HaCon

Traffic • Software • Service

Specification

HAFAS Public Access Gateway

Easy access to

HAFAS journey planner systems

Version 1.07

2012-06-04

HaCon Ingenieurgesellschaft mbH
Lister Str. 15
30163 Hanover
Germany

1 Changelog

Version	Date	Changes
1.7	2012-06-04	- Added Changelog - Added SystemInfo service

2 Introduction

The public interface is implemented as a ReST[1] (**Re**presentational **S**tate **T**ransfer) interface which provides different methods for the different functionalities of the journey planner, which are the following services:

- Location
- Trip
- DepartureBoard
- ArrivalBoard
- JourneyDetail
- Geometry
- SystemInfo

While Location, Trip, ArrvialBoard and DepartureBoard can be called directly the JourneyDetail-Method can only be called by a reference given in a result of the Trip, or DepartureBoard services. Geometry-Method can only be called by a reference given in a result of the Trip or JourneyDetail request.

The system only implements read-only GET requests which are called by given service URLs and multiple GET parameters to specify the requested journey planner information. The parameter values need to be in ISO-8859-1 URL encoded. The result of each request will be delivered either as XML or JSON (see 3.7) response. If the encoding of URL parameters is not right, the behaviour of the system might deliver unexpected results.

From now on it is assumed, that you have been provided with a base URL of the HAFAS system. The following documentation of the different requests are described based on this given base url <*baseurl*>.

[1] See http://rest.elkstein.org/ for a tutorial on ReST interfaces.

3 General principles

There are some general principles which are valid for the different services which are described in this section.

3.1 Coordinates

Coordinates are always in the WGS84 system, represented as decimal degrees in the interval -90 to 90 for the latitude (lat) and -180 to 180 for the longitude (long).

3.2 Date and time formats

Dates are always represented in the format YYYY-MM-DD. This applies both for request parameters as for dates in responses. Times are always represented in the format HH:MM in 24h nomenclature.

3.3 Stateless service vs. data dependency

All services of the provided interface are stateless as it is required for a ReST protocol. But this has its limitation concerning the journey planner's timetable data. As soon as the timetable data is exchanged (in most cases daily on weekdays), IDs of stops/stations are not necessary valid anymore. The same applies for reference URLs to retrieve journeyDetails. The storage of stop/station IDs and reference URLs to journeyDetails for a longer period except the current user session is not recommended therefore and can only be done on own risk for undetermined behaviour when reusing these IDs or references.

3.4 Route index

A route is the list of stops/stations where a vehicle like a train or bus stops. Every stop/station on a route has its own index which can be used as a reference. This index is also used to identify distinctively if the same stop/station if it is contained several times in one route.

3.5 Realtime information

Realtime information will be included in the service as far as it is available in the web based journey planner. It is always delivered in addition to the planned departures and arrivals.

3.6 Versioning

Due to enhancements of the API the input parameters and the results can change over time. Different Versions of the API will be available at the same time.
The requested version can be specified by using the version number in the path info:
http://<baseUrl>/<version>/<servicename>
The version part is optional, if it is omitted, the latest version will be used. Be aware that omitting the version can break your client when a new API version is introduced. If your client should always use a special version of the api (v2 for example), your url would look like this: http://<baseUrl>/v2/<servicename>

3.7 Response Format

The interface returns responses either in XML (default) or JSON format.
In order to request a JSON response you have to append the following parameter to each call of the interface: **`format=json`**. If JSONP is needed you can append an additional parameter to specify the name of callback function, the JSON object will be wrapped by a function call with this name: **`jsonpCallback=mycallback.`**
The JSON content is generated by converting the xml content to JSON automatically. The conversion is done by the following simple rules:

- Element names become object properties
- Text (PCDATA) becomes an object property with name "$"
  <a>foo becomes { "a": { "$" : "foo" } }
- Nested elements become nested properties
  <a>foo<c>foo</c>
  becomes
  { "a": { "b" : { "$": "foo" }, "c": { "$": "foo"} } }
- If there are multiple elements with the same name the JSON code contains an array for these element.
  <a>foo1foo2
  becomes
  { "a": { "b" : [{"$": foo1" }, {"$": "foo2" }] } }

- Attribute names become object properties

  foo2

  becomes

  { "a": { "atb" : "foo1", "$" : "foo2" } }

The following example shows a trip in a xml response and the resulting conversion to JSON:

XML:

```
 <Trip>
   <Leg name="Buss 61" type="BUS" id="9015014006100032" direction="Vänersborg Resecentrum">
    <Origin name="Trollhättan, Resecentrum, Trollhättan" type="ST" id="9022014081032014" routeIdx="8" time="13:12"
date="2011-09-13" track="L " />
    <Destination name="NÄL, Trollhättan" type="ST" id="9022014081089002" routeIdx="22" time="13:33" date="2011-09-13"
track="B " />
   </Leg>
 </Trip>
```

JSON:

```
"Trip": {
        "Leg": {
                "name": "Buss 61",
                "type": "BUS",
                "id": "9015014006100032",
                "direction": "Vänersborg Resecentrum",
                "Origin": {
                        "name": "Trollhättan, Resecentrum, Trollhättan",
                        "type": "ST",
                        "id": "9022014081032014",
                        "routeIdx": "8",
                        "time": "13:12",
                        "date": "2011-09-13",
                        "track": "L "
                },
                "Destination": {
                        "name": "NÄL, Trollhättan",
                        "type": "ST",
                        "id": "9022014081089002",
                        "routeIdx": "22",
                        "time": "13:33",
                        "date": "2011-09-13",
                        "track": "B "
                }
        }
}
```

3.8 Authentification

Every client using the api needs to pass a valid authentification key in every request.

The following parameter has to be appended to the url: **`authKey=<your_key_here>.`**

Please contact Västtrafik in order to request an authentification key.

4 Services

4.1 Location Service

There are 3 different types of the location service which can be used to get a list of locations using different input parameters.
The response format for all services is defined in `hafasRestLocation.xsd` (see also 5.1 for further details).

4.1.1 Location.name Service

The `location.name` service can be used to perform a pattern matching of a user input and to retrieve a list of possible matches in the journey planner database. Possible matches might be stops/stations, points of interest and addresses. For reasons of backward compatibility the service name `location` can be used as an alias for `location.name`.

The service has only one GET parameter which is called `input`. This parameter contains a string with the user input. The result is a list of possible matches (locations) where the user might pick one entry to perform a trip request with this location as origin or destination or to ask for a departure board or arrival board of this location (stops/stations only) .

The URL to call the service is the following:
http://<baseurl>/location.name?input=user%20input

4.1.2 Location.allstops Service

The `location.allstops` service returns a list of all stops available in the journey planner. Be aware that a call of this service is very time consuming and should be only requested when it is really needed.

The URL to call the service is the following:

http://<baseurl>/location.allstops

4.1.3 Location.nearbystops Service

The `location.nearbystops` service returns a list of stops around a given center coordinate. The returned results are ordered by their distance to the center coordinate.

Possible parameters:

Name	Use	Range	Default	Description
originCoordLat	Mandatory	See 3.1	-	Latitude of center coordinate
originCoordLong	Mandatory	See 3.1	-	Longitude of center coordinate
maxNo	Optional	1-1000	10	Maximum number of returned stops
maxDist	Optional	1-3000	1000	Maximum distance from the center coordinate

The URL to call the service is the following:
http://<baseurl>/location.nearbystops?originCoordLong=11.981211&originCoordLat=57.709792&maxNo=5

4.2 Trip service

4.2.1 Origin, destination and via

The `trip` service calculates a trip from a specified origin to a specified destination. These might be stop/station IDs or coordinates based on addresses and points of interest validated by the location service or coordinates freely defined by the client.

Both origin and destination are mandatory parameters for the trip service.
The parameters are named either `originId` or `originCoordLat`, `originCoordLong`, and `originCoordName`. For the destination the parameters are named either `destId` or `destCoordLat`, `destCoordLong` and `destCoordName`.
The `origin/dest-CoordName` parameters are the names of the address at the specified coordinate.

It is possible to define a via stop/station. This forces the journey planner to search for trips which pass the defined station. The parameter is called `viaId`.

4.2.2 Date and time

The departure time and date are defined with the parameters `date` and `time`. If the date is not set the current date will be used (server time). If the parameter time is not set the current server time will be used to perform the request.
To specify that the given time and date is not the departure time but the latest time to arrive at the destination you can use the parameter `searchForArrival=1`.

4.2.3 Types of transport

It is possible to switch off specific means of transport by using one of the following optional parameters:

- `useVas=0` `// Vasttågen`
- `useLDTrain=0` `// Long Distance Trains`
- `useRegTrain=0` `// Regional Trains`
- `useBus=0`
- `useBoat=0`
- `useTram=0`

The default value is that all means of transport are switched on (value 1). If no parameter is set this default value applies.

In addition to the flags above you can use `excludeDR=1` to exclude journeys which require tel. registration, by default they are included.

4.2.4 Walks from and to coordinates

In order to set the maximum walking distance from/to a coordinate the parameter `maxWalkDist` can be used to pass the distance in meters (default value: 2000).

The walking speed can be influenced using the parameter `walkSpeed`. The passed value is given in percent of “normal” speed (valid range: 20%--180%). Västtrafik’s official implementations use the following values for its preset walking speeds:

- Slow: 85%
- Normal: 100% (Y km/h)
- Fast: 115%

NOTE: the settings for `maxWalkDist` and `walkSpeed` don’t apply to walks between stops when changing from a public transport leg to another, only to the walks from/to a origin or destination coordinate.

4.2.5 Changes and change times

The maximum number of changes in the journeys returned by the trip service can be set using the parameter `maxChanges`=<integer>.

When calculating journey suggestions, the journey planner uses values for the minimal margin (in minutes) for changes between different public transport legs. This value, called *change margin*, is used to compensate for deviations from the planned time table times of the arriving and departing vehicles involved in a change, to minimize the risk of missing a connection. For each stop area, a default change margin has been defined. A common value of the default change margin is 5 minutes, but there are stops with both longer and shorter default change margins (in the approximate range 3-10 minutes).

In addition to the change margin, walk times between different stop areas are taken into account. Walk times between stop points within a stop area are currently not taken into account specifically, but are included implicitly in the default change margin.

In order to prolong the minimal change times between the public transport legs of the returned journeys the parameter `additionalChangeTime`=<number of minutes> can be set. The default value is 0 minutes. The minimal time of a change will thus be calculated as *default change margin* + `additionalChangeTime` + *walk time between stop areas*.

If the default change margin should be ignored, you can set the parameter `disregardDefaultChangeMargin=1` (default is “0”). The minimal time of a change will then be calculated as *0* + `additionalChangeTime` + *walk time between stop areas*.
IMPORTANT NOTE: journeys that are presented when the default change margin has been disregarded are not covered by Västtrafik's travel warranty (Swedish: resegaranti). This will be clarified in the trip response by addition of the attribute travelWarranty="false" to each trip-tag. When travel warranty has been void, this must be clearly communicated in connection with the presented journey suggestions, eg. as *"The travel warranty is void for this journey."/"Resegaranti gäller ej för denna resa."*

4.2.6 Response formatting

If the reference URL for the Journey Detail Service (4.4) is not needed in the result, you can pass the parameter `needJourneyDetail=0`.

If the parameter `needGeo=1` is passed in the URL of the trip service, the result will contain a reference link for each leg of the resulting trips which can be used to request the geometry of the leg using the Geometry service (4.5).

The number of the returned trips in the result can be specified using the parameter `numTrips=`<integer> with a valid range of 1 – 6. Note that this is an approximated number and the result may contain more trips than specified.

4.2.7 Example request

A trip request for a trip from Göteborg C to some coordinate on the 19th of September in 2011 at 7:02 am excluding busses as means of transport looks like this:

http://<baseurl>/trip?originId=9022014008000000&destCoordLat=<lat>&destCoordLong=<long>&destCoordName=<NameOfDestination>&date=2011-09-19&time=07:02&useBus=0

4.2.8 Response

As a result the service returns a result with the calculated trip with base information for every leg of the found trips. This will include arrival and departure stop/station, arrival and departure time (incl. realtime), and vehicle accessibility information, if available.

4.3 Stationboard services

The station board can be retrieved by a call to the `departureBoard`service. This method will return the next 20 departures (or less if not existing) from a given point in time or the next departures in a given timespan (see below).

The service can only be called for stops/stations by using according ID retrieved by the `location` method. The parameter is called `id`. The time and date are defined with the parameters `date` and `time`.

It is possible to switch off certain means of transport by using one or several of the following optional switches

- `useVas=0` `// Vasttågen`
- `useLDTrain=0` `// Long Distance Trains`
- `useRegTrain=0` `// Regional Trains`
- `useBus=0`
- `useBoat=0`
- `useTram=0`

The default value of these switches is 1 (on) which also applies if the parameter isn't defined at all.

In addition to the flags above you can use `excludeDR=1` to exclude journeys which require tel. registration, by default they are included.
The parameter `timeSpan` can be used to get the next departures in a specified timespan of up to 24 hours (unit: minutes, maximum value: 1439). If this parameter is not set, the result will contain the next 20 departures. If `timeSpan` is set you can further reduce the number of returned journeys by adding the parameter

`maxDeparturesPerLine`, which will cause only the given number of journeys for every combination of line and direction.

If the reference URL for the Journey Detail Service (4.4) is not needed in the result, you can pass the parameter `needJourneyDetail=0`.

In order to get only departures of vehicles with a specified direction you can use the parameter `direction=<stopid>.`

A departure board for Göteborg C main station for the next 20 departures on 19th September, 2011 at 07:02am excluding all busses can be retrieved by calling

`http://`*`<baseurl>`***`/departureBoard?id=`***`9022014008000000`*****`&date=2011-09-19&time=07:02&`***`useBus=0`**

As a response the service will return a result according to `hafasRestDepartureBoard.xsd`. This will contain a list of departures with train/line number, type of transport, vehicle accessibility (if available), departure times (incl. realtime), departure stop/stations (might be different from requested stop), direction text and a track information if available. Every departure will also contain a reference to the journey detail service.

In addition to departure boards the service `arrivalBoard` delivers arriving journeys at a specified stop. The parameters are identical to the parameters of the departureBoard service.

As a response the service will return a result according to `hafasRestArrivalBoard.xsd`. This will contain a list of arrival with train/line number, type of transport, vehicle accessibility (if available), arrival times (incl. realtime), departure stop/stations (might be different from requested stop), the name of the origin stop and a track information if available. Every arrival will also contain a reference to the journey detail service.

4.4 Journey detail service

The `journeyDetail` service will deliver information about the complete route of a vehicle. This service can't be called directly but only by reference URLs in a result of a `trip` or `departureBoard` request. It contains a list of all stops/stations of this journey including all departure and arrival times (with realtime data if available) and additional information like specific attributes about facilities and other texts.

The response will be returned a result according to the format described in `hafasRestJourneyDetails.xsd.`

4.5 Geometry service

The Geometry service will return the polyline for a leg. This service can't be called directly but only by reference URLs in a result of a trip or JourneyDetail request.
The result contains a list of WGS84 coordinates which can be used to display the polyline on a map.
The response will be returned as a result according to the format described in hafasRestGeometry.xsd.

4.6 SystemInfo Service

The SystemInfo service provides information about the travelplanner system and the underlaying data. It will return the begin of end of the timetable period and the creation date of the timetable data. It is called without any parameters (except the authentification key):

`http://``<baseurl>`**`/ systeminfo?authKey=<your authentification key>`**

5 Response formats

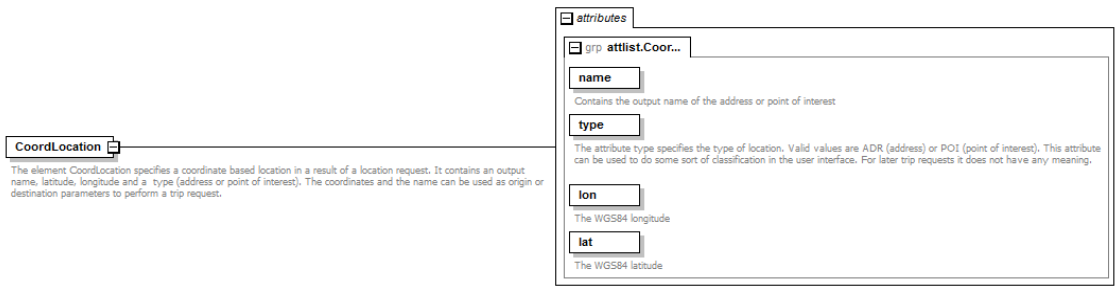
All services return their responses either in XML or JSON format (see 3.7). Every response is defined in a separate XSD file. The following sections will describe the responses more in detail. The formats might be enhanced in the future so the implementation of the parsing should be implemented in view of future possible changes.

5.1 Location response

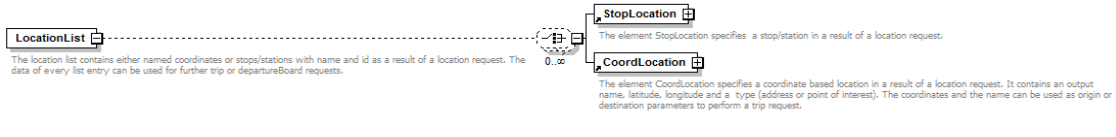
The location consists of a list of entries, which are either stops/stations or named coordinates. The root element of the response is `LocationList`.

Schema **hafasRestLocation.xsd**

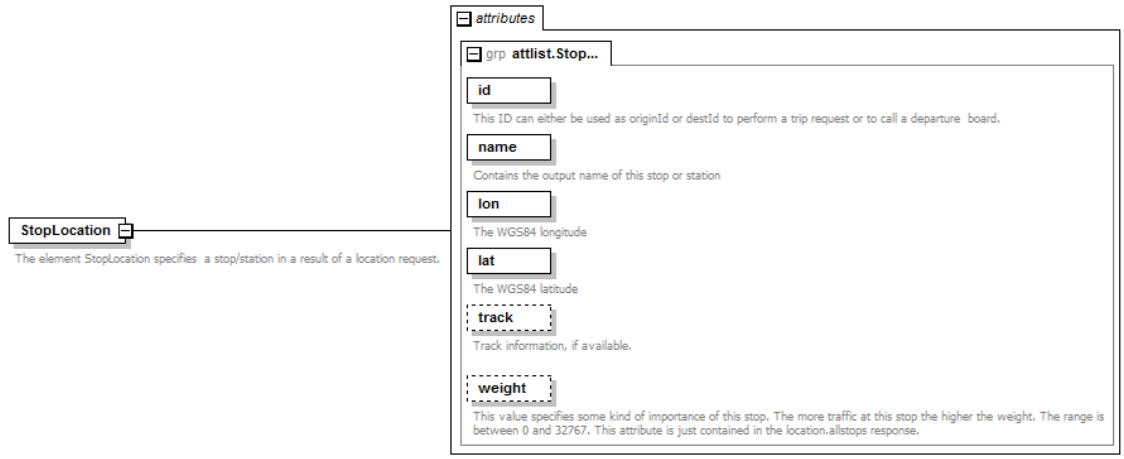
element **CoordLocation**

diagram	
used by	element **LocationList**

element **LocationList**

diagram	
children	**StopLocation** **CoordLocation**

element **StopLocation**

diagram	
used by	element **LocationList**

Example Response

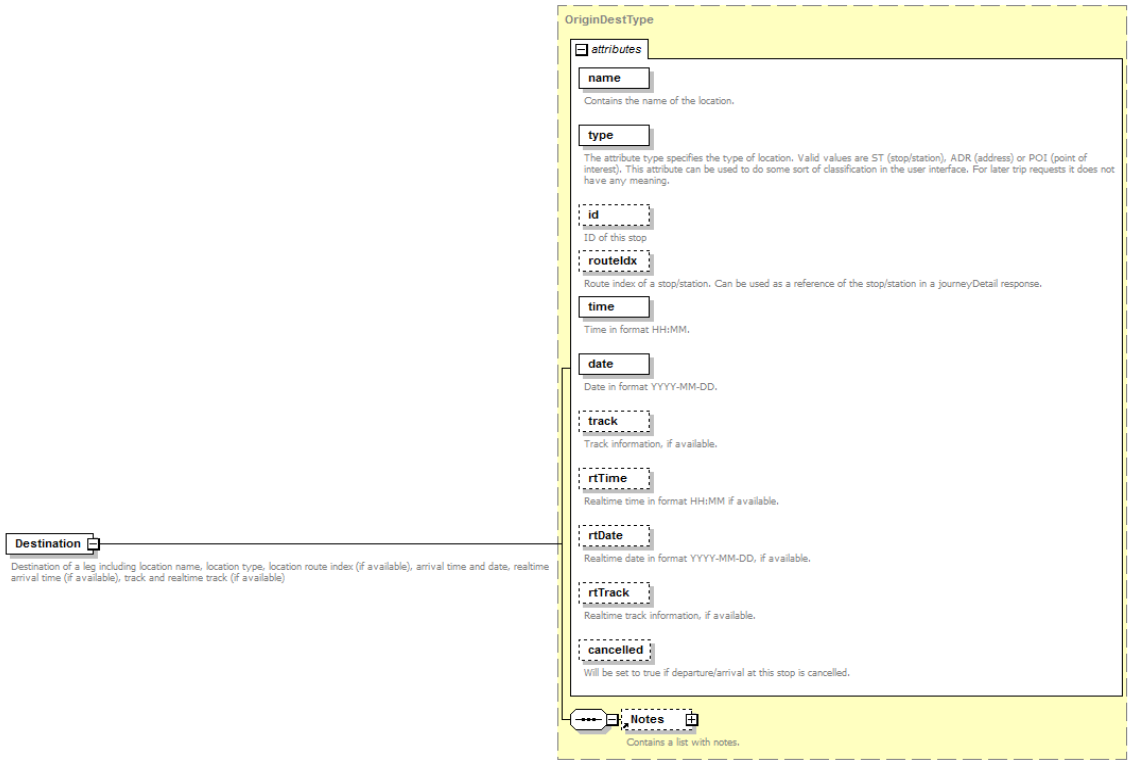
```
<LocationList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="..."
servertime="10:30" serverdate="2012-01-25" >
  <StopLocation name="Brunnered, Trollhättan" lon="12.401176" lat="58.262900" id="9021014081188000" />
  <StopLocation name="Brunnaliden, Trollhättan" lon="12.272810" lat="58.252095" id="9021014081871000" />
  <CoordLocation name="BRUNSOPPEVÄGEN, 461 54 TROLLHÄTTAN" lon="12.283085" lat="58.262469" type="ADR" />
  <!-- ... -->
</LocationList>
```


5.2 Trip Response

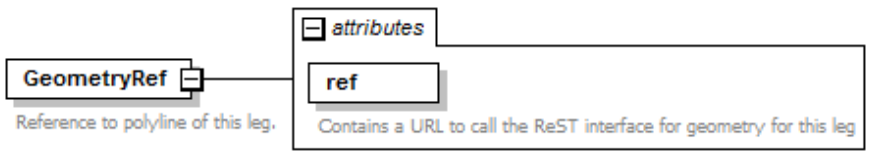
The trip response consists of a list of trips. Every trip has one to many legs with an origin and destination. The root element of the response is `TripList`.

Schema **hafasRestTrip.xsd**

element **Destination**

diagram	
used by	element **Leg**

element **GeometryRef**

diagram	
used by	element **Leg**

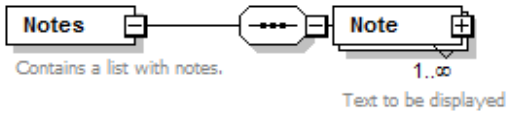
element **JourneyDetailRef**

diagram	
used by	element **Leg**

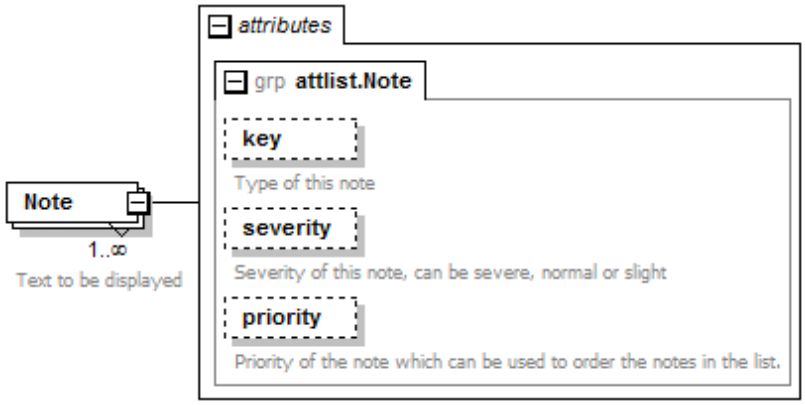
element **Leg**

diagram	
children	**Origin** **Destination** **Notes** **JourneyDetailRef** **GeometryRef**
used by	element **Trip**

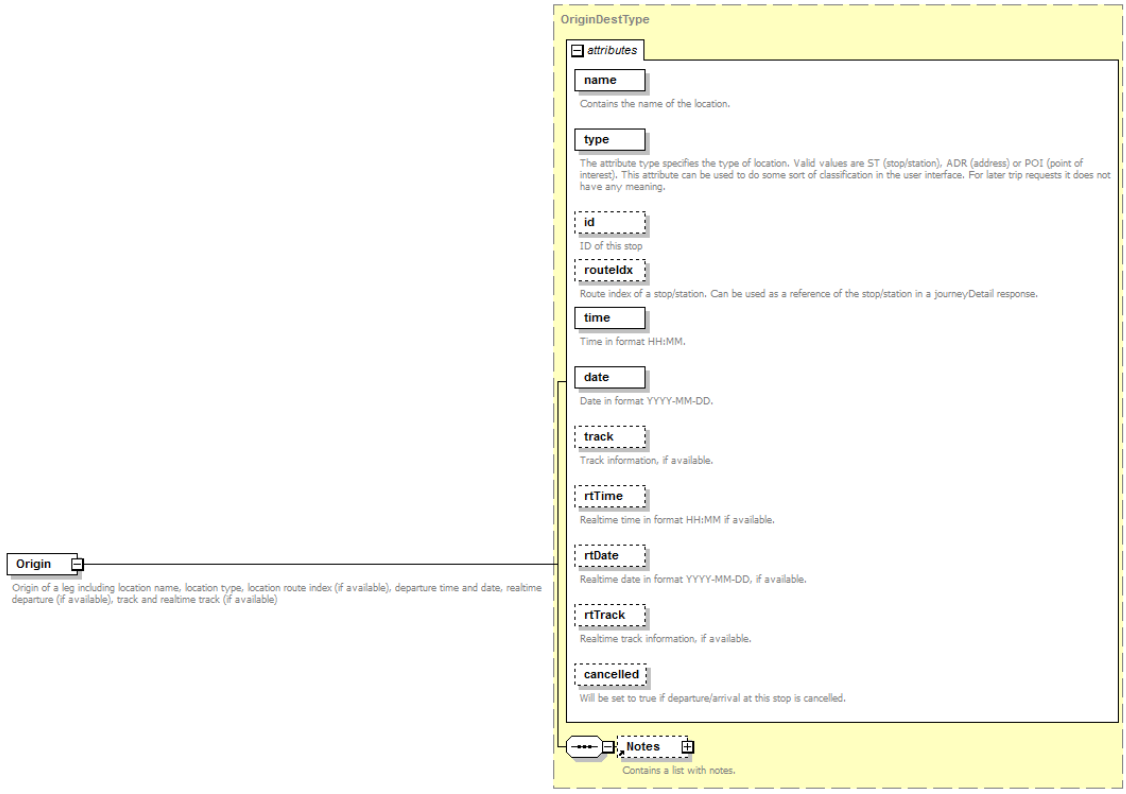
element **Notes**

diagram	
children	**Note**
used by	element **Leg**

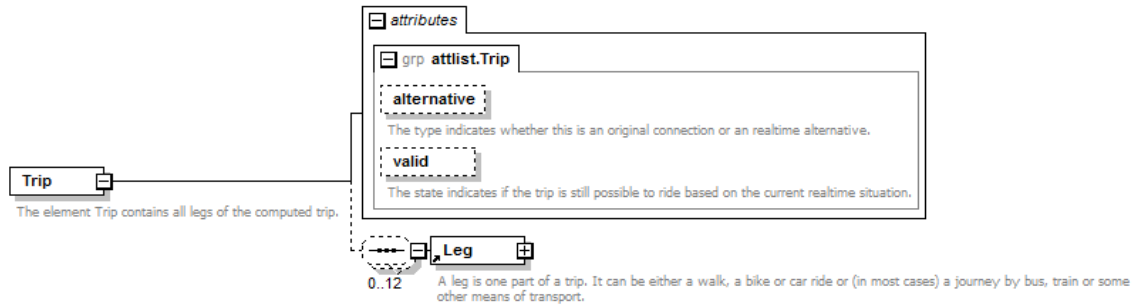
element **Notes/Note**

diagram	

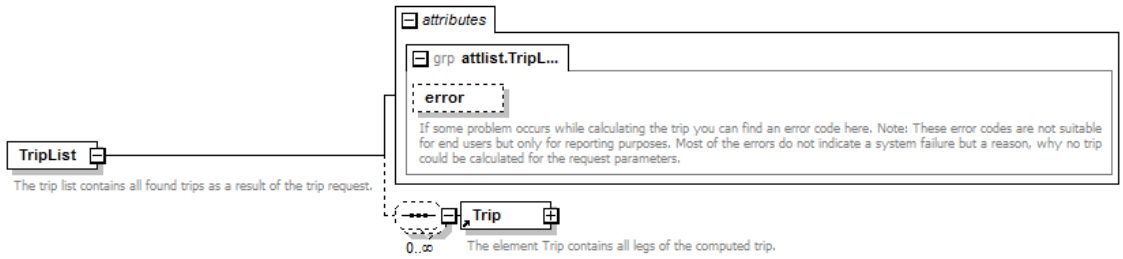
element **Origin**

diagram	
used by	element **Leg**

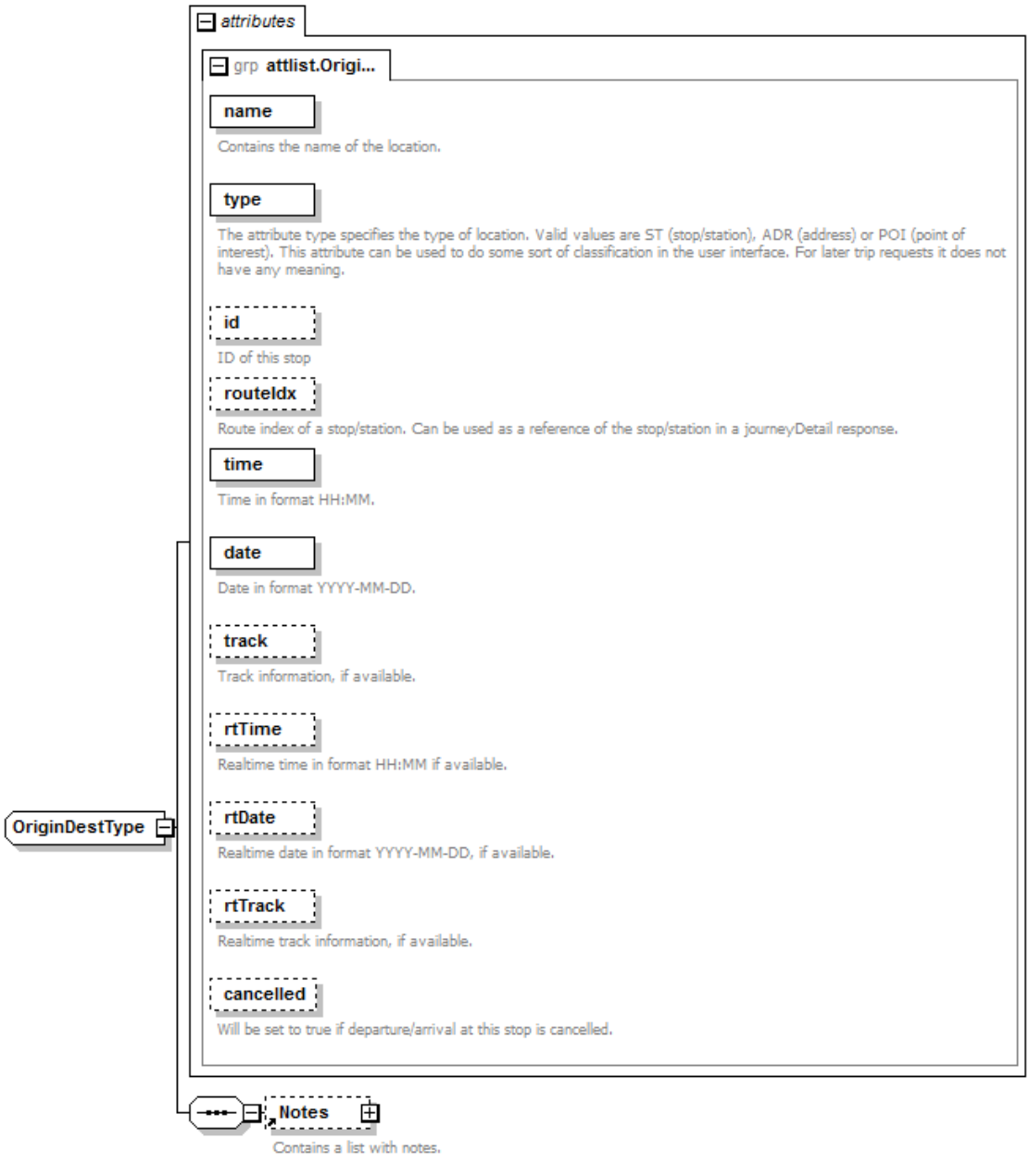
element **Trip**

diagram	
children	**Leg**
used by	element **TripList**

element **TripList**

diagram	
children	Trip

complexType **OriginDestType**

diagram	

used by	elements **Destination** **Origin**

Example Response

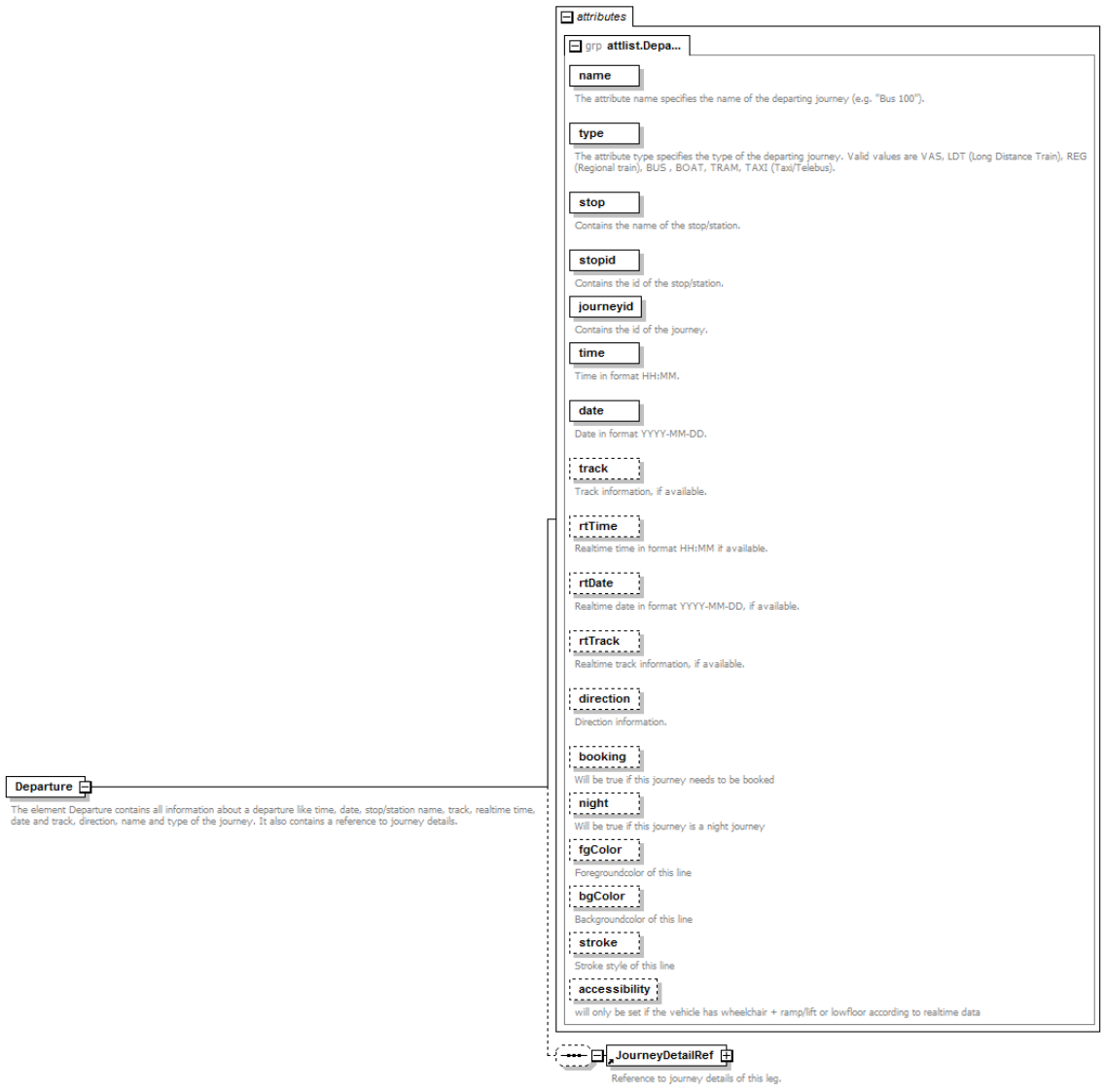
```
<TripList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="..." servertime="10:30"
serverdate="2012-01-25" >
   <Trip>
   <Leg name="Walk" type="WALK">
     <Origin name="Sahlgrenska Huvudentré, Göteborg" type="ST" id="9022014005600001" time="11:35" date="2011-09-16" />
     <Destination name="Sahlgrenska, Göteborg" type="ST" id="9022014005597001" time="11:40" date="2011-09-16" />
   </Leg>
   <Leg name="9 RONDEN" type="BUS" id="9015014890900002" direction="Åmål" booking="1" fgColor="#003273"
bgColor="#ffffff" stroke="Solid">
     <Origin name="Sahlgrenska, Göteborg" type="ST" id="9022014005597001" routeIdx="0" time="11:40" date="2011-09-16"
track="A " />
     <Destination name="NÄL, Trollhättan" type="ST" id="9022014081089003" routeIdx="17" time="13:25" date="2011-09-16"
track="C " />
     <Notes>
       <Note key="booking" severity="high" priority="1">Turen måste förbeställas på tel: 020-91 90 90 .|Ronden är avsedd för
sjukresor, särskild taxa gäller.</Note>
     </Notes>
     <JourneyDetailRef ref=" http://…" />
   </Leg>
  </Trip>
  </Trip>
```


5.3 Departure board response

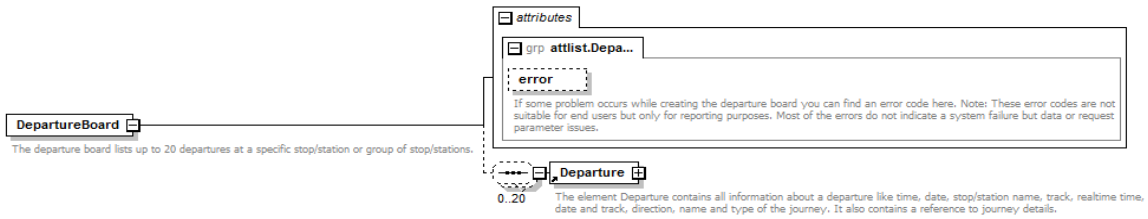
The departure board response contains a list of departures incl. all information concerning times, tracks, realtime data and journey. It also contains reference URLs to get more details for the different journeys. The root element is `DepartureBoard`.

Schema **hafasRestDepartureBoard.xsd**

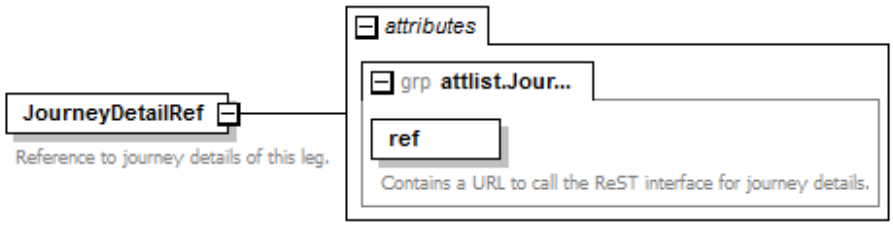
element **Departure**

diagram	
children	JourneyDetailRef
used by	element DepartureBoard

element **DepartureBoard**

diagram	
children	**Departure**

element **JourneyDetailRef**

diagram	
used by	element **Departure**

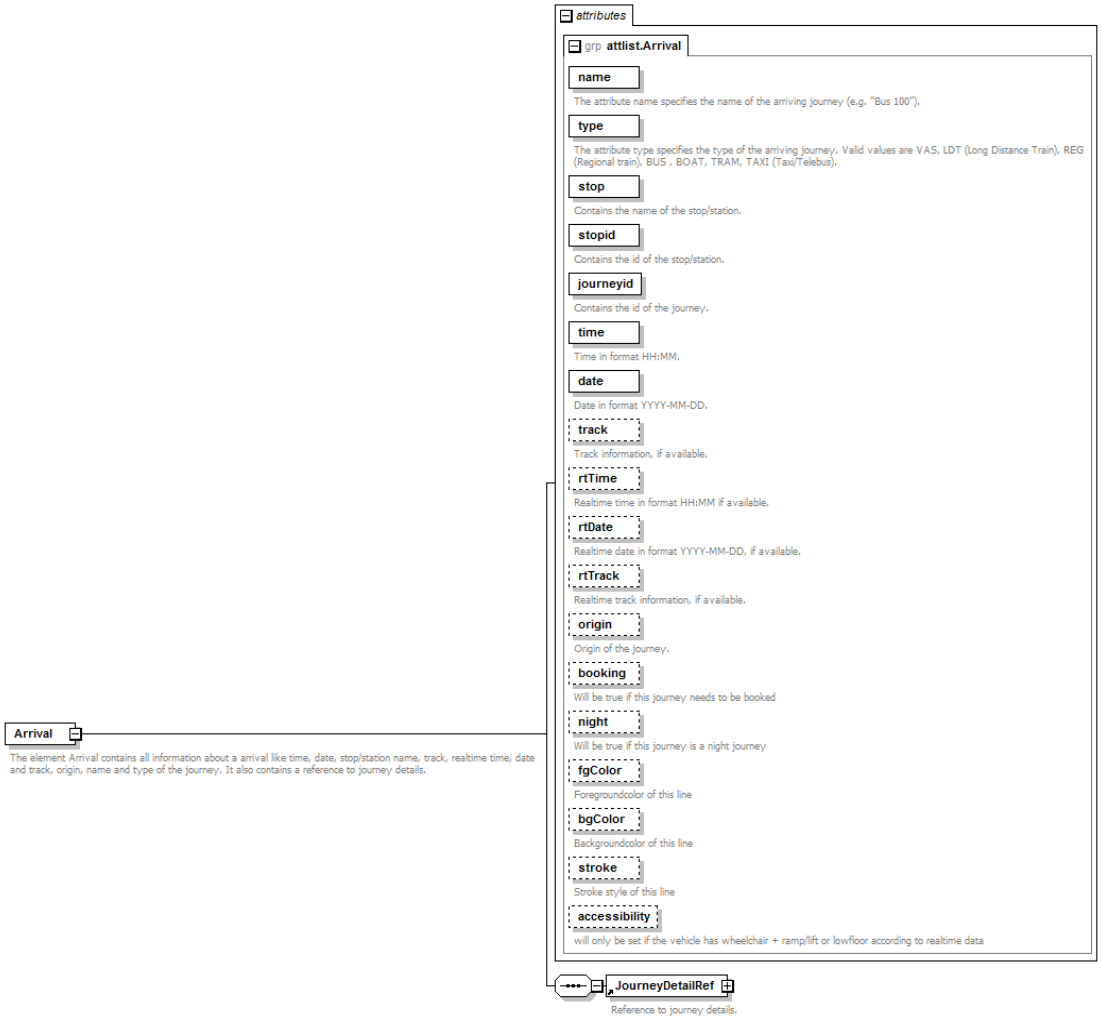
Example Response

```
<DepartureBoard xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="..."
servertime="10:30" serverdate="2012-01-25" >
  <Departure name="ÖRESUNDSTÅG" type="LDT" stopid="9022014008000001" stop="Göteborg C, Göteborg" time="14:42"
date="2011-09-16" journeyid="9015074110101087" direction="Malmö" track="1" fgColor="#003273" bgColor="#ffffff"
stroke="Solid">
    <JourneyDetailRef ref="http://…" />
  </Departure>
  <Departure name="VÄSTTÅGEN" type="VAS" stopid="9022014008000001" stop="Göteborg C, Göteborg" time="14:47"
date="2011-09-16" journeyid="9015014130103758" direction="Skee" track="1" fgColor="#003273" bgColor="#ffffff" stroke="Solid">
    <JourneyDetailRef ref=" http://…" />
  </Departure>
  </- ... ->
</DepartureBoard>
```

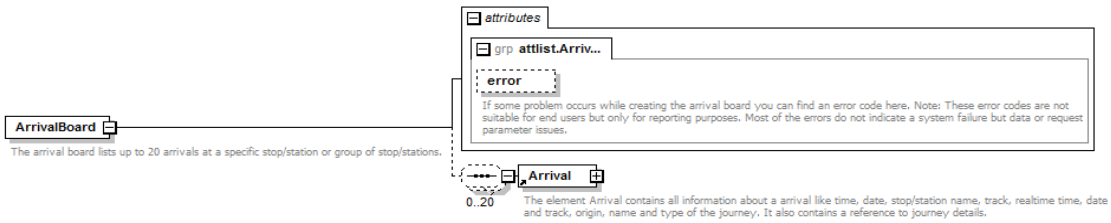

5.4 Arrival board response

The arrival board response contains a list of arrivals incl. all information concerning times, tracks, realtime data and journey. It also contains reference URLs to get more details for the different journeys. The root element is `ArrivalBoard`.

element **Arrival**

<table>
<tr><td>diagram</td><td> 
</td></tr>
<tr><td>children</td><td>JourneyDetailRef</td></tr>
<tr><td>used by</td><td>element ArrivalBoard</td></tr>
</table>

element **ArrivalBoard**

diagram	
children	**Arrival**

element **JourneyDetailRef**

diagram	
used by	element **Arrival**

Example Response

```
<ArrivalBoard xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="..."
servertime="10:30" serverdate="2012-01-25" >
  <Arrival name="PENDELTÅG" type="VAS" stopid="9022014008000001" stop="Göteborg C, Göteborg" time="14:42" date="2011-
09-16" journeyid="9015014131103553" origin="Alingsåsterminalen, Alingsås" track="1" fgColor="#003273" bgColor="#ffffff"
stroke="Solid">
    <JourneyDetailRef ref="http://..." />
  </Arrival>
  <Arrival name="PENDELTÅG" type="VAS" stopid="9022014008000014" stop="Göteborg C, Göteborg" time="14:42" date="2011-
09-16" journeyid="9015014132103056" origin="Kungsbacka station, Kungsbacka" track="14" fgColor="#003273" bgColor="#ffffff"
stroke="Solid">
    <JourneyDetailRef ref="http://..." />
  </Arrival>
  <Arrival name="VÄSTTÅGEN" type="VAS" stopid="9022014008000001" stop="Göteborg C, Göteborg" time="14:57" date="2011-
09-16" journeyid="9015014160208829" origin="Skövde Resecentrum Tåg, Skövde" track="1" fgColor="#003273" bgColor="#ffffff"
stroke="Solid">
    <JourneyDetailRef ref="http://..." />
  </Arrival>
<!-- ... -->
<ArrivalBoard/>
```


5.5 Journey detail response

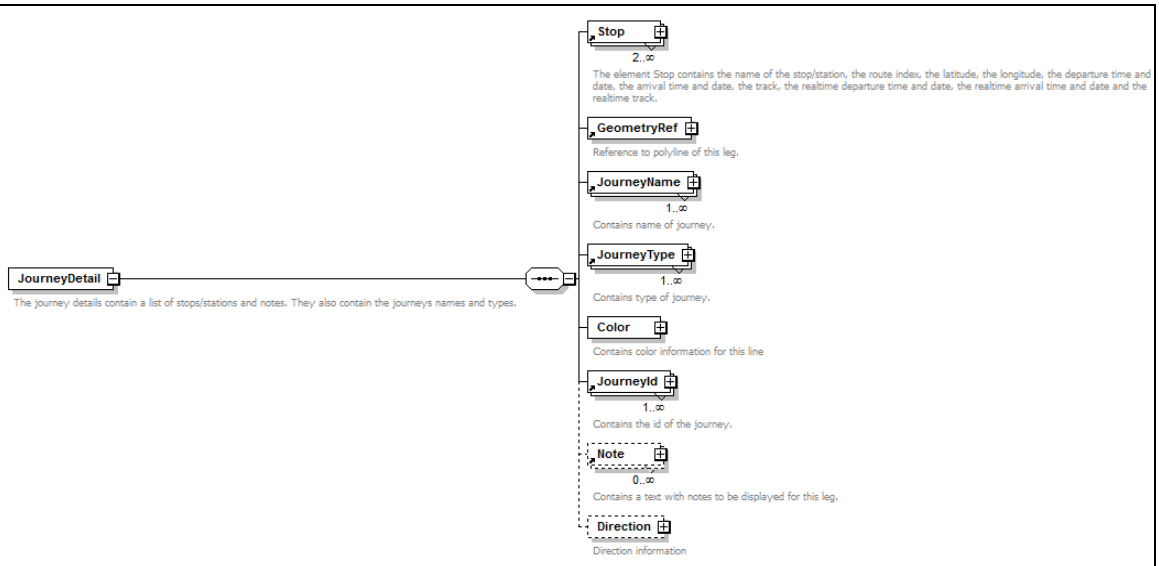
The journey detail response delivers all information about a single journey (vehicle route). It contains a list of stops including their indexes on the route and their coordinates. It contains also all times, tracks and realtime information if available for the whole route. It also contains the journeys name and type (there might be different names and types on parts of the journey). Finally it contains notes including information about their validity on segments of the total route.

Schema **hafasRestJourneyDetail.xsd**

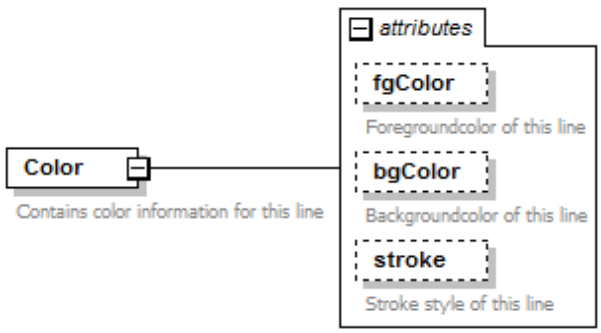
element **GeometryRef**

diagram	
used by	element **JourneyDetail**

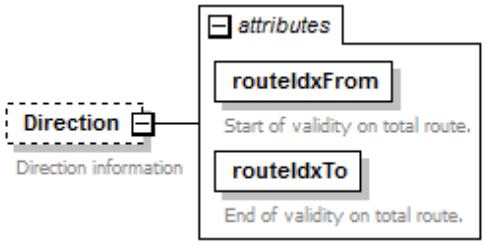
element **JourneyDetail**

diagram	
children	**Stop GeometryRef JourneyName JourneyType Color JourneyId Note Direction**

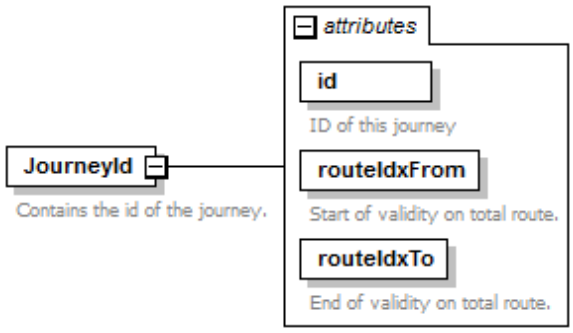
element **JourneyDetail/Color**

diagram	

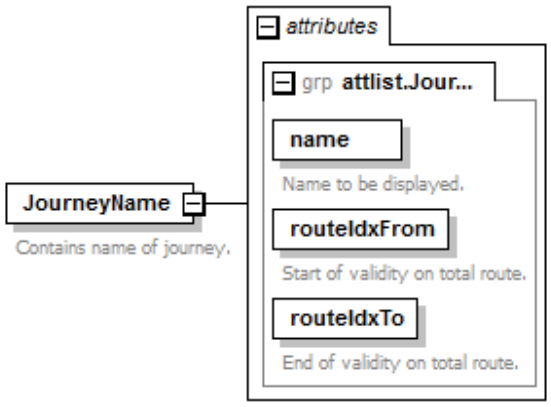
element **JourneyDetail/Direction**

diagram	

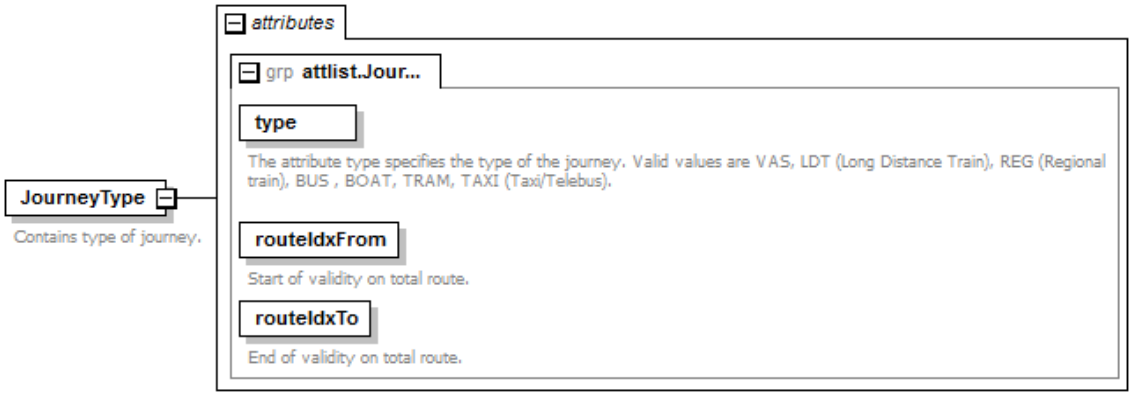
element **JourneyId**

diagram	
used by	element JourneyDetail

element **JourneyName**

diagram	
used by	element **JourneyDetail**

element **JourneyType**

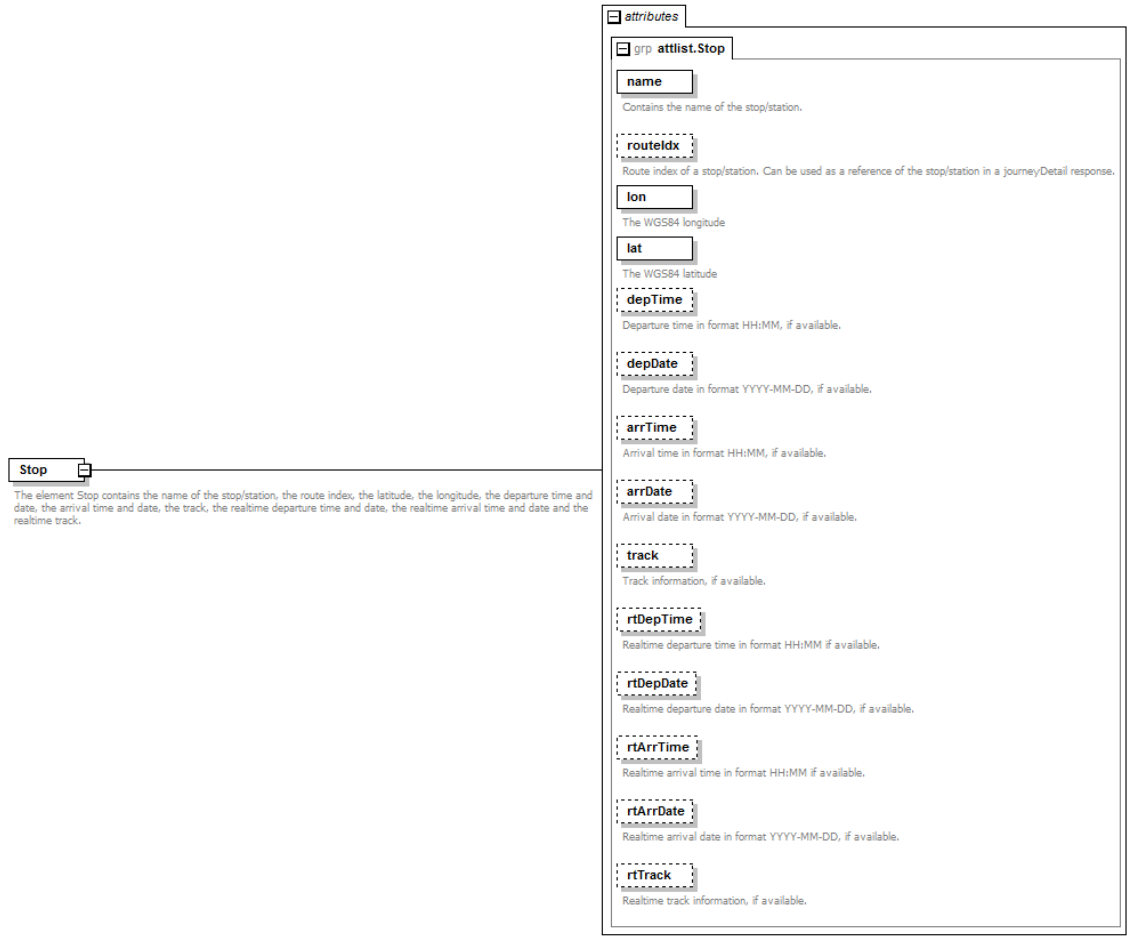
diagram	
used by	element **JourneyDetail**

element **Note**

diagram	
used by	element **JourneyDetail**

element **Stop**

diagram



used by element **JourneyDetail**

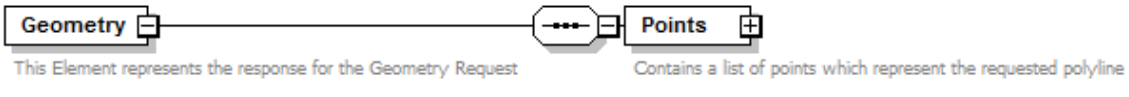
Example Response

```
<?xml version="1.0" encoding="iso-8859-1"?>
<JourneyDetail xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://..."
servertime="10:30" serverdate="2012-01-25" >
  <Stop name="Alingsåsterminalen, Alingsås" id="9022014017767013" lon="12.529866" lat="57.926991" routeIdx="0"
depTime="14:03" depDate="2011-09-16" track="O" />
</- ... ->
  <Stop name="Sävenäs Station, Göteborg" id="9022014006605001" lon="12.025337" lat="57.724734" routeIdx="10"
arrTime="14:36" arrDate="2011-09-16" depTime="14:36" depDate="2011-09-16" track="1" />
  <Stop name="Göteborg C, Göteborg" id="9022014008000001" lon="11.974503" lat="57.708859" routeIdx="11" arrTime="14:42"
arrDate="2011-09-16" track="1" />
  <Color fgColor="#003273" bgColor="#ffffff" stroke="Solid" />
  <GeometryRef ref="http://..." />
  <JourneyName name="Lok TÅG" routeIdxFrom="0" routeIdxTo="11" />
  <JourneyType type="VAS" routeIdxFrom="0" routeIdxTo="11" />
  <JourneyId id="9015014131103553" routeIdxFrom="0" routeIdxTo="11" />
  <Direction routeIdxFrom="0" routeIdxTo="11">Göteborg</Direction></JourneyDetail>
```


5.6 Geometry Response

The Geometry service will return the polyline for a leg. The result contains a list of WGS84 coordinates which can be used to display the polyline on a map.

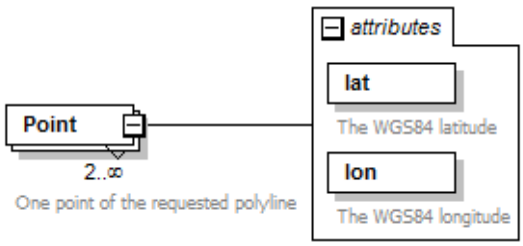
element **Geometry**

diagram	
children	**Points**

element **Geometry/Points**

diagram	
children	**Point**

element **Geometry/Points/Point**

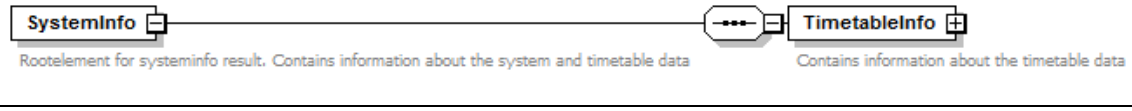
diagram	

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Geometry xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://..."
servertime="10:30" serverdate="2012-01-25" >
  <Points>
    <Point lat="57.795569" lon="12.049248" />
    <Point lat="57.795524" lon="12.049212" />
    ...
    <Point lat="57.680543" lon="11.901744" />
    <Point lat="57.680534" lon="11.901708" />
  </Points>
</Geometry>
```

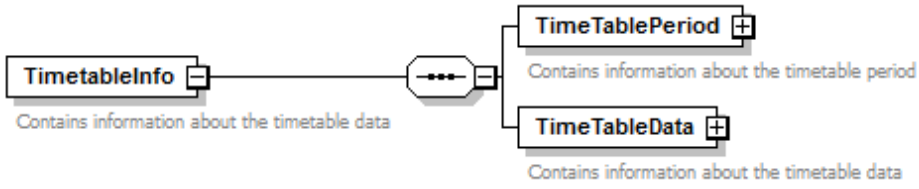

5.7 SystemInfo response

The SystemInfo response will contain information about the travelplanner system and the underlaying data.

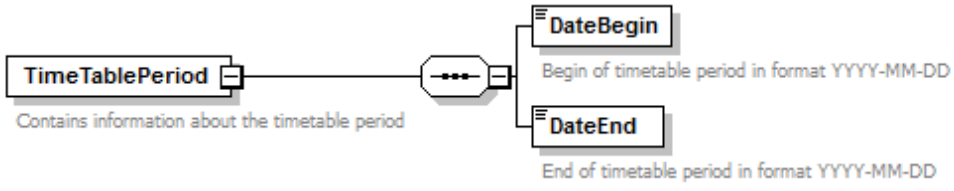
element **SystemInfo**

diagram	
children	TimetableInfo

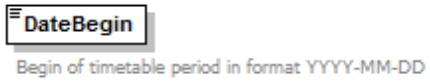
element **SystemInfo/TimetableInfo**

diagram	
children	TimeTablePeriod TimeTableData

element **SystemInfo/TimetableInfo/TimeTablePeriod**

diagram	
children	DateBegin DateEnd

element **SystemInfo/TimetableInfo/TimeTablePeriod/DateBegin**

diagram	

element **SystemInfo/TimetableInfo/TimeTablePeriod/DateEnd**

diagram	 End of timetable period in format YYYY-MM-DD

element **SystemInfo/TimetableInfo/TimeTableData**

diagram	 Contains information about the timetable data Creation date of timetable data in format YYYY-MM-DD
children	CreationDate

element **SystemInfo/TimetableInfo/TimeTableData/CreationDate**

diagram	 Creation date of timetable data in format YYYY-MM-DD

```
<?xml version="1.0" encoding="iso-8859-1"?>
<SystemInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://.../hafasRestSysteminfo.xsd">
  <TimetableInfo>
    <TimeTablePeriod>
      <DateBegin>2011-12-11</DateBegin>
      <DateEnd>2012-06-30</DateEnd>
    </TimeTablePeriod>
    <TimeTableData>
      <CreationDate>2012-05-16</CreationDate>
    </TimeTableData>
  </TimetableInfo>
</SystemInfo>
```

