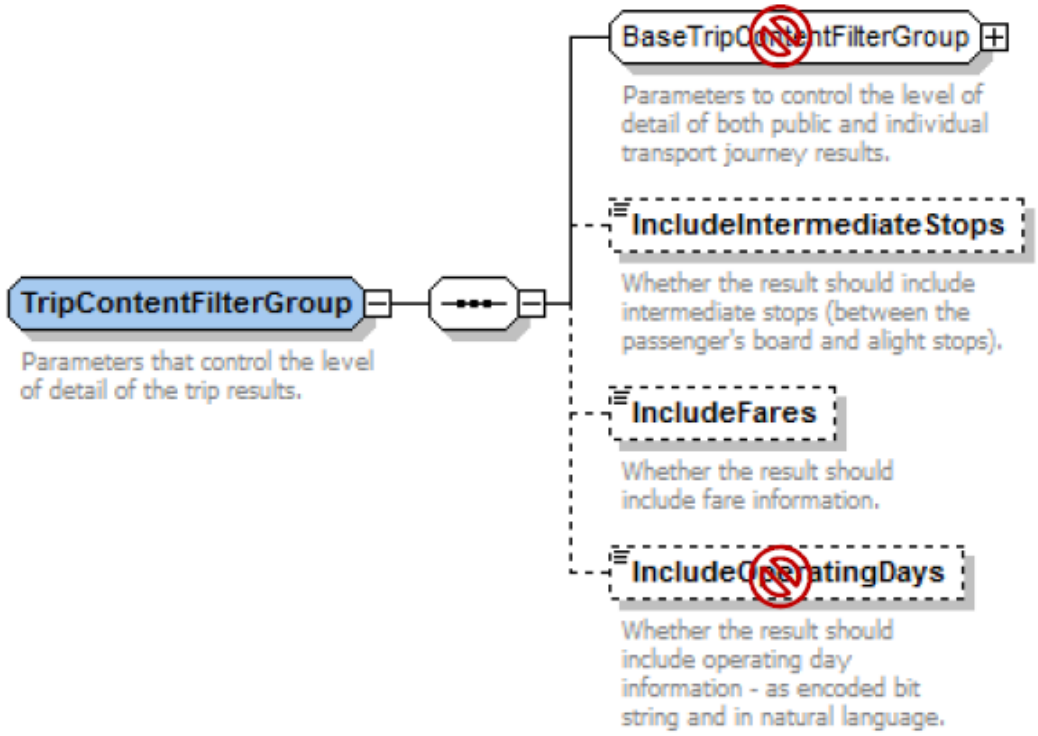


Das Inhaltsmodell von *Params* wird in der *TripParamStructure* beschrieben:

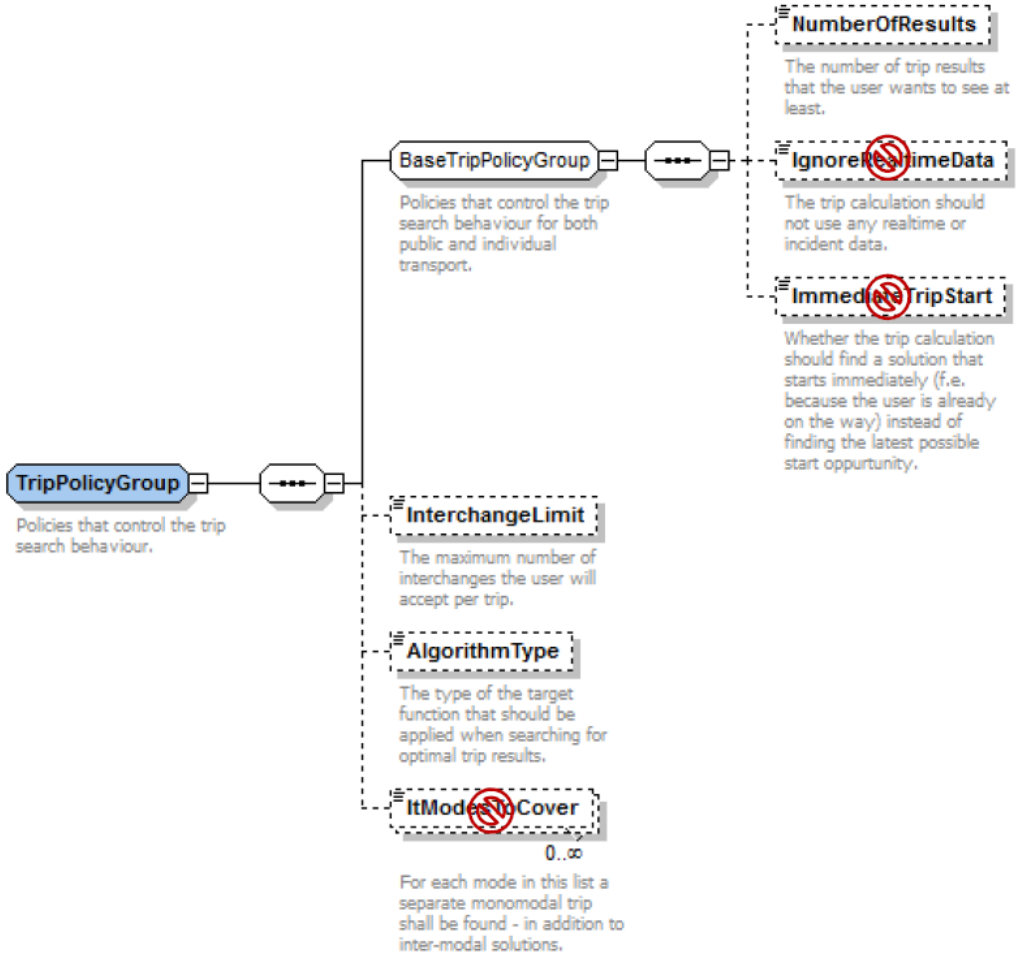


- *PtModeFilter*: Angabe von Verkehrsmittelein- bzw. ausschlüssen
- *TripMobilityFilterGroup*: Angabe von Mobilitätseinschränkungen
- *TripPolicyGroup*: Parameter zur Fahrtberechnung
- *TripContentFilterGroup*: Steuerung der Elemente in der Ergebnisstruktur

Mit der *TripContentFilterGroup* können Filter für die Antwortstruktur vorgegeben werden. So kann beispielsweise gesteuert werden, ob Zwischenhalte in der Antwort übermittelt (*IncludeIntermediateStops*) und Fahrpreisinformationen ausgegeben werden sollen (*IncludeFares*):

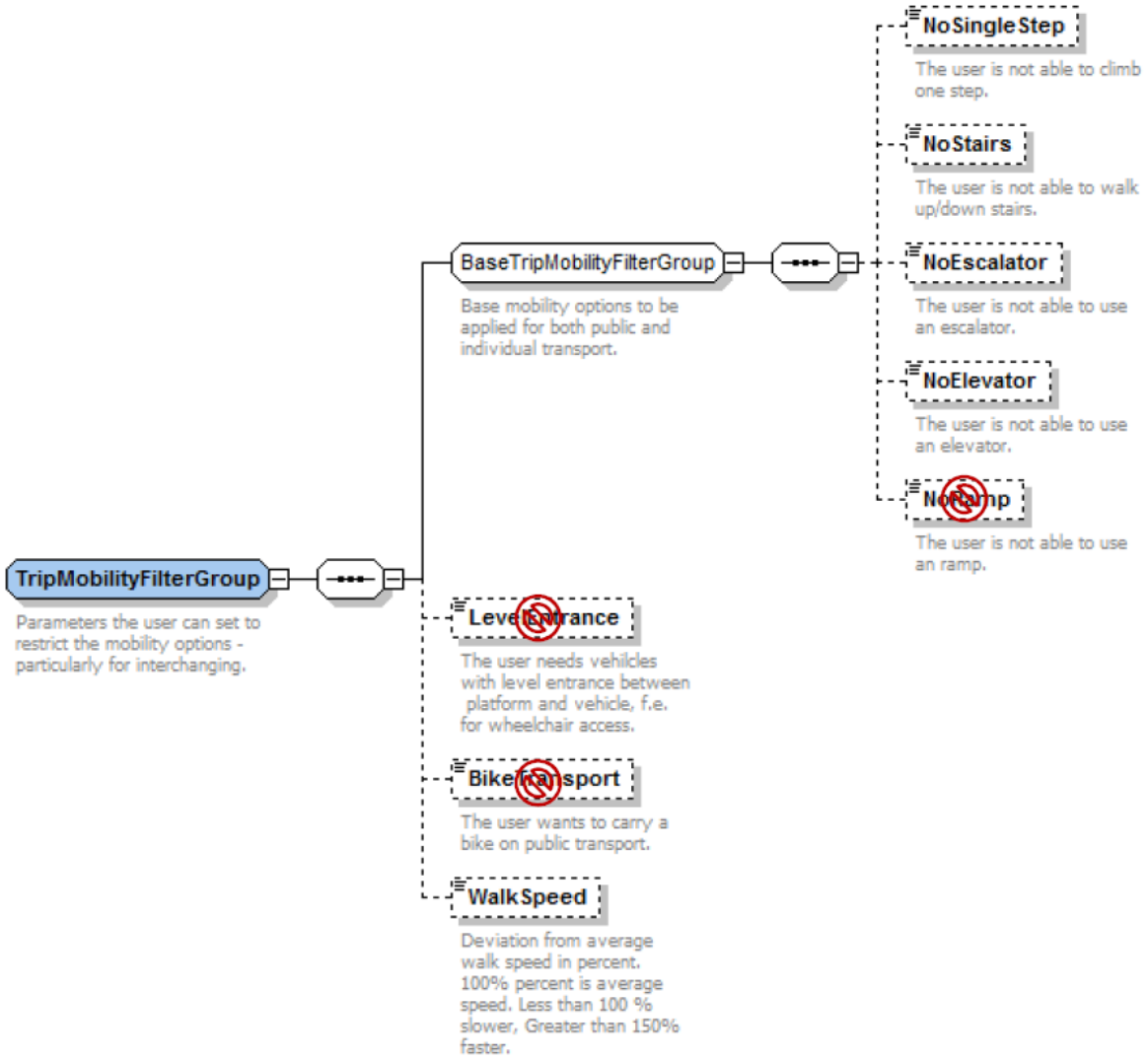


Innerhalb der *TripPolicyGroup* können Parameter übergeben werden, die die Fahrtberechnung beeinflussen:



- *NumberOfResults*: Anzahl der Fahrten, die zurückgeliefert werden soll
- *InterchangeLimit*: Maximale Anzahl der zugelassenen Umstiege
- *AlgorithmType*: legt den Algorithmus der Berechnung fest. Zugelassene Werte:
    - *fastest* (Verbindung mit schnellster Reisezeit)
    - *minChanges* (Verbindung mit wenigen Umstiegen)
    - *leastWalking* (Verbindung mit wenig Fußwegen)
    - *leastCost* (konstengünstigste Verbindung)

Die *TripMobilityFilterGroup* erlaubt die Auswahl von Mobilitätseinschränkungen bei der Suche. Ihre Struktur ist wie folgt definiert:



- *NoSingleStep*: Keine Stufen (alles ebenerdig)
- *NoStairs*: Keine Treppen
- *NoEscalator*: Keine Rolltreppen
- *NoElevator*: Keine Fahrstühle
- *WalkSpeed*: Gehgeschwindigkeit prozentual zur Normalgeschwindigkeit. Der Wert 100 stellt den Standard dar. Werte kleiner 100 stellen eine langsamere Geschwindigkeit dar, Werte größer 100 eine schnellere

1.2.2.2 Verbindungsauskunft – Antwortstrukturen

Das Ergebnis einer intermodalen Verbindungsanfrage wird innerhalb des Elements *TripResponse* übertragen.

Auszug der Antwort zur Anfrage aus Kapitel 3.2.2.1:

```
<?xml version="1.0" encoding="utf-8"?>
<Trias version="1.0" xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" xmlns:mw="http://services.mentzdv.de/2010/07/middleware">
 <ServiceDelivery>
  <siri:ResponseTimestamp>2016-03-30T16:10:19</siri:ResponseTimestamp>
  <siri:ProducerRef>VVO</siri:ProducerRef>
  <siri:ResponseMessageIdentifier>16-10-17-010</siri:ResponseMessageIdentifier>
  <DeliveryPayload>
   <TripResponse>
    <TripResult>
     <ResultId>ID251132621431972051867216323215135242477019 9</ResultId>
     <Trip>
      <TripId>ID591201721661892261926514015520029245132203 8</TripId>
      <Duration>PT3H21M</Duration>
      <StartTime>2016-03-30T14:53:00</StartTime>
      <EndTime>2016-03-30T18:14:00</EndTime>
      <Interchanges>2</Interchanges>
      <TripLeg>
       <LegId>ID942021563343175927315164194123614025159</LegId>
       <TimedLeg>
        <LegBoard>
         <StopPointRef>de:14612:28:12</StopPointRef>
         <StopPointName>
```

```
      <Text>Dresden Hauptbahnhof</Text>
      <Language>germ</Language>
    </StopPointName>
    <PlannedBay>12</PlannedBay>
    <ServiceDeparture>
      <TimetabledTime>2016-03-30T14:53:00</TimetabledTime>
    </ServiceDeparture>
    <StopSeqNumber>1</StopSeqNumber>
  </LegBoard>
  <LegAlight>
    <StopPointRef>de:14524:41032:2</StopPointRef>
    <StopPointName>
      <Text>Zwickau, Hauptbahnhof</Text>
      <Language>germ</Language>
    </StopPointName>
    <PlannedBay>2</PlannedBay>
    <ServiceArrival>
      <TimetabledTime>2016-03-30T16:27:00</TimetabledTime>
    </ServiceArrival>
    <StopSeqNumber>2</StopSeqNumber>
  </LegAlight>
  <Service>
    <OperatingDayRef>2016-03-30</OperatingDayRef>
    <JourneyRef>germ:ddb:90510:4790</JourneyRef>
    <LineRef>germ:ddb:90510</LineRef>
    <DirectionRef>outward</DirectionRef>
    <Mode>
      <PtMode>rail</PtMode>
      <Name>
        <Text>
        </Text>
        <Language>germ</Language>
      </Name>
    </Mode>
    <PublishedLineName>
      <Text>4790</Text>
```

```
            <Language>germ</Language>
          </PublishedLineName>
          <DestinationStopPointRef>9000966</DestinationStopPointRef>
          <DestinationText>
            <Text>Hof Hbf</Text>
            <Language>germ</Language>
          </DestinationText>
        </Service>
      </TimedLeg>
    </TripLeg>
    …
```

Beispiel 4: *TripResponse*

1.2.3 StopEventRequest (Abfahrtstafeln) – 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.

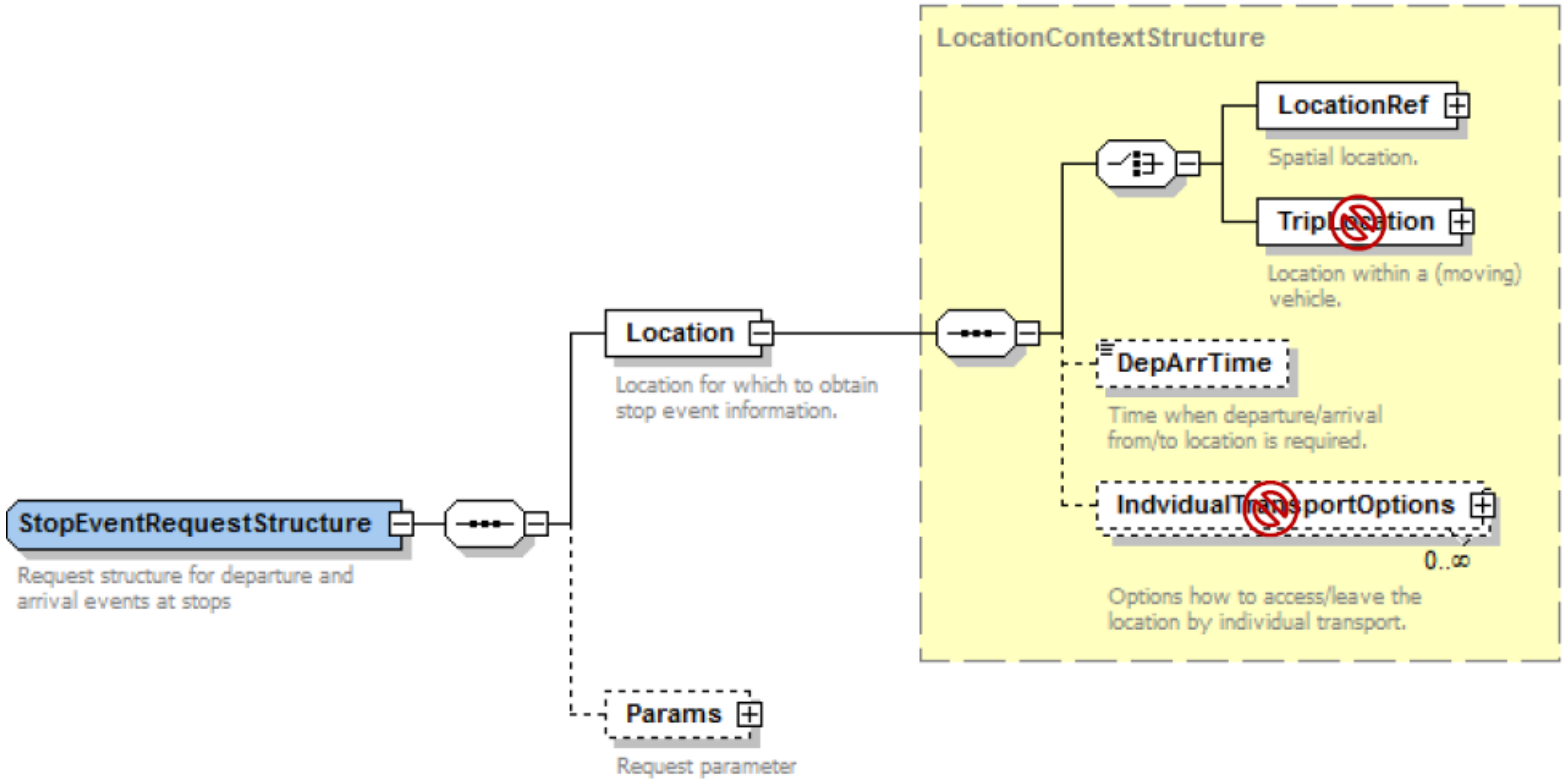
1.2.3.1 Abfahrtstafeln – Anfragestrukturen

Die Abfahrts- bzw. Ankunftstafel wird über das Element *StopEventRequest* angefordert. Das entsprechende Äquivalent in der EFA ist der XML_DM_REQUEST. Eine TRIAS-Anfrage zu einem *StopEventRequest* könnte z.B. so aussehen:

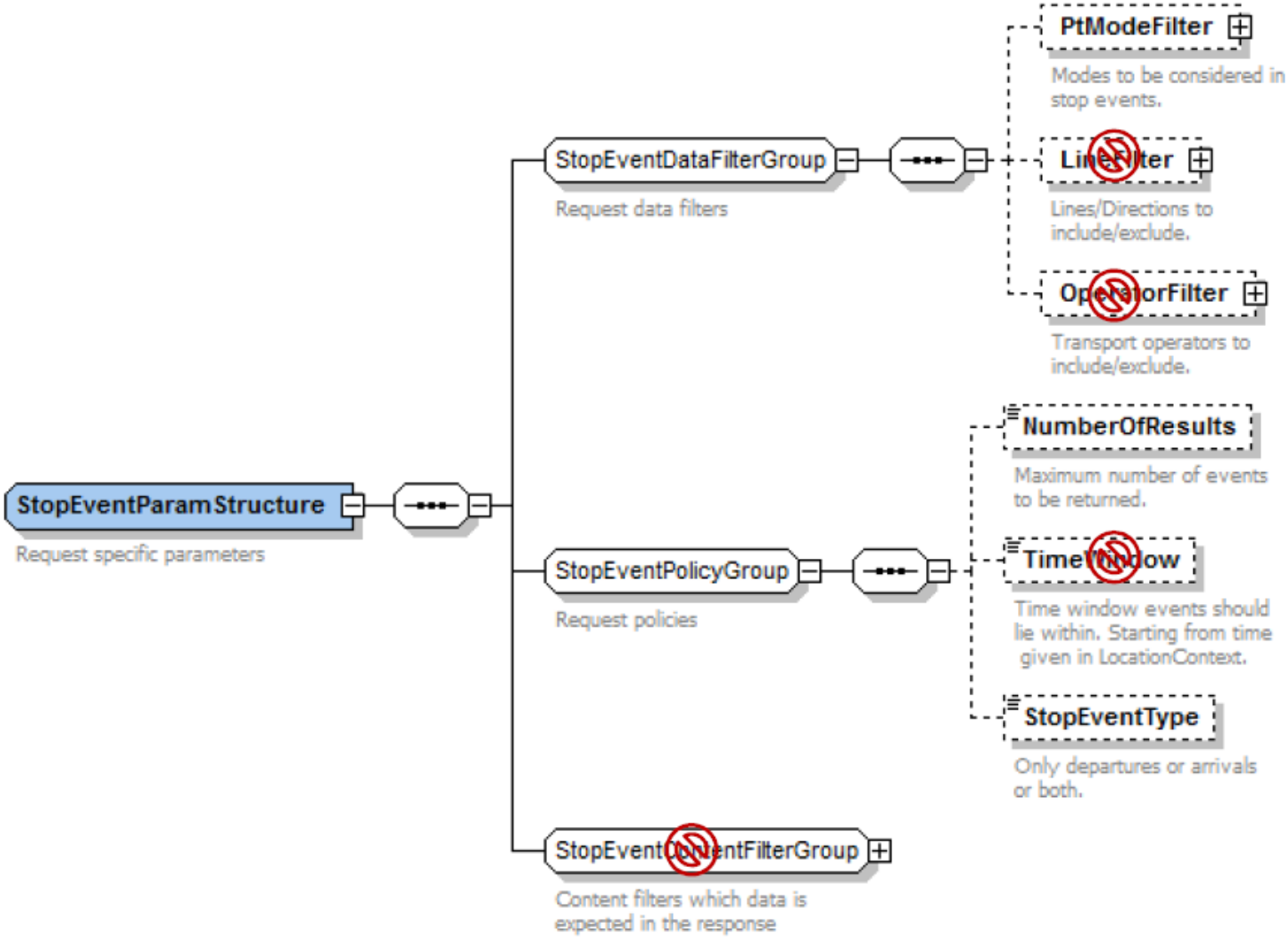
```
<?xml version="1.0" encoding="UTF-8"?>
<Trias version="1.0" xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="trias file:///C:/mentzdv/Trias/TRIASSchemas/Trias.xsd">
  <ServiceRequest>
    <siri:RequestTimestamp>2016-03-30T13:36:00</siri:RequestTimestamp>
    <siri:RequestorRef>SEUS</siri:RequestorRef>
    <RequestPayload>
      <StopEventRequest>
        <Location>
          <LocationRef>
            <StopPlaceRef>de:14612:28</StopPlaceRef>
            <LocationName>
              <Text>Dresden, Hauptbahnhof</Text>
              <Language>de</Language>
            </LocationName>
          </LocationRef>
        </Location>
        <Params>
         <StopEventType>departure</StopEventType>
        </Params>
      </StopEventRequest>
    </RequestPayload>
  </ServiceRequest>
</Trias>
```

Beispiel 5: *StopEventRequest*

Die zu Grunde liegende Struktur der Anfrage sieht wie folgt aus:



Innerhalb der *StopEventParamStructure* können der Anfrage Parameter mitgegeben werden:



Folgende Parameter werden unterstützt:

- *PtModeFilter*: Ein- bzw. Ausschluss von Verkehrsmitteln
- *NumberOfResults*: Anzahl der Ergebnisse
- *StopEventType*: Ankunfts- und/oder Abfahrtsmonitor (Werte: *departure*, *arrival* oder *both*)

1.2.3.2 Abfahrtstafeln – Antwortstrukturen

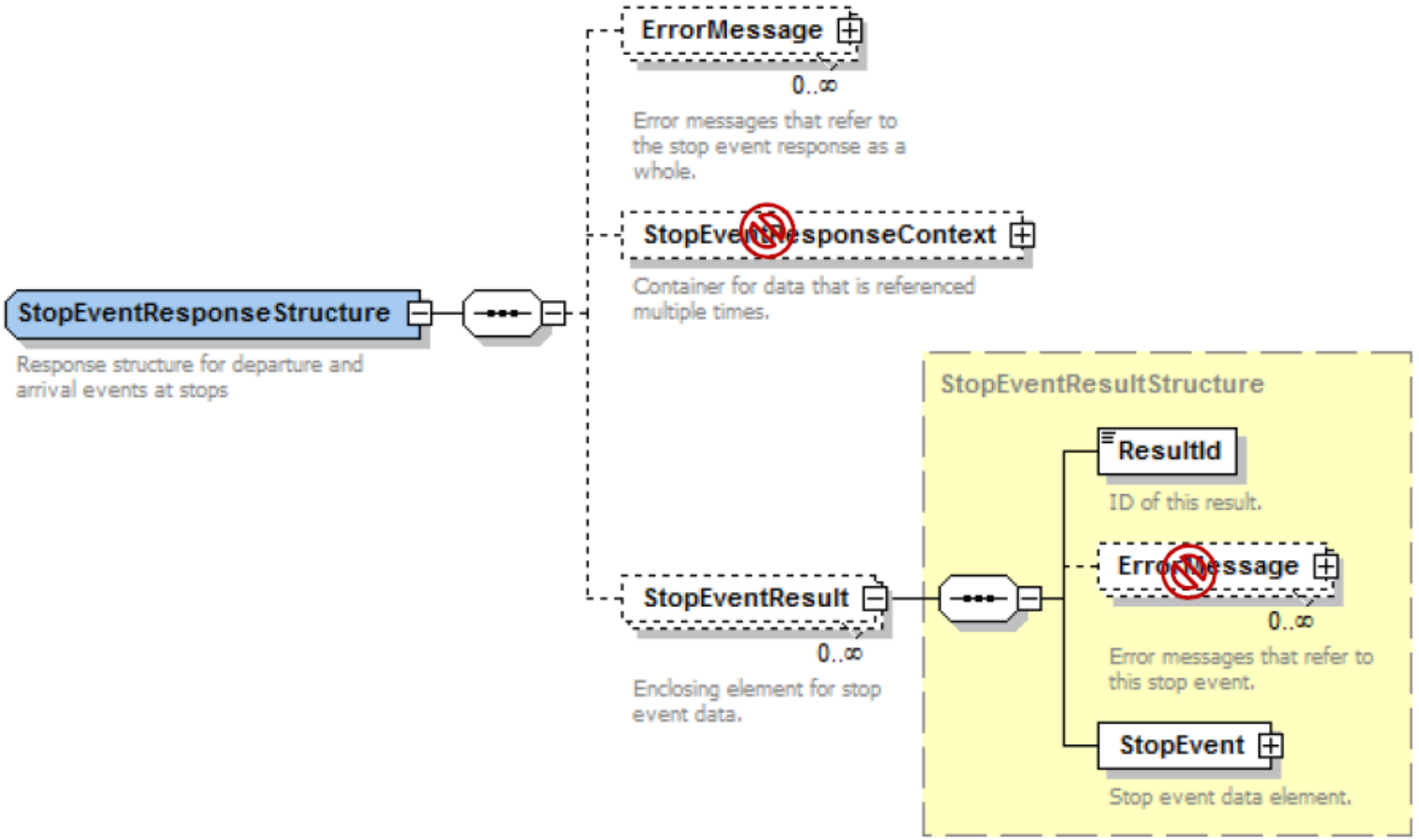
Das Ergebnis einer Abfahrtstafelanfrage wird mittels eines Elements *StopEventResponse* übertragen und hier nur als Auszug ausgegeben, da die Antworten sehr umfangreich sein können.

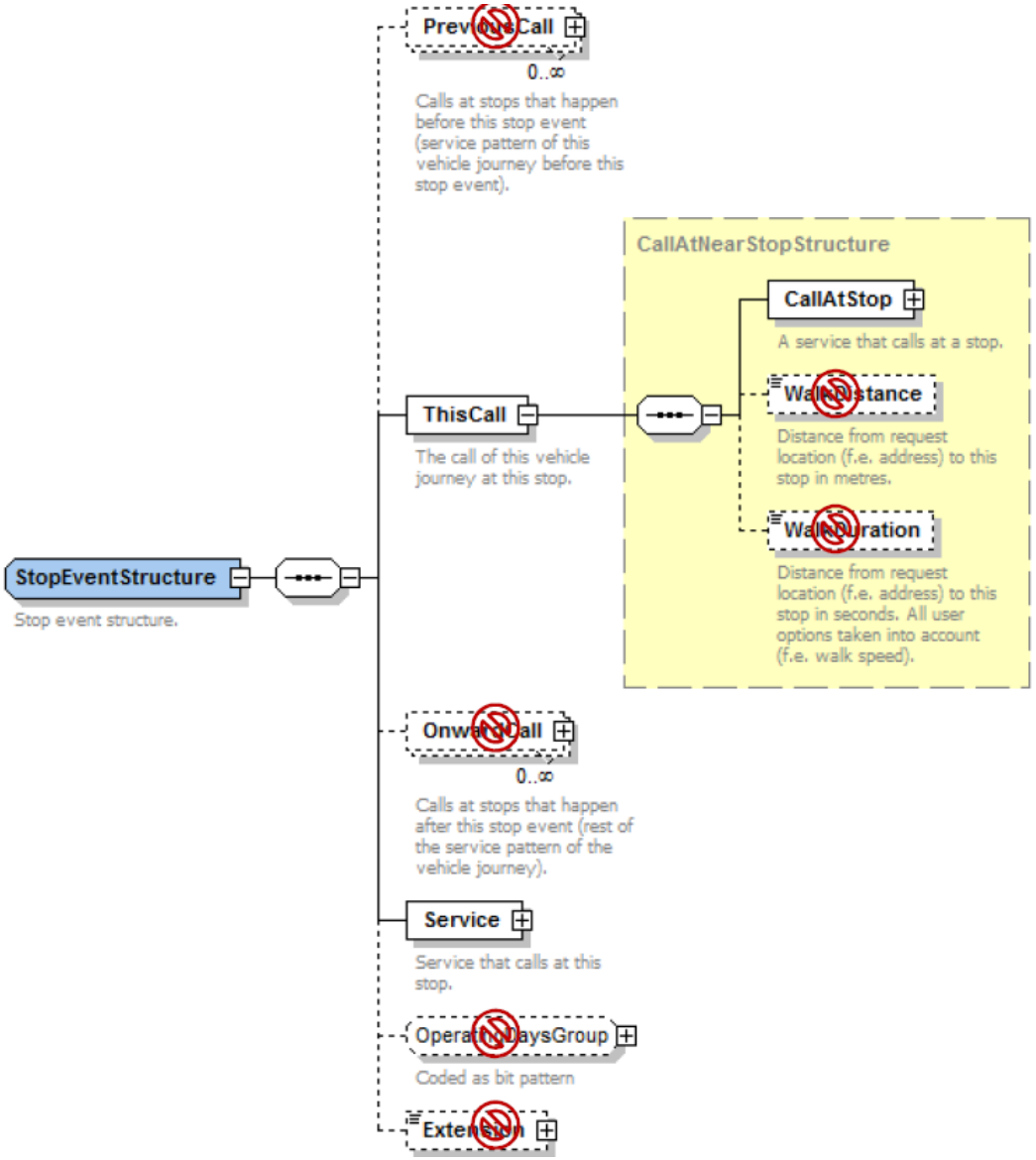
```
<?xml version="1.0" encoding="utf-8"?>
<Trias version="1.0" xmlns="trias" xmlns:siri="http://www.siri.org.uk/siri" xmlns:mw="http://services.mentzdv.de/2010/07/middleware">
  <ServiceDelivery>
    <siri:ResponseTimestamp>2016-03-30T16:10:40</siri:ResponseTimestamp>
    <siri:ProducerRef>VVO</siri:ProducerRef>
    <siri:ResponseMessageIdentifier>16-10-40-011</siri:ResponseMessageIdentifier>
    <DeliveryPayload>
      <StopEventResponse>
        <StopEventResult>
          <ResultId>ID1902910246208503373177951842401783225 1197</ResultId>
          <StopEvent>
            <ThisCall>
              <CallAtStop>
                <StopPointRef>de:14612:28:2</StopPointRef>
                <StopPointName>
                  <Text>Dresden Hauptbahnhof</Text>
                  <Language>germ</Language>
                </StopPointName>
                <PlannedBay>2</PlannedBay>
                <ServiceDeparture>
                  <TimetabledTime>2016-03-30T16:10:00</TimetabledTime>
                </ServiceDeparture>
                <StopSeqNumber>1</StopSeqNumber>
              </CallAtStop>
```

```
      </ThisCall>
      <Service>
        <OperatingDayRef>2016-03-30</OperatingDayRef>
        <JourneyRef>germ:voe:11010:449</JourneyRef>
        <LineRef>germ:voe:11010</LineRef>
        <DirectionRef>outward</DirectionRef>
        <Mode>
          <PtMode>tram</PtMode>
          <Name>
            <Text>Straßenbahn</Text>
            <Language>germ</Language>
          </Name>
        </Mode>
        <PublishedLineName>
          <Text>10</Text>
          <Language>germ</Language>
        </PublishedLineName>
        <DestinationStopPointRef>33000001</DestinationStopPointRef>
        <DestinationText>
          <Text>Dresden Bahnhof Mitte</Text>
          <Language>germ</Language>
        </DestinationText>
      </Service>
    </StopEvent>
  </StopEventResult>
  …
```

Beispiel 6: *StopEventResult*

Die Struktur der Antwort lässt sich wie folgt darstellen:





Die *CallAtStopStructure* enthält alle Informationen zur Haltestelle und zu den An- bzw. Abfahrten: