VDV Die Verkehrs-unternehmen

VDV Recommendation 453

03/2017

Real data interface - version 2.5

CP	Connection protection
DPI	Dynamic passenger information
VIS	Visualisation
GM	General message service

Editing
Committee for Information Processing (AIV)

VDV Die Verkehrs-
unternehmen

VDV Die Verkehrs-unternehmen

Client

Federal Ministry for Traffic, Construction and Housing

Invalidenstrasse 44

D -10115 Berlin

Contractor

Beratungsgesellschaft für Leit-, Informations- +Computertechnik GmbH (BLIC)

Rheinstraße 45

D - 12161 Berlin

Editing:
Dipl.-Ing. Gustav Thiesing

The initial version of the interface was created with funding from BMVBW. This means that the specification is freely available.

The changes with regard to the previous version have been discussed in the 'Real-time Data Interface VDV Work Group' involving among others:

Dr. Christoph Blendinger	DB Mobility Logistics AG	Frankfurt am Main
Michael Beck	initplan GmbH	Karlsruhe
Baumann, Mathias	Interautomation	Berlin
Wolfgang Bibergeil	Funkwerk AG - Traffic & Control Communication	Kiel
Bratta, Tino	BLIC GmbH	Berlin
Volker Braun	IVU Traffic Technologies AG	Aachen
Eckardt, Frithjof	Eckardt Software Management GmbH	Hannover
Peter Elsensohn	Technische Informationssysteme GmbH	Rankweil
Wolfram Fiekert	HaCon Ingenieurgesellschaft mbH	Hannover
Michael Frankenberg	HaCon Ingenieurgesellschaft mbH	Hannover
Hesse, Roland	Mentz GmbH	Munich
Daniel Hollenstein	Swiss National Railway (SBB)	Bern
Hoppe, Klaus	Scheidt & Bachmann	Kiel
Isajkin, Waldemar	init	Karlsruhe
Kehren, Peter	IVU	Aachen
Kluge, Wolfgang	Dilax	Berlin
Werner Kohl	Mentz GmbH	Munich
Karl Horst Lenzen	T-Systems GEI GmbH	Mülheim
Stefan Lindenlaub	Swiss National Railway (SBB)	Bern
Günther Martinez-Dreyer	PSI Transcom GmbH	Berlin
Achim Müller	STAM	Buxtehude
Rittmeier, Raffael	Verkehrsverbund Bremen/Nieders. GmbH	Bremen
Daniel Rubli	Trapeze Switzerland GmbH	Neuhausen am Rheinfall
Siaden Ortega, Ute	Eckardt Software Management GmbH	Hannover
Wichtermann, Jürg	Swiss National Railway	Bern
Thiesing, Gustav	BLIC GmbH	Berlin
Zilgens, Wolfgang	Ingenieurgruppe IVV	Aachen

VDV Die Verkehrs-
unternehmen

History of changes from V 2.3.2 to V 2.5

VDV, Winfried Bruns

Position	Change	Author	Date
6.2.3.3 / 6.2.4.2.3 / 6.2.4.3.1 / 6.3.7.3 / 6.3.8.3.1 / 6.4.3.3.1	Definition of CPISchedule / FetcherInfo / CPIDeviation / DISSchedule / DISDeviation / VISDeviation: Description of DirectionText (RichtungsText) and FromDirectionText (VonRichtungsText).	SL/DHo RuD	12.12.2014 16.02.2015
6.2.3.3.1	Definition of TripInfo: New optional element of PatternID added.	RuD	16/02/2015
6.2.4.2.2	Time-based data (definition of TimeFilter): New optional element of PreviewTime added.	SL/DHo	12/12/2014
6.2.4.3.1	The 'ExpectedCPIArrivalTime' field has been changed from mandatory to optional.	SL/DHo	23/02/2015
	General layout of the tables and documents reworked	RuD	04/05/2015
5.1.2	SubscriptionReply (AboAntwort) expanded to include the alternative “AcknowledgeWithSubscriptionID” (BestaetigungMitAboID)	RuD / SL / DHo	26/05/2015
From here on, corrections from V 2.4 to V 2.4.1			
5.1.1	Communication progression diagram expanded to include ClientStatusRequest (ClientsStatusAnfrage) and ClientStatusReply (ClientsStatusAntwort)	DHo	10/02/2016
5.1.4.1	DataSupplyRequest (DatenAbrufenAnfrage): Description of sub-element “sender” corrected.	DRu	07/12/2015
5.1.8.2 / 5.1.8.3	StatusReply (StatusAntwort) / ClientStatusRequest (ClientStatusAnfrage): DataVersionID (DatenVersionID) changed from mandatory to optional.	WK, SL, DHo	10/02/2016
From here on, corrections from V 2.4.1 to V 2.5			
0	New chapter, ‘Using AllData (DatensatzAlle)’, added (CR_0010). According to a discovery by Roland Hesse, the item “DIS/VISTripDelete (AZV/VIS-FahrtLoeschen) for departed/ended trips for which the ValidUntilTimeStamp (VerfallZst) of the last DIS/VISDeviation (AZB/VISFahrplanlage) has not yet expired (unless otherwise agreed)” was deleted.	DRu DRu	28/10/2016 05/03/2017
5.1.4.2.2	New chapter ‘Expansions to replace TripDelete (FahrtLoeschen)’ added (CR_0040).	DRu	28/10/2016
06/01/2012	Central declaration for publication-relevant texts added to avoid the need to indicate publication relevance for the applicable elements. (Derived from CR_0015-0018)	SL	14/06/2016

6.2.4.2 / 6.3.8.2 / 6.4.3.2	New, optional element “TripDelete-ThroughDeviation” (FahrtLoeschen-DurchFahrplanlage) added for CPISubscription/DISSubscription/VISSubscription (AboASB/AboAZB/AboVIS) (CR_0040).	DRu	28/10/2016
6.2.4.3.1 / 6.3.8.3.1 / 6.4.3.3.1	Description of TripStatus (FahrtStatus) expanded to include “TripDelete” (FahrtLoeschen) and “TripCancelled” (FaelltAus) (CR_0040).	DRu	28/10/2016
6.2.4.3.2 / 6.3.8.3.5 / 6.4.3.3.2	CPI/DIS/VISTripDelete (ASB/AZB/VIS-FahrtLoeschen) chapter supplemented with the note that, in the case of a subscription with TripDeleteThroughDeviation (FahrtLoeschen-DurchFahrplanlage) = “true”, the TripDelete (FahrtLoeschen) should no longer be used (CR_0040).	DRu	28/10/2016
6.2.3.3 / 6.2.4.3.1	New optional element “ArrivalPlatformText” (AnkunftssteigText) added for CPISchedule (ASBFahrplan) and CPIDeviation (ASBFahrplanlage) (CR_0018). New optional element “ArrivalSectorText” (AnkunftsSektorenText) added for CPISchedule (ASBFahrplan) and CPIDeviation (ASBFahrplanlage) (CR_0015)	SL	14/06/2016
6.2.4.2.3	New operational element “DeparturePlatformText” (AbfahrtssteigText) added to FetcherInfo (AbbringerInfo) (CR_0018) New optional element “DepartureSectorText” (AbfahrtsSektorenText) added for FetcherInfo (AbbringerInfo) (CR_0015)	SL	14/06/2016
6.3.7.3 / 6.3.8.3.1	New optional elements “ArrivalPlatformText (AnkunftssteigText)” and “DeparturePlatformText (AbfahrtssteigText)” added for DISSchedule (AZBFahrplan) and DISDeviation (AZBFahrplanlage) (CR_0018). New optional elements “ArrivalSectorText (AnkunftsSektorenText)” and “DepartureSectorText (AbfahrtsSektorenText)” added for DISSchedule (AZBFahrplan) and DISDeviation (AZBFahrplanlage) (CR_0015). New optional, multiple element “TripNameText” (FahrtBezeichnerText) added for DISSchedule (AZBFahrplan) and DISDeviation (AZBFahrplanlage) (CR_0016). New optional elements “NoBoarding” (Einsteigeverbot), “NoAlighting” (Aussteigeverbot) and “PassThru” (Durchfarht) added for DISSchedule (AZBFahrplan) and DISDeviation (AZBFahrplanlage) (CR_0017).	SL	14/06/2016
6.2.4.2 / 6.3.8.2 / 6.4.3.2	Changes by Dru (28/10/16) withdrawn	RH	14/11/2016
6.2.4.3.2 / 6.3.8.3.5 / 6.4.3.3.2	Changes by Dru (28/10/16) withdrawn	RH	14/11/2016

6.2.4.3.2 / 6.3.8.3.5 / 6.4.3.3.2	Note, that optional elements in TripDelete (FahrtLoeschen) must be filled out if possible	RH	14/11/2016
6.3.8.2 5.1.2.1	New operational element ‘OnlyUpdate (NurAktu-alisierung)’ added to DISSubscription (CR_0038). Note added indicating that general data must be sent after a subscription update.	DRu	15/12/2016

1 Foreword

The VDV Recommendation entitled 'Integration Interface for Automatic Vehicle Location and Control Systems' has been drawn up within the framework of the (FOPS) 70.0701/2002 City Traffic Research Project of the Federal Ministry of Traffic, Construction and Housing (BMVBW), conducted by BLIC and IAV under the technical guidance of the VDV. Various other partners from the AVLC industry along with some major transport authorities were also involved with the specification. The 'Integration Interface for Automatic Vehicle Location and Control Systems' Recommendation establishes common requirements on the design and structure of the interfaces between AVLC systems. In addition to the principal and functional requirements, the requirements on the data exchange in particular are outlined in detail.

This general specification document should promote consistency, modularity and manageability of the software and hardware for the purpose of comprehensive data exchange.

The motivation for the extension of version 1.0 of March 2001 to version 2.0 stems from the experiences gained during implementation. The interface has been technically revised, restructured and supplemented with new dedicated services. A common technical communications infrastructure has been created for all dedicated services, which allows any implementation work or extensions to be carried out more quickly and at less cost when compared with version 1.0. The new architecture is open to further services.

The following services are specified:

Service	Purpose
Reference data service for connection protection (REF-CP)	Exchange of planned timetables for connection protection
Process data service for connection protection (CP)	Exchange of actual data for connection protection
Reference data service for passenger information (REF-DPI)	Exchange of location-related planned timetables for passenger information
Process data service for passenger information (DPI)	Exchange of actual data for passenger information
Process data service for visualisation (VIS)	Exchange of actual data for the visualisation of vehicles in third-party control centres
General message service (GMS)	Exchange of written information between the control centres

In this version 2.4.1, various errors have been corrected and the functionality partially extended. Furthermore, implementation tips have been added, which do not refer directly to the interface specification but are included to clarify use of the interface as well as application and interpretation of the transmitted data.

The associated XML structure definition has the file name: VDV453_incl_454_V2017.a.xsd

The latest XML schema (see the VDV Internet site www.vdv.de/i-d-s-downloads.aspx), acts as a reference for the implementation of services and in addition to the services mentioned above also contains the REF-SIS and SIS timetable information reference and process data services.

The respective text line represents an explanation of the given application.

The CEN TS 00278181 SIRI standard provides an extended functionality for the individual services, which has been developed from the VDV Recommendation. With regard to requirements on the data exchange, which exceed the specification of the given recommendation, it is necessary to check whether these requirements are satisfied by the CEN standard. If so, an interface should be created on the basis of the SIRI standard rather than a proprietary solution.

2 Introduction

2.1 General

The aim of the 'Integration Interface for Automatic Vehicle Location & Control Systems' project is to implement an interface that will link the various AVLC systems of different transport authorities and possibly also different manufacturers. In cities and particularly in urban agglomerations many transport authorities have installed AVLC systems in recent years in order to optimise and improve their operational processes. The manufacturers of such systems often follow different strategies with regard to the hardware and software, which creates a very different system architecture. The problem is that the functional boundaries and system resources of the individual AVLC systems are not prepared for data matching processes with other transport authorities. There are system limits between the individual AVLC systems, which should not be noticeable to the end user, i.e. the passenger. The passenger simply demands a consistently high standard of quality and a functioning transport service. For the realisation of connection protection between the routes of neighbouring transport authorities for example, it is necessary to coordinate the operational processes at the corresponding interfaces of the individual systems. System variety and individual design of the individual AVLC set-ups do mean there is a danger of escalating costs for each individual system coupling. This recommendation follows the approach of creating a universal mechanism with a universal interface for the integration of AVLC systems, which allows the individual operators to implement such functionality at acceptable costs, with regard to procurement and operation.

The technical side of the concept is based on standard technologies (HTTP/XML). These carry out the essential tasks of an interface. On the one hand they reduce the cost of implementation and on the other ensure the necessary compatibility for the implementations of various manufacturers.

Transferability – even to small AVLC systems – is guaranteed, as both the applied technology and the volume of exchange data take up little resources.

The following have been identified as essential “soft” requirements on the inter-operational organisation of the transport authority, which would be introduced by such a concept:

- Inter-operational agreement
- Inter-operational commercial reconciliation
- As well as an agreed (partial) database used universally by all participant operations

2.2 Objectives

There are additional requirements for control system set-ups, in that the operator must be able to integrate such a system into the existing technical operational management configurations. At the same time however, the system concept must be designed so as to ensure the latest status of the technology can be taken into consideration at all times. In view of the further

development of operational control technology it should also be remembered that the automation of operational management is an essential requirement with regard to the efficient and streamlined operation of transport networks. Increasingly, transport authorities are striving towards the aim of using the operational management and the directly associated procedures to organise the processes of preparation, operational execution and post-processing so that they are fully automatic and largely free of any user intervention. Added to that, there is an increasing interlinking of organisational and operational installations.

Furthermore, an essential conceptual aim in connection with the introduction of new technologies is in the improvement of the associated efficiency indicators for procurement, operation and maintenance.

2.3 Mission statement

As a result of their complexity and high investment costs, the technical operational management tools of transport authorities are mainly characterised by a long life. Incremental renewals as well as subsequent network extensions have inevitably lead to the current situation in which different technical development generations are used alongside one another in the same operation. Particularly for systems that have not yet been contracted there is a demand for future-safe operation and with that compatibility with new systems.

For this reason, it must be possible to integrate any new control technology installations into the existing configuration of the technical systems and they must support the existing operational procedures. Of further important significance is an approach that follows the principle of "least possible expenditure with the greatest possible benefit", as it represents the basis for effectiveness and efficiency and simultaneously reflects the company vision of many transport authorities.

In view of the progressive level of automation, it is essential that the mission statement corresponds to the known and associated requirements. In this context, a modular design of the technical solution and the possibility of a step-by-step implementation of the intended aims are also inevitable.

3 Introduction and basic terms

3.1 Connection Protection (CP)

3.1.1 Tasks and Targets

Any direct processing of all desired journey combinations is practically impossible in larger cities and urban regions and totally impractical in terms of efficient operational management. Within the scope of planning their services, networks and journeys, a transport authority must consider the prevailing traffic streams and connection possibilities between the individual routes in the planned timetable within the context of connection planning.

The basic problem when it comes to planning connections, is that for the passenger on a 'split journey', i.e. one which involves a change in vehicles, the interchange times must represent a certain level of convenience or acceptance. It is therefore necessary to minimise the waiting times when planning connections within trips. If the changeover times are too long there is a danger of the connection being rejected by the passenger. Furthermore, when defining a connection it is necessary to consider the constraints imposed by the locality and timings (connection times between the arrival stop and the new departure stop) as well as the operational and general specifications that are valid for specific time periods and the majority of all connections.

Although this problem can be adequately managed where there are simply two routes and one connection interchange, it leads to a situation when dealing with numerous connection interchanges and multiple routes where only a few selected connection relationships can be reasonably considered as scheduled connections within the scheduled timetable.

Two run vehicles are always required for a connection: The 'feeder' and the 'fetcher'.

3.1.2 The feeder - fetcher principle

The internal connection (connection) protection is already an established dispatch function. By comparing at least two timetable statuses at the connection stop for a specific passenger connection time required by the passengers, the fetcher vehicle can be instructed to wait for the feeder vehicle at a defined time.

The intended solution for implementing connection protection follows the feeder - fetcher principle. In this context, two vehicles are involved in one connection relationship. By the mutual referencing of feeder-fetcher relationships (i.e. the feeder is simultaneously the fetcher) it is possible to implement multiple and group connections.

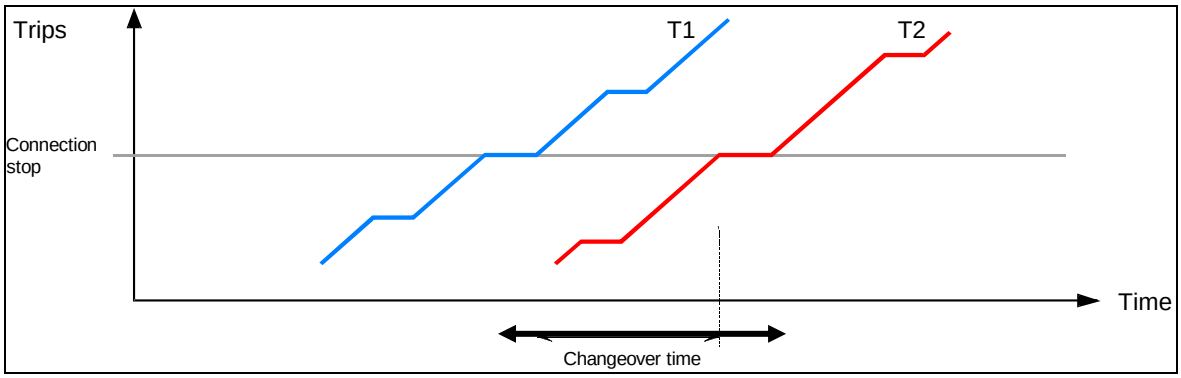

Fig. 3-1: Feeder-Fetcher connection

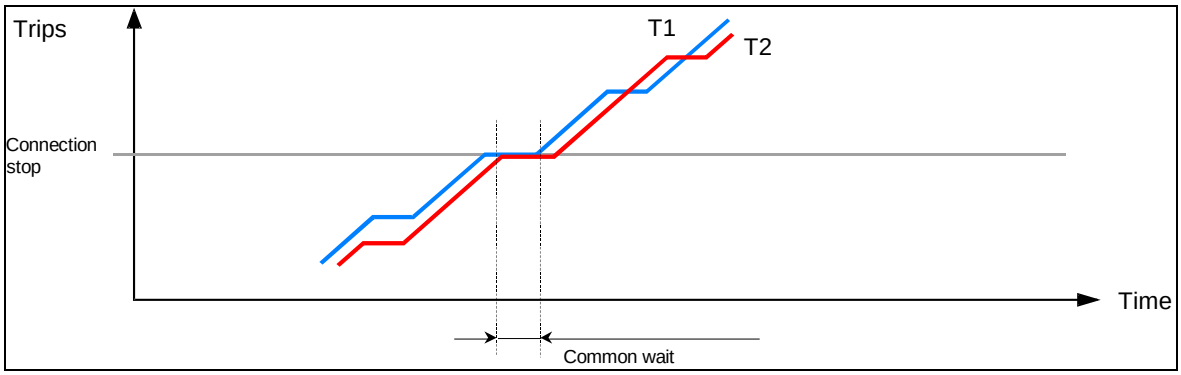

Fig. 3-2: Group connection

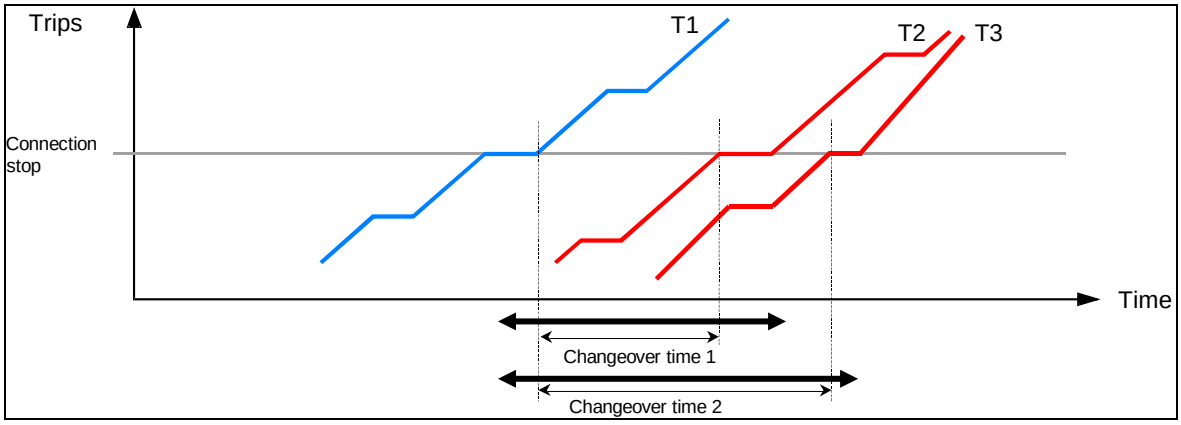

Fig. 3-3: Multiple connection

3.1.3 Definition of inter-operational connection protection

As with internal connection protection, the inter-operational connection protection should function as automatically as possible by means of mutual data exchange between the dispatch systems of the transport agencies and only prompt for dispatcher intervention in case of doubt (violation of a limiting condition). The basic requirement for automated connection protection is as follows - the AVLC software which controls the fetcher vehicles (e.g. sends instructions to wait for a delayed feeder) must contain the necessary dispatch tools and functions. The feeder AVLC does not require this functionality, unless it is responsible for fetcher functions in the opposite direction.

The matching process between the transport agencies must include a common, bilateral definition of the connections.

3.1.4 Operational models

3.1.4.1 Stations

Stations generally comprise multiple platforms. Defining the entire station as a stop means that it is then impossible to separate changeover times from a platform as opposed to a bus stop in the station foyer. It is preferable to define the individual platforms as stops. The problem here, however, is the fact that trains are often diverted to different arrival platforms which means they never actually arrive at the pre-specified connection area. This requires corresponding programming in the feeder system, which is able to represent and manage any platform changes for the fetcher vehicles. The feeder system therefore continues to report predictions to the fetcher for the connection area that was originally planned, even though the feeder does not arrive at the specified connection protection area. This is resolved internally by mapping the 'old' feeder stop/platform to the new stop/platform.

3.1.4.2 Multiple connections

Many AVLC systems permit the definition of so-called block or system connections (n:m relations). These involve several trips and multiple stops, each of which have different feeder and fetcher functionalities.

With the trip-based connection protection (see 3.1.8), these connections must be broken down into individual connections (1:1 relations). Each fetcher is assigned a single feeder.

With the time-based connection protection (see 3.1.9) a fetcher can have several feeders (n:1 relation).

3.1.4.3 Multiple served stops

If a stop is served more than once during a trip, location and trip label are no longer sufficient to uniquely identify an arrival. For this reason an additional counter is implemented. Arrival at a stop has a lower counter value than the next arrival at the same stop. The counter need not

increase sequentially. Either the connection time (coded as a whole number) or the stop index can be used directly.

3.1.5 Journey and connection planning (planned timetable)

This recommendation describes the technical interface of the connection protection functionality. This technical implementation does not replace the inter-operational requirements that must be agreed in advance nor any cross-operational agreements between the various transport authorities. All operational and traffic related restrictions and conditions must be clarified prior to the planning process and implementation of the connection protection.

Using the created timetables it is possible to identify possible connection points, which are then defined as general internal and cross-operational parameters.

It is necessary to record the participating routes and their functions (feeder or fetcher vehicle).

The next step is to define the time frame (e.g. via time points or trip names), in which the connection is to be protected. At the same time, there is a restriction to the most important connection times from the point of view of the passenger, i.e. early morning and late evening or periods with longer headway times. In this connection it is necessary to consider what happens when a connection is upheld, with the risk of endangering possible follow-on connections.

In accordance with the local conditions, it is necessary to record the passenger connection times between the relevant stops.

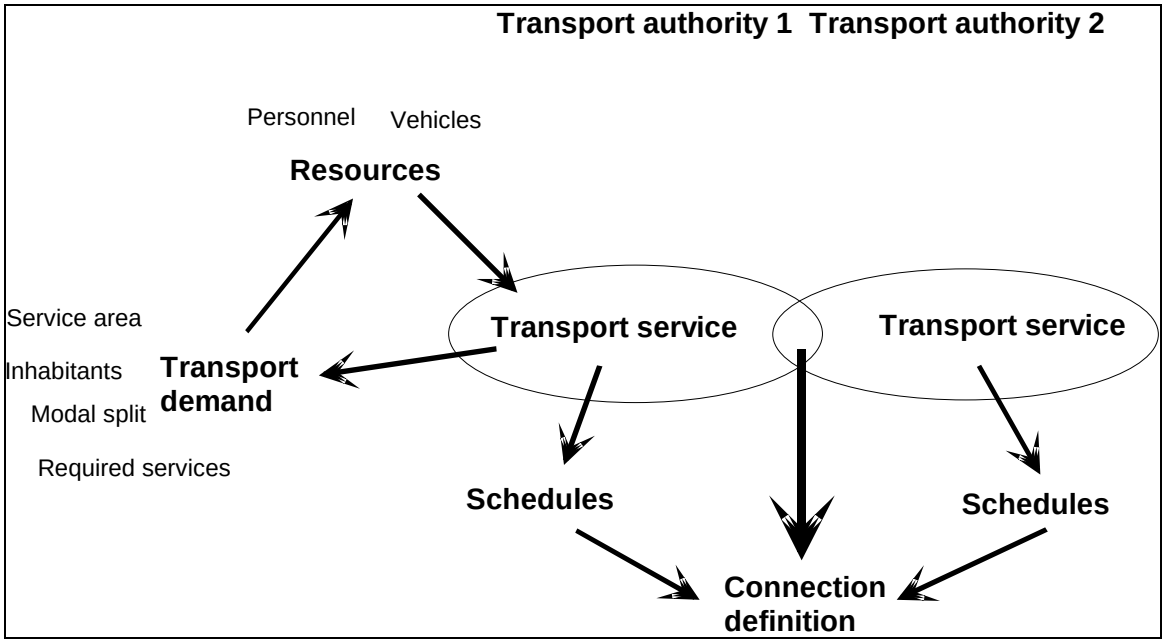

Fig. 3-4: Planned timetable and connection definition

3.1.6 Connection areas

Connection areas are used to create connection relationships. They replace the direct reproduction of the internal stop names of the respective systems. This unlinks the data management of the systems. It also negates the significance of the internal key. In this way it is possible to directly link entire stop areas of two agencies within one connection relationship.

In its simplest configuration, a connection area consists of exactly two stops - a feeder stop and a fetcher stop. By definition the interchange time is zero.

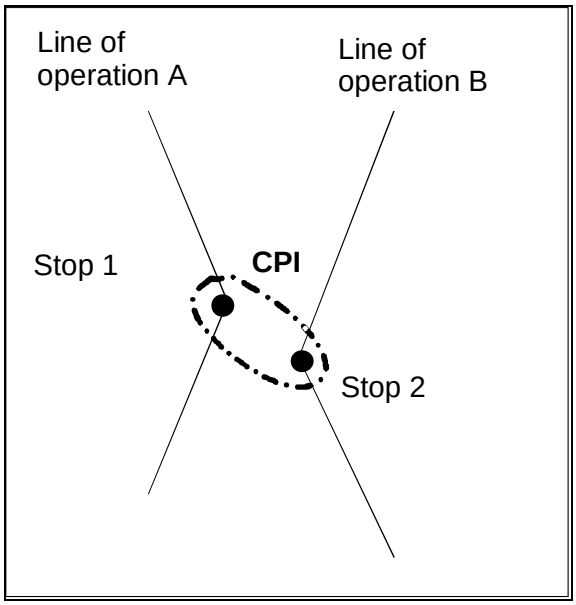

Fig. 3-5: Connection area (CPI)

If interchange times are to be defined, then on the fetcher side the representation of the local stop within the connection area must be extended with the interchange time. The fetcher system must then add the interchange time to the predicted arrival time. It can then inform the passengers waiting at the fetcher stop of the actual arrival time. Several fetcher stops with different interchange times can be assigned to a connection area. In the relevant operational data management, the interchange times are attributes for assigning connection area to stop. Interchange times are only defined on the fetcher side.

The local conditions often allow several feeder and/or fetcher stops to be assigned to one connection area. This saves time and resources when managing the operational data. In order for this to be possible however, all feeder stops that are assigned to a connection area must have the same interchange time to a fetcher stop that is also assigned to the connection area. This must be true for all fetcher stops.

The following table shows the interchange times of a typical interchange situation (with three feeder and three fetcher stops). As columns F1 and F2 are identical, both feeder stops can be assigned to the connection area. Any unoccupied fields (those without connections) act as placeholders, i.e. they represent 'wild cards'.

Stops	F1	Z2	Z3
A1	1 min	1 min	1 min
R2	2 mins	2 mins	4 mins
R3	1 min	No connection	7 mins

Table 1 - Matrix of interchange times for determining connection area

In the example here, this results in not nine (number of feeder stops multiplied by the number of fetcher stops) but only two:

CPI1: Feeder F1/F2, fetcher: D1/D2/D3

CPI2: Feeder F3, fetcher: D1/D2/D3

The data management only defines where a connection relationship is required, i.e. between which stops, and the criteria on which this connection relationship depends. Connection relationships are generally defined specific to route and direction (e.g. from route 10, direction X to route 1, direction Y). In addition, connection relationships are often only monitored at specific times of the day. The assignment tables of the connection areas must be extended accordingly.

3.1.7 Passenger information on the interior signs

During the trip in the direction of a connection area, all relevant information concerning the status of the pending connection relationships is collected in the vehicle. This information should be stored with easy access so that if necessary it can be released on demand. This information is provided both visually and acoustically at a suitable point before reaching the connection area.

3.1.8 Trip-based connection protection

The trip-specific connection protection corresponds to the approach of the connection protection function according to VDV453 Version 1.0. Here, planned timetables (reference data) are exchanged at the start of the operating day. In the subsequent phase of the process data exchange, only the actual delay statuses are communicated.

Above VDV453 Version 2.0, updates relating to the planning data are also possible. This means that extra trips can be reported as they happen.

The incorporation of severely delayed trips (as a replacement for follow-on trips) remains impossible.

The advantage of this procedure is the early definition of the connections. This means the passengers can be informed in advance of the actual connections taking place.

3.1.9 Time-based connection protection

Compared with Version 1.0, the time-based connection protection is a new method of connection protection. It allows connections to be protected without any previous exchange of reference data. The request is sent to the feeder system when the fetcher is actually approaching

the connection area. The request defines a place and time period, which restricts the number of trips on the basis of their predicted arrival time in the connection area. Route and direction filters can be used to further restrict the data volume.

The time-related connection protection can therefore cover all manual interactions, which can lead to a change in the departure board at the feeder stop. Even severely delayed vehicles can become feeders.

A possible disadvantage is the later notification of the connection definition. With suitable definitions in the data management however, an equivalent level of forward planning can be achieved in the definition of the connections.

3.2 Dynamic passenger information (DPI)

3.2.1 Tasks and targets

The aim of the 'dynamic passenger information (DPI) service' is to create the possibility of showing third-party trips at the local, commonly served stops. Data exchange is location based, i.e. departure boards for previously defined locations are transmitted.

A distinction must be made here between passenger information in the vehicle and that at the stop. Connection information in the feeder vehicles is implemented within the connection protection service. Visual and dynamic passenger information at stops has since become an essential element of automated control systems.

The DPI service is described with the following properties:

- → Individual predictions for every DPI sign (location-based predictions).
- → Quick and regular cleardown (via radio or AVLC message)
- → Start stop / destination / 'via' texts (codes and full texts)
- → Trip-specific texts
- → Special route texts
- → Text length specifications
- → Preview times for the necessary communication

3.2.2 Data supply and control

The DPI service address the common use of DPI signs. A third-party company wishes to display the arrival/departure of a vehicle on a DPI sign of a different transport authority. The system that controls the DPI signs is called the 'display owner' system. The system that wishes to show a vehicle on a third-party DPI sign is called the 'display user' system.

Two different procedures are covered or specified by the DPI service:

- → Controlling individual DPI signs with full text information
- → Controlling individual DPI signs with coded values in place of full texts

The full text method represents the simplest possibility for supplying the DPI signs with data. It is particularly suitable for situations where few DPI signs are being controlled and the available transmission capacity between the AVLC and the DPI sign allows this.

The version based on a coded-supply can be used where the transmission capacity between the AVLC and the DPI signs is restricted. Using codes significantly reduces the required capacity.

In its simplest form the passenger information does not require the exchange of reference data. It is only necessary to exchange data in real time. The only prerequisite is a list of display areas, which allows the AVLC to determine where the 'third-party' DPI signs required to show the information are located. Some systems however still require access to the planning data in order to be able to process third-party trips accordingly. For these systems there is an additional reference data interface, which provides the necessary timetable information.

In accordance with the principles behind the system, the 'display owner' carries the responsibility for what information is displayed. The third-party AVLC system provides corresponding display area information comprising the following specifications:

- → Trip ID
- → Arrival time
- → Departure time (optional)
- → Route and direction text
- → Stopping position (e.g. platform 5, bay C, etc.)
- → Trip destination
- → Via stop details
- → Comments
- → Special route texts
- → Trip-specific texts

Controlling display systems using codes demands a common operational data supply. (Third-party) texts and corresponding codes must be made known to the display owner system. Full texts are transmitted via the interface, as was previously the case. Through the availability of tables on the display owner side, a text comparison can be used to determine the associated code which is then transmitted to the DPI signs.

3.2.3 Display areas

In a similar way to connection protection, the common use of DPI signs requires a common referencing system. For every jointly used DPI sign (or an associated group of signs) a so-called display area is created with a unique code (DISID (AZBID)).

The DPI sign user system links it's own stops located in the display area with the code of the display area, which allows it to then link the arrivals or departures at a stop with a third-party DPI sign. The DPI sign owner on the other hand links the internal code of its DPI sign with that

of the commonly defined display areas. This completes the referencing chain between the display user and the display owner.

3.3 Visualising the third-party vehicles (VIS)

3.3.1 Tasks and targets

The 'visualisation of third-party vehicles' (VIS) service supports control centre staff in the observation and dispatch of third-party vehicles in the local AVLC. This service offers the ability to connection information about these third-party vehicles in order to be able to display them in the operational representations of the local system. This service provides the following data:

- Geographical position of the vehicle (map representation)
- Position of the vehicle with regard to its pattern (route ladder representation)
- Display in a vehicle list
- Further information on the trip (start and destination stop, service characteristics, etc.)

Schematic ladder representations require an underlying pattern-based model of the trip. This can be achieved by supplying dummy patterns within the local system. Functions for direct dispatch are not made available via this interface. Other channels (radio) must be used for that. In addition, a common pattern supply is required for a route ladder representation. This however is not covered by the interface and must therefore be achieved by other means should such a representation be desired.

3.3.2 Visualisation areas

As the representation of third-party vehicles is generally selective (e.g. specific to route group), the interface offers the possibility of requesting the data accordingly. So-called visualisation areas are created for this purpose. These are abstract terms that define the type and volume of data to be requested.

In general, a visualisation area is usually defined specific to a route. A visualisation area is assigned to one or more routes. With that, any subsequent request prompts an exchange of the data relating to the routes assigned to the visualisation area. As an alternative, a visualisation area can also be defined spatially (within a certain perimeter) or specific to a trip. This however would demand a time-consuming verification process in run time, to assign the vehicles to a specific visualisation area.

Visualisation areas must be agreed between the participating systems bilaterally, in accordance with the operational requirements.

3.4 General message service (GMS)

3.4.1 Tasks and targets

The general message service (GMS) is used by the employees of the participating control systems to exchange operational information. It provides a method of exchanging text messages between the control centres. This service can therefore be seen as a replacement for fax and email communication. The advantages when compared with these other solutions are immediate transmission and simple integration into the control centre software.

Messages can be sent but also revoked (deleted at the recipient server). In addition to the actual content, the messages are also assigned a validity.

Note: The VDV AK handling of incident messages (UmS) includes a recommendation for transmitting incident messages using the SIRI-SX (instead of GMS) service.

3.4.2 Message channels

Within the message service, every control centre message is assigned a so-called message channel. The message channel represents a bilateral arrangement with regard to classifying the content.

This means that the messages can be structured and managed in accordance with the requirements of the control centre staff.

3.4.3 Message formats

Both structured (csv, xml, etc.) and unstructured information can be transmitted via the message service. The xml content model permits free configuration.

There is also a format attribute, which allows the automatic detection of various formats.

4 Architecture

4.1 Communication vs. services

The interface consists of two layers.

1. Communication layer
2. Technical service layer

The communication layer defines a standard procedure for requesting and then exchanging data. This procedure is hereafter referred to as the subscription method. The data consuming system creates so-called subscriptions, which define the type and amount of data to be exchanged. This definition is technically specific and with that already encapsulated in the technical layer. The communication procedure is the same for all services and represents the interface infrastructure (message referencing, error handling, reset behaviour). Reusing it for the various technical services ensures cost-effective implementation and extension of the interface.

The dedicated services are based on the communication layer and address various application areas such as connection protection, DPI etc. These services are not connected, which means that any number of dedicated services can be implemented. This guarantees application-specific implementation.

4.2 Reference data versus process data

The data exchanged via the technical services can be split into two classes:

- Reference data (planning data)
- Process data (actual data)

Every dedicated service (connection protection, DPI, etc.) must always include the process data exchange, as it involves an exchange of real-time information between two control centres. The exchange of reference data is then only necessary if the process data cannot be used on its own and must have previously been compared against reference data in order to calculate an incremental value.

The exchange of reference data generally represents an alternative to the exchange of data at the level of data management. The implementation and use of a reference data service depends on the application case as well as the technical operational requirements. Each individual service can therefore consist of two separate technical services. Within the interface, the process data exchange and reference data exchange are implemented as two separate independent dedicated services.

4.3 Applied protocols

Two protocols are used in the interface:

1. HTTP/1.1 as the transport protocol
2. XML 1.0 for recording the technical data

The XML specification uses XML Schema Version 1.0.

5 'Basic infrastructure' interface description

5.1 Subscription method

5.1.1 Overview

The so-called subscription method defines a common basic communication structure, on which all dedicated services are based. The subscription method consists of a set of request and reply messages, which define an asynchronous communication structure.

The concept follows the client-server model. System A (server) can make data available to another system B (client).

The concept is event-based. The data changes on the basis of an action in the server system (A), which is then communicated to the client system (B) (see

).

The client and server first agree which information is to be exchanged. This is achieved by so-called subscriptions. Subscriptions are defined on the client side. The client sends a subscription request to the server and with that registers interest for specific data (step 1). The data concerned is defined within the actual subscription request. After confirmation from the server, the client can expect a subsequent supply of data.

The server (A) then informs the client (B) about new or modified data by means of a corresponding message (step 2). The client (B) can then retrieve the corresponding data from the server (A) (step 3).

In order to detect a server breakdown, status requests can be periodically sent to the server. With a status reply, the server confirms its functionality (step 4).

Subscriptions have a life span as defined by the client and once expired are automatically deleted by the server. Deletion can also be achieved prior to this by the client (step 5).

The services are managed separately, according to service type. Subscriptions are referenced via so-called SubscriptionID (AboID). A SubscriptionID (AboID) is unique within a service. The client is responsible for assigning the SubscriptionID (AboID).

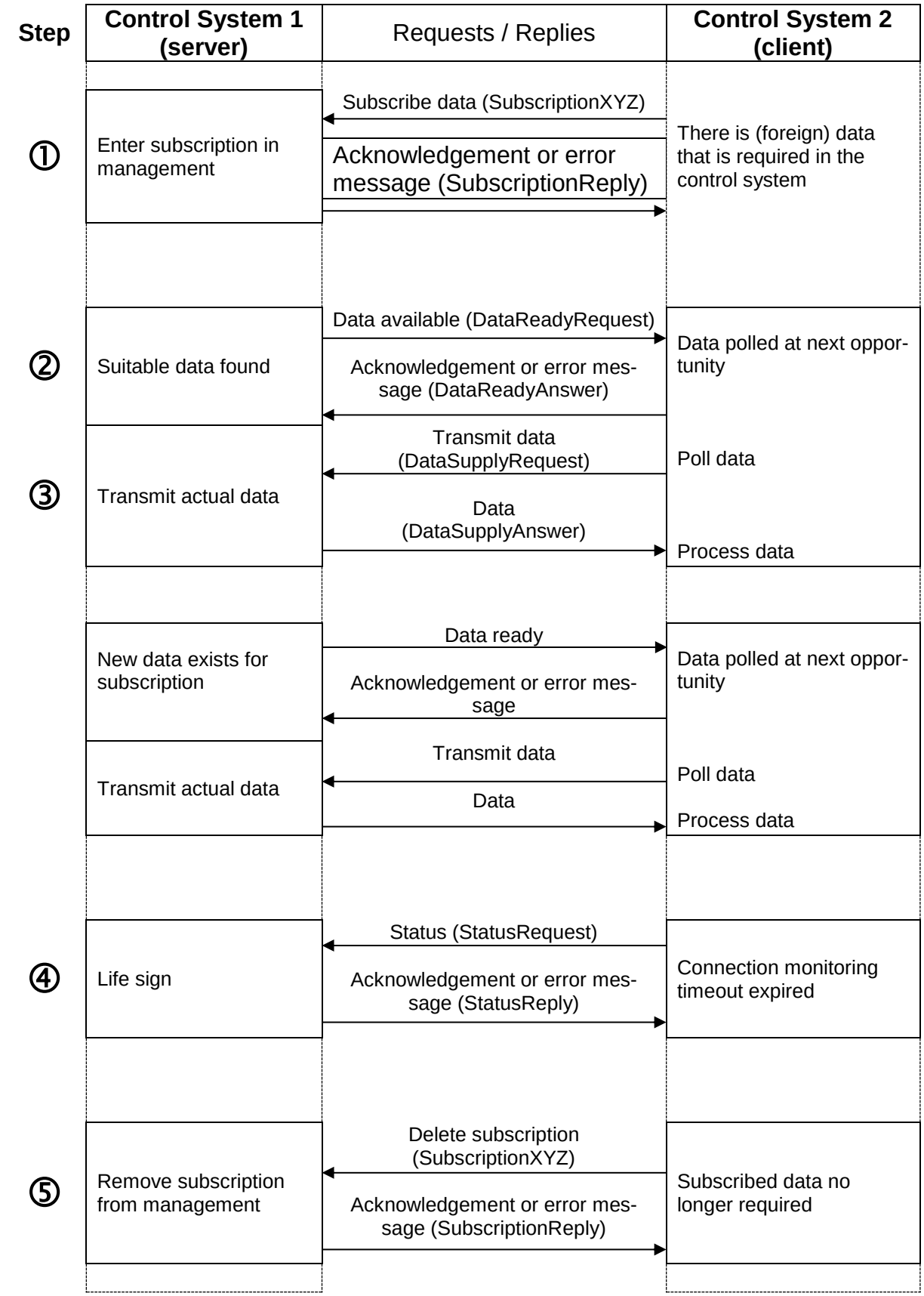

Fig. 5-1: Communication procedure for the subscriptions

5.1.2 Setting up subscriptions

If data is to be polled by an external system, subscriptions must be set up for this purpose. This is achieved by the sending of a so-called SubscriptionRequest (AboAnfrage) by the client.

5.1.2.1 Subscription requests (AboAnfrage)

A SubscriptionRequest (AboAnfrage) defines one or more subscriptions (= sub-elements). Each subscription has an identification that is unique to the service (SubscriptionID (AboID)). This is generated and managed by the client.

Within the subscriptions, service-specific sub-elements define the possible replies. All messages to define a subscription follow the SubscriptionXYZ naming convention, where 'XYZ' represents the respective service code (CPI, DIS, VIS, …) or the corresponding action.

The following subscription requests have been defined:

- Connection protection
  Retrieving feeder data (*CPIRefSubscription (AboASBRef) 6.2.3.2, CPISubscription (AboASB) 6.2.4.2*)
- Passenger information
  Retrieving data for the third-party trips that are approaching a DPI sign. (*DISRefSubscription (AboAZBRef) 6.3.7.2, DISSubscription (AboAZB),* 6.3.8.2)
- Visualisation
  Retrieval of the trip data for the visualisation areas (*VISSubscription (AboVIS)* 6.4.3.2*)*
- Message service
  Retrieval of text messages from the third-party control centre
  (*GMSSubscription (AboAND)* 6.5.4.1)
- Delete
  Deletion of one or all subscriptions (*DeleteSubscription/All 0* (AboLoeschen/Alle)

Definition of the SubscriptionRequest (AboAnfrage):

Sender (Sender):	(attribute) Control centre code of the requesting system.
TimeStamp (Zst):	(attribute) Time stamp of the creation of the request.
XSDVersionID:	(attribute, optional) Version of the interface being used by the client (file name of the XSD file).
CPIRefSubscription (AboASBRef):	(sub-element, alternative, multiple) Retrieves reference data for the connection protection process.
CPISubscription (AboASB):	(sub-element, alternative, multiple) Retrieves process data for the connection protection process.
DISRefSubscription (AboAZBRef):	(sub-element, alternative, multiple) Retrieves reference data for the passenger information service.
DISSubscription (AboAZB):	(sub-element, alternative, multiple) Retrieves process data for the passenger information service.

VISSubscription (AboVIS):	(sub-element, alternative, multiple) Retrieves process data for the visualisation service.
GMSSubscription (AboAND):	(sub-element, alternative, multiple) Retrieves process data for the message service.
DeleteSubscription (AboLoeschen):	(alternative, multiple) Deletes a single subscription.
DeleteSubscrip-tionsAll (Abo-LoeschenAlle):	(alternative) Deletes all subscriptions.

A SubscriptionRequest (AboAnfrage) may only contain the sub-elements of one specific service. If a SubscriptionRequest (AboAnfrage) is created with a SubscriptionID (AboID) and there is already a subscription with this name, the existing subscription is overwritten.

All subscriptions of all services are given a ValidUntilTimeStamp (VerfallZst) by the client when they are created. This defines how long the server must save and manage the subscriptions. The time stamp should be selected so that is past the last potential data registration time point. Subscriptions however should not have a longer validity than is necessary, as each subscription takes up valuable resources. Subscriptions can have validities that stretch beyond the limits of an operational day. With the optional XSDVersionID, the client can specify which interface version it is using. Please refer to 5.1.2.2.

The initialising message after each SubscriptionRequest (AboAnfrage) must generally contain all data relevant to the subscription (or, if using PendingData (WeitereDaten) = true, a portion of the data). Exceptions: DISSubscription (AboAZB) or SISSubscription (AboAUS) with the element 'OnlyUpdate' (NurAktualisierung).

The following example is a simplified representation of the request from AVLC A to set up two subscriptions with IDs 25 and 26 and a validity of one hour.

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T05:00:00">
 <AboXYZ AboID="25" VerfallZst="2001-08-08T06:00:00">
  <Info1> ... </Info1>
  <Info2> ... </Info2>
 </AboXYZ>
 <AboXYZ AboID="26" VerfallZst="2001-08-08T06:00:00">
  <Info1> ... </Info1>
  <Info2> ... </Info2>
 </AboXYZ>
</AboAnfrage>
```

5.1.2.2 Acknowledging subscriptions (SubscriptionReply)

After the data producer (server) has received the request, it confirms with a *SubscriptionReply (AboAntwort)*.

Definition of the SubscriptionReply (AboAntwort):	
XSDVersionID:	(attribute, optional) Version of the interface being used by the server (file name of the XSD file).
Acknowledge (Bestaetigung):	(sub-element, alternative) Contains a global acknowledgement with any error handling (without SubscriptionID (AboID)) for the entire SubscriptionRequest (AboAnfrage)).
AcknowledgeWithSubscriptionID (BestaetigungMitAboID):	(sub-element, alternative, multiple) Contains separate error handling information for each subscription (SubscriptionID (AboID)).

With the optional XSDVersionID element in SubscriptionRequest (AboAnfrage) and SubscriptionReply (AboAntwort), client and server are able to exchange their respective interface codes. This allows compatibility checks on both sides. The current version is specified in the *Version* attribute in the XML structure definition (file name of the XSD file, e.g. "xsd_2017.a"). In this situation it is not essential that both sides are using the same version. In certain application cases, there is sometimes enough retrospective compatibility in order to allow a functioning data exchange in specific contexts. In a similar way, neither partner is obliged to carry out compatibility conversions when different interface versions are in use. The procedure for dealing with different interface versions and possible migration paths must be clarified on a project-by-project basis.

Definition of Acknowledge (Bestaetigung):	
TimeStamp (Zst):	(attribute) Time stamp of the creation of the acknowledgement.
Result (Ergebnis):	(Attribute) 'ok' in the absence of any errors, 'notok' if it was not possible to process the request.
ErrorNumber (Fehlernummer):	(attribute) Specification of a number for a more exact classification of the error (see 6.1.10).
ErrorText (Fehlertext):	(optional) Written description of the error.
DataValidFrom (DatenGueltigAb):	(optional) Start of the validity time period. Must not be greater than the maximum of the StartTime defined in the SubscriptionRequest and the actual time (SISRef service).
DataValidUntil (DatenGueltigBis):	(optional) End of the data horizon of the data producer. Not included if the request lies completely within the data horizon.

ShortestPossibleCycleTime (KuerzMoegliicherZyklus):	(optional) Minimum separation between two updates. Depends on the processing cycle of the AVLC.

If *Acknowledge (Bestaetigung)* has been received by the data consumer (client), it can be certain that the subscription has been set up successfully. Otherwise, the request must be resent. This action overwrites any subscriptions which may have already been set up on the server side.

Definition of *AcknowledgeWithSubscriptionID (BestaetigungMitAboID):*	
SubscriptionID (AboID):	(attribute) References every individual subscription created by the request.
Acknowledge (Bestaetigung):	(sub-element) Delivers information on the subscription of the request assigned to the SubscriptionID (ABOID), indicating if its processing was successful or not.

Since the standard versions VDV453 v2.4 and VDV454 v2.0, a *SubscriptionRequest (Abo-Anfrage)* with several subscriptions can be answered with a *SubscriptionReply (AboAntwort)* with multiple dedicated SubscriptionAcknowledgeWithSubscriptionID (AboBestaetigung-MitAboID) and error messages for every subscription (SubscriptionID (AboID)).

5.1.2.2.1 Procedure for multiple AcknowledgeWithSubscriptionID for a SubscriptionRequest:

A SubscriptionRequest (AboAnfrage) may contain several subscriptions (SubscriptionID (AboID)).

In the SubscriptionReply (AboAntwort), a separate acknowledgement ("AcknowledgeWith-AboID" (BestaetigungMitAboID)) must be sent back to the client for every subscription.

The following approach applies to the processing:

All subscriptions which are successfully set up by the server must be acknowledged with "ok" in the corresponding AcknowledgeWithSubscriptionID (BestaetigungMitAboID).

If at least one subscription of a SubscriptionRequest (AboAnfrage) cannot be set up, a corresponding error message must be returned for the subscription using "AcknowledgeWithSubscriptionID" (BestaetigungMitAboID). The error text shall include a detailed description of the cause of the error.

If no AcknowledgeWithSubscriptionID (BestaetigungMitAboID) is sent for a subscription in the SubscriptionRequest (AboAnfrage), the subscription is classified as **not** acknowledged by the server. The subscriber must then assume that the corresponding subscription could not be set up.

The following example illustrates the use of "AcknowledgeWithSubscriptionID" (BestaetigungMitAboID) when one of the subscriptions could not be set up successfully:

```
<AboAntwort>
      <BestaetigungMitAboID AboID="25">
            <Bestaetigung
                  Zst="2001-08-08T05:00:10"
                  Ergebnis="ok"
                  Fehlernummer="0">
                  <DatenGueltigAb>2001-08-08T05:00:10</DatenGueltigAb>
            </Bestaetigung>
      </BestaetigungMitAboID>
      <BestaetigungMitAboID AboID="26">
            <Bestaetigung
                  Zst="2001-08-08T05:00:10"
                  Ergebnis="notok"
                  Fehlernummer="0">
                  <Fehlertext>… AboXYZ nicht gültig …</Fehlertext>
            </Bestaetigung>
      </BestaetigungMitAboID>
</AboAntwort>
```

Migration scenario:

It is recommended to continue to support the previous procedure with global Acknowledge (Bestaetigung) in the SubscriptionReply (AboAntwort) during the transition time.

The client supports SubscriptionReply (AboAntwort) with SubscriptionAcknowledgeWithSubscriptionID (AboBestaetigungMitAboID)	**The server supports SubscriptionReply (AboAntwort) with SubscriptionAcknowledgeWithSubscriptionID (AboBestaetigungMitAboID)**	**Description**
No	No	See "Procedure if there is only one error message for the entire SubscriptionRequest (AboAnfrage)".
No	Yes	If the client does not support a multiple SubscriptionReply (AboAntwort), it should also not send any SubscriptionRequest (AboAnfrage) with multiple subscriptions (See "Procedure if there is only one error message for the entire SubscriptionRequest (AboAnfrage)").

Yes	No	If the server does not support a multiple SubscriptionReply (AboAntwort), the client must be prepared to receive a simple SubscriptionReply (AboAntwort) with 'notok' and with SubscriptionID (AboID) in response to a SubscriptionRequest (AboAnfrage) with multiple subscriptions (see "Procedure if there is only one error message for the entire SubscriptionRequest (AboAnfrage)").
Yes	Yes	A SubscriptionRequest (AboAnfrage) with multiple subscriptions can be answered as follows: • With Acknowledge (Bestaetigung) with 'ok' if all subscriptions could be set up successfully, • With Acknowledge (Bestaetigung) with 'notok' if no subscription could be set up, • With AcknowledgeWithSubscriptionID (BestaetigungMitAboID) with dedicated 'ok' and 'notok' and error messages for the individual SubscriptionIDs (AboIDs).

5.1.2.2.2 Procedure with a global Acknowledge for the entire SubscriptionRequest:

As several subscriptions can be set up within one SubscriptionRequest (AboAnfrage), but there is only a single error message for the entire process, the procedure in case of error is as follows:

If any of the subscriptions within a SubscriptionRequest (AboAnfrage) cannot be set up, a corresponding error message is returned. The error text should describe in greater detail which subscriptions could not be created.

If an error occurs within a SubscriptionRequest (AboAnfrage), the entire request is invalid and must be rejected. Subscriptions can neither be established nor deleted (see 6.1.10).

The following example illustrates an unsuccessful Acknowledge (Bestaetigung) when all subscriptions were set up successfully:

```
<AboAntwort>
      <Bestaetigung
            Zst="2001-08-08T05:00:10"
            Ergebnis="ok"
            Fehlernummer="0">
            <DatenGueltigAb>2001-08-08T05:00:10</DatenGueltigAb>
      </Bestaetigung>
</AboAntwort>
```

The ***Acknowledge (Bestaetigung)*** element provides information on whether or not it was possible to process the request, any error codes and descriptions, information concerning the maximum possible refresh rate of the data-producing system as well as the available data scope (6.1.1).

Implementation notes:

In order to be able to create a data exchange process during the initial commissioning or when testing the connection of a system coupling, we recommend that each subscription is manifested within a separate *SubscriptionRequest (AboAnfrage)*. This means it is possible to differentiate between possible sources of error, for example due to incompatible meta data.

With a successful SubscriptionRequest (AboAnfrage), all subscriptions must also be established on the sender side.

5.1.3 Provision of data

5.1.3.1 Signalling the availability of data (DataReadyRequest)

If a subscription is already set up and new data is ready, the data consumer is informed of the existence of updated data by means of a DataReadyRequest (DatenBereitAnfrage). This happens with every change to the data that has been assigned to the subscription. The signalling refers to all subscriptions of a service.

Definition of the DataReadyRequest (DatenBereitAnfrage):	
Sender (Sender):	(attribute) Control centre code of the data producing system.
TimeStamp (Zst):	(attribute) Time stamp of the change message.

The following example shows the signalling of new data in an AVLC A service:

```
<DatenBereitAnfrage
      Sender="AVLC A"
      Zst="2001-08-08T08:00:00">
</DatenBereitAnfrage>
```

5.1.3.2 Confirmation of data readiness (DataReadyReply)

The data consumer (client) confirms reception of the signalling by means of a DataReadyReply (DatenBereitAntwort) message. This message contains one Acknowledge (Bestaetigung) element:

Definition of DataReadyReply (DatenBereitAntwort):	
Acknowledge (Bestaetigung):	(sub-element) Contains information on error handling

The following example shows a possible reply to the above request:

```
<DatenBereitAntwort>
      <Bestaetigung
            Zst="2001-08-08T08:00:10"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
</DatenBereitAntwort>
```

The client can now retrieve the data. If the client does not wish to retrieve the data at this time, it can be postponed to a later time. Data polling is independent of the change signalling.

5.1.4 Polling data

Data is polled at the initiative of the data consumer (client). Only the current, actual information is transferred. There is no historical data.

5.1.4.1 Requesting a data connection (DataSupplyRequest)

The request usually occurs after the signalling of updated data (DataReadyRequest (DatenBereitAnfrage)), but can occur at any time after the subscription has been set up. The client sends a DataSupplyRequest (DatenAbrufenAnfrage) message, which prompts the server to deliver the data that has been updated since the last DataSupplyRequest (DatenAbrufenAnfrage):

Definition of the DataSupplyRequest (DatenAbrufenAnfrage):	
Sender (Sender):	(attribute) Control centre code of the requesting system.
TimeStamp (Zst):	(attribute) Time stamp of the request.
AllData (DatensatzAlle):	(optional) true, if all (including any non-updated) records are to be reported, otherwise false; default = 'false'

The following example shows the request for updated data only:

```
<DatenAbrufenAnfrage Sender="CC code AVLC B" Zst="2001-08-08T08:01:00">
      <DatensatzAlle>false</DatensatzAlle>
```

```
</DatenAbrufenAnfrage>
```

If *AllData (DatensatzAlle)* is set to *true* then the system sends all the records of all active subscriptions and not only the data that has been updated since the last request. (For further details, see 0).

Implementation notes:

If a DataSupplyRequest (DatenAbrufenAnfrage) is sent before at least one subscription has been established, the server must report an error. An 'empty' DataSupplyAnswer (DatenAbrufenAntwort) is not permitted.

If a DataSupplyRequest (DatenAbrufenAnfrage) was already sent by the client to the server, a DataSupplyAnswer (DatenAbrufenAntwort) must be received from the client (answer or timeout) before a new DataSupplyRequest (DatenAbrufenAnfrage) is sent. It is therefore recommended to not send any further DataSupplyRequest (DatenAbrufenAnfrage) if there is still an active DataSupplyRequest (DatenAbrufenAnfrage).

5.1.4.2 Transferring data (DataSupplyAnswer)

The server responds with the updated data within a DataSupplyAnswer (DatenAbrufenAntwort) message. The content is service-specific.

The PendingData (WeitereDaten) element indicates whether the content of DataSupplyAnswer (DatenAbrufenAntwort) contains all updated data, or whether for technical reasons the transmission has been broken down into smaller packages. The data consumers can poll this data from the data producers using additional DataSupplyRequests (DatenAbrufenAnfragen). With the last data packet the PendingData (WeitereDaten) element is set to “false”. Contrary to the standard procedure for optional fields, the default value of *PendingData (WeitereDaten)* is "*false*". A missing PendingData (WeitereDaten) element therefore indicates that the complete data transmission is completed with this packet.

The message contained within a data packet must intrinsically represent a valid XML document, which satisfies the XML schema definition.

Definition of DataSupplyAnswer (DatenAbrufenAntwort):

Acknowledge (Bestaetigung):	(sub-element) Contains information on error handling
PendingData (WeitereDaten):	(optional, default "false") “true” if there is additional data to be polled, otherwise “false”.
FeederMessage (Zubringernachricht):	(sub-element, alternative, multiple) Contains feeder messages in the connection protection service.
FetcherMessage (Abbringernachricht):	(sub-element, alternative, multiple) Contains fetcher messages in the connection protection service.
DISMessage (AZBNachricht):	(sub-element, alternative, multiple) Contains messages relating to third-party vehicles serving a particular display area.
VISMessage (VISNachricht):	(sub-element, alternative, multiple) Contains information concerning trips that are to be visualised in a third-party control centre.

GMSMessage (AND-Nachricht):	(sub-element, multiple) Contains information concerning current operational events that are to be conveyed to a dispatcher in a third-party control centre.

With the DataSupplyAnswer (DatenAbrufenAntwort), the server is given authorisation to reset the update flag of the subscription for which data has been sent. A renewed *DataSupplyRequest (DatenAbrufenAnfrage)* therefore no longer elicits the data of the recently requested subscription.

The following example shows a possible server reply to the above request (independent of service):

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T08:01:10"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <ElementXYZ Zst="2001-08-08T08:00:00">
            <SubElement1> ... </SubElement1>
            <SubElement2> ... </SubElement2>
            <SubElement3> ... </SubElement3>
            </ElementXYZ>
      <ElementXYZ Zst="2001-08-08T07:59:00">
            <SubElement1> ... </SubElement1>
            <SubElement2> ... </SubElement2>
            <SubElement3> ... </SubElement3>
            </ElementXYZ>
</DatenAbrufenAntwort>
```

5.1.4.2.1 Using AllData

The following data is sent for DataSupplyRequest (DatenAbrufenAnfrage) with AllData (DatensatzAlle) = 'true':

- With all services, the aim must be for a data consumer to delete all previously received data and to replace it with the newly received data.
- All data that satisfies the conditions of the active subscriptions is transmitted. For the reference services, all data of the time frame confirmed in the SubscriptionReply (AboAntwort) is sent again. If the data provider is no longer able to do this because the system has already deleted trips which start after the confirmed DataValidFrom (DatenGueltigAb) from the memory, an error message can be returned, after which the client must subscribe to the data again.
- For CP, CPIDeviation (ASBFahrplanlagen) (with AtCPIPoint (AufASB) = 'true', if supported) is also sent for all feeders which have already arrived but for which the ValidUntilTimeStamp (VerfallZst) has not yet expired (if the partners have not agreed upon a different period). This can also include arrival messages for feeders which have already left the connection area again, but whose passengers may still be on their way to a fetcher vehicle which must be held back for them to reach it.
- The following TripDelete (FahrtLoeschen) should be sent again (in full if possible so that the data can be represented on any present scheduled data):

- DISTripDelete (AZBFahrtLoeschen) for vehicles which departed early and for which the schedule was not reached,
- TripDelete (FahrtLoeschen) (if possible, with a cause) for cancelled trips within the validity of the active subscription.

5.1.5 Deleting data subscriptions (DeleteSubscription/DeleteSubscriptionsAll)

Subscriptions are automatically deleted by the server after expiration of the validity. If a subscription needs to be deleted by the client earlier than planned, a new *SubscriptionRequest (AboAnfrage)* must be created (5.1.2), which contains the *DeleteSubscription (AboLoeschen)* sub-elements, each of which deletes a subscription.

The following example deletes subscriptions '12' and '5':

```
<AboAnfrage Sender="CC code AVLC B" Zst="2001-08-08T08:10:00">
      <AboLoeschen>12</AboLoeschen>
      <AboLoeschen>5</AboLoeschen>
</AboAnfrage>
```

In order to delete all subscriptions of a service, specify one DeleteSubscriptionsAll (AboLoeschenAlle) element with the value 'true' instead of the separate DeleteSubscription (AboLoeschen) sub-elements:

```
<AboAnfrage Sender="CC code AVLC B" Zst="2001-08-08T08:10:00">
      <AboLoeschenAlle>true</AboLoeschenAlle>
</AboAnfrage>
```

5.1.6 Resume after an interruption

Any interruption to the data connection is detected by time-out messages of the HTTP protocol. The possible error statuses are as follows:

Lost message	by	Status / action
SubscriptionRequest (AboAnfrage)	Server	No response. Client must re-send the request.
SubscriptionReply (AboAntwort)	Client	The subscription has been set up, the client overwrites with a new request.
DataReadyRequest (DatenBereitAnfrage)	Client	The server resends as there is no response
DataReadyReply (DatenBereitAntwort)	Server	Renewed transmission of the request until a response is received from the client.
DataSupplyRequest (DatenAbrufenAnfrage)	Server	Client fails to receive an answer. It must assume that the response has been lost (worst case) and requests all data again (*AllData* (*DatensatzAlle*)), please refer to 0for more details).

Lost message	by	Status / action
DataSupplyAnswer (DatenAbrufenAnt-wort)	Client	Data has been lost, renewed polling is not possible as the subscription update flag has been reset by the server. The client must initiate a DataSupplyRequest (DatenAbru-fenAnfrage).
DeleteSubscription (AboLoeschen)	Server	Client re-sends the message until it receives a response or an error message relating to the unknown SubscriptionID (AboID).
StatusRequest (Sta-tusAnfrage)	Client	If the reply remains missing or the status = "nok", the sender assumes that the connection is interrupted for a longer period of time and repeats the request. All services are no longer available until the client receives a Sta-tusReply (StatusAntwort) = "ok" to a sent StatusRequest (StatusAnfrage). (see 5.1.8.2)
StatusReply (Sta-tusAntwort)	Server	See StatusRequest (StatusAnfrage).

Table 2 - Error statuses and actions after a loss of connection

5.1.7 Resume after a crash

If the client loses its subscription data - possibly after a crash - the subscriptions must be re-created. The first step is to delete all the subscriptions at the server (DeleteSubscriptionsAll (AboLoeschenAlle)). The subscriptions must now be created again.

The client is initially unaware if the server loses its subscription data. DataReadyRequest (DatenBereitAnfragen) is missing but this cannot be distinguished from normal operation, which means that the server crash cannot be detected. In order to recognise this situation, cyclic *StatusRequest (StatusAnfrage)* messages (5.1.8.1) must be sent to the server. The server specifies the time stamp relating to the start of the service within the *StatusReply (Sta-tusAntwort) (*5.1.8.2*)*. If the service started after the subscription was set up, loss of the sub-scription must be assumed. The procedure is now the same as for client loss of data: Delete and re-create all subscriptions.

5.1.8 Alive handling

Status polling is used to establish the availability of services. Two separate information chan-nels are used for this (target URL status.xml, clientstatus.xml), which every service must pro-vide.

5.1.8.1 Requests (StatusRequest)

If the client wishes to establish whether the service is still 'alive', it sends a StatusRequest (StatusAnfrage) to the server and waits for a StatusReply (StatusAntwort).

Definition of the StatusRequest (StatusAnfrage):

Sender (Sender):	(attribute) Control centre code of the requesting system.
TimeStamp (Zst):	(attribute) Time stamp of the creation of the request.

5.1.8.2 Responses (StatusReply, Status)

If the reply arrives and is positive, then the service is available. The StatusReply (StatusAntwort) describes the overall availability of all the information channels of a service. If a channel is faulty, the entire service is no longer available.

Implementation notes:

If a StatusRequest (StatusAnfrage) is not answered or is answered with Status = "nok" by the server, the client must assume that the entire service is unavailable. In this case, the client continues to send StatusRequest (StatusAnfrage) to the server to check when the service is available again.

As long as the client does not receive a StatusReply (StatusAntwort) = "ok", it should sent no further requests (e.g. SubscriptionRequest (AboAnfrage), DataReadyRequest (DatenBereitAnfrage), DataSupplyRequest (DatenAbrufenAnfrage)) to the server to avoid applying more stress to the server and flooding it with requests should it be experiencing a system problem

Definition of the StatusReply (StatusAntwort):

Status (Status):	(sub-element) Indicates whether the service is available.
DataAvailable (DatenBereit):	If true, there is data ready to be retrieved
StartServiceTimeStamp (StartDienstZst):	Specifies the time of the start of the service. If the service is not available any value can be provided here.
DataVersionID (DatenVersionID):	(optional) Defines the active data version.

The Status (Status) element contains only one Result (Ergebnis) element in addition to the creation time stamp, which delivers true if the service is available.

Definition of Status (Status):

TimeStamp (Zst):	(attribute) Time stamp of the creation of the status information.
Result (Ergebnis):	(attribute) 'ok' if the service is available, otherwise 'notok'

In addition, the *StatusRequest (StatusAnfrage)* allows the client to recognise whether a service has been restarted in the meantime and whether the data version has changed. Within the StatusRequest (StatusAnfrage) the server specifies the last start of the service. If the start occurred after the set-up of a subscription, the service was restarted in the meantime (5.1.7).

As soon as the server shares an updated *StartServiceTimeStamp (StartDienstZst)* value in a *StatusReply (StatusAntwort)*, and the *DataVersionID (DatenVersionID)* has changed or is missing (e.g. because it is not yet supported by the system), the client must assume that the server service was rebooted and the data supply, incl. the subscription, has been lost.

If the server wishes to signal a new data version, it must inform the client by means of a simultaneous updating of *StartServiceTimeStamp (StartDienstZst)* and *DataVersionID (DatenVersionID)*.

As soon as the server shares an updated *StartServiceTimeStamp (StartDienstZst)* in a *StatusReply (StatusAntwort)*, but the *DataVersionID (DatenVersionID)* remains unchanged, the client can assume that the server service was rebooted and that the existing data supply is still available. Therefore, the client need **not** delete the data and subscriptions related to that service. In this case, the subscriptions do **not** need to be updates and the data does **not** need to be requested again.

The following example shows a StatusRequest (StatusAnfrage) with a corresponding (successful) StatusReply (StatusAntwort):

Client request (AVLC A):

```
<StatusAnfrage Sender="AVLC A" Zst="2002-02-14T14:03:49"/>
```

Server reply: Service available, there is no data to be retrieved:

```
<StatusAntwort>
      <Status Zst="2002-04-02T14:00:00" Ergebnis="ok"/>
      <DatenBereit>true</DatenBereit>
      <StartDienstZst>2002-04-02T06:00:00</StartDienstZst>
</StatusAntwort>
```

5.1.8.3 ClientStatusRequest

If the server wishes to check the client status, it sends a ClientStatusRequest (ClientStatusAnfrage) to the client and waits for a response (ClientStatusReply (ClientStatusAntwort))

Definition of the ClientStatusRequest (ClientStatusAnfrage):

Sender (Sender):	(attribute) Control centre code of the requesting system.
TimeStamp (Zst):	(attribute) Time stamp of the creation of the request.
WithSubscriptions (MitAbos):	(attribute, optional) States whether the reply should also contain the list of currently active subscriptions. *default="false"*
StartServiceTimeStamp (StartDienstZst):	Specifies the time of the start of the service. If the service is not available any value can be provided here.
DataVersionID (DatenVersionID):	(optional) Defines the active data version in the server.

Within the request, the server can specify whether it wishes to receive a list of the subscriptions that are currently active for the service. This enables the server to initiate data matching between client and server.

At the same time, the server sends its StartServiceTimeStamp (StartDienstZeit) and its data version in the request, which gives the client the opportunity to react to any changes:

As soon as the server shares an updated *StartServiceTimeStamp (StartDienstZeit)* in a *ClientStatusRequest (ClientStatusAnfrage)*, and the *DataVersionID (DatenVersionID)* has changed or is missing (e.g. because it is not supported by the system), the client must assume that the server service was rebooted and the data supply, incl. the subscription, was lost.

If the server wishes to signal a new data version, it must inform the client by means of a simultaneous updating of *StartServiceTimeStamp (StartDienstZst)* and *DataVersionID (DatenVersionID)*.

As soon as the client shares an updated *StartServiceTimeStamp (StartDienstZst)* in a *ClientStatusRequest (ClientStatusAnfrage)*, but the *DataVersionID (DatenVersionID)* remains unchanged, the client can assume that the server service was rebooted but that the data supply is still available. The client does **not** need to delete the data and subscriptions related to this service. In this case, the subscriptions do **not** need to be updates and the data does **not** need to be requested again.

Definition of ClientStatusReply (ClientStatusAntwort):

Status:	(sub-element) Indicates whether the service is available.
StartServiceTimeStamp (StartDienstZst):	Specifies the time of the start of the service. If the service is not available any value can be provided here.
ActiveSubscriptions (AktiveAbos):	(sub-element, optional) The list of subscriptions that are active from the client's point of view.

The Status element contains only one Result element in addition to the creation time stamp, which delivers true if the service is available.

Furthermore, if the ClientStatusRequest (ClientStatusAnfrage) is sent WithSubscriptions (MitAbos), the ActiveSubscriptions (AktiveAbos) element contains the list of subscriptions that are valid for the respective service.

If the server detects a difference between its subscription list and the list from the client, the server can either implicitly eradicate the difference, by not deleting the active subscriptions from the client's point of view and registering the extra subscriptions. It can then start to prepare data for these subscriptions or set the StartServiceTimeStamp (StartDienstZst) in it's StatusReply (StatusAntwort) to the current time, which forces a reinitialisation of the client. The second method is recommended.

If the structure of ActiveSubscriptions (AktiveAbos) is empty, the client has no active subscriptions. If the server detects any, they should be silently removed.

If the client is in an initialisation phase, i.e. it is deleting or setting up subscriptions, it should completely leave out the ActiveSubscriptions (AktiveAbos) element. In this situation the server does not initiate a matching process.

Definition of ActiveSubscriptions (AktiveAbos):

CPIRefSubscription (AboASBRef):	(sub-element, alternative, multiple) Subscriptions for reference data for the connection protection process.
CPISubscription (AboASB):	(sub-element, alternative, multiple) Subscriptions for process data for the connection protection process.
DISRefSubscription (AboAZBRef):	(sub-element, alternative, multiple) Subscriptions for reference data for the passenger information process.
DISSubscription (AboAZB):	(sub-element, alternative, multiple) Subscriptions for process data for the passenger information process.
VISSubscription (AboVIS):	(sub-element, alternative, multiple) Subscriptions for process data for the visualisation process.
GMSSubscription (AboAND):	(sub-element, alternative, multiple) Subscriptions for process data for the message service.

The following examples show a StatusRequest (StatusAnfrage) with a corresponding (successful) ClientStatusReply (ClientStatusAntwort):

Request from the server (AVLC A):

```
<ClientStatusAnfrage Sender="AVLC A" Zst="2013-02-14T14:03:49" Mit-
Abos="true"/>
```

Example 1: Client reply: The service is available, the list of active subscriptions is included:

```
<ClientStatusAntwort>
      <Status Zst="2013-02-14T14:03:50" Ergebnis="ok"/>
      <StartDienstZst>2013-02-14T10:00:00</StartDienstZst>
      <AktiveAbos>
            <AboXYZ AboID="26" VerfallZst="2013-02-15T06:00:00">
                  ...
                  ...
            </AboXYZ>
            <AboXYZ AboID="30" VerfallZst="2013-02-15T17:00:00">
                  ...
                  ...
            </AboXYZ>
      </AktiveAbos>
</ClientStatusAntwort>
```

Example 2: Client reply: The service is available, there are no active subscriptions and neither should any be active:

```
<ClientStatusAntwort>
      <Status Zst="2013-02-14T14:03:50" Ergebnis="ok"/>
      <StartDienstZst>2013-02-14T10:00:00</StartDienstZst>
      <AktiveAbos/>
</ClientStatusAntwort>
```

Example 3: Client reply: Service available, client is initialising and does not want to give out any information concerning active subscriptions:

```
<ClientStatusAntwort>
      <Status Zst="2013-02-14T14:03:50" Ergebnis="ok"/>
      <StartDienstZst>2013-02-14T10:00:00</StartDienstZst>
</ClientStatusAntwort>
```

5.2 HTTP connection

5.2.1 Procedure

Message exchange via HTTP is achieved with the POST method. Data in the form of requests can be transferred from a client to a server. The data prepared by the server is transferred in the response to the request.

HTTP-POST example:

```
POST /leitsystem1/ans/status.xml HTTP/1.1
Host: test1:1111
Content-Type: text/xml
Charset="iso-8859-1"
Content-Length: 64

<StatusAnfrage Sender="AVLC A" Zst="2002-02-14T14:03:49"/>
```

Example of an HTTP reply:

```
HTTP/1.1 200 OK
Content-Type: text/xml Charset="iso-8859-1"
Content-Length: 87

<StatusAntwort>
      <Status Zst="2002-04-02T14:00:00" Ergebnis="ok"/>
      <DatenBereit>true</DatenBereit>
      <StartDienstZst>2002-04-02T06:00:00</StartDienstZst>
</StatusAntwort>
```

5.2.2 Character set

The ISO-8859-1 character set is used exclusively.

5.2.3 Service codes

The provision of specific data and its system-wide processing is referred to as a 'service'.

At the moment, the on-line interface supports the following services:

Service	Code	Description
Connection protection reference data service	cpiref	On the server side it provides the planning data for the feeders. On the client side this data is processed in the connection protection.
Connection protection process data service	CP	On the server side, this provides the current real-time data for the feeders. On the client side this data is processed in the connection protection.
Passenger information reference data service	Dfiref	On the server side provides departure boards for the DPI signs provided with reference data.
Passenger information process data service	Dpi	On the server side provides the passenger information data. This is shown on the corresponding DPI signs on the client side.
Visualisation of trips	Vis	Provides the trip data server side, which is then represented in the control centre on the client side.
Message service	Gms	Provides text messages on the server side.

Table 3 - HTTP codes

5.2.4 Request URL

All requests must be directed at specific URLs. The request URL depends on the service and on the type of request.

Name	Request code	Description
Poll status	status.xml	This request is used to test whether a service on the polled server is responding. The control system code and service code are returned as a response. This request is used to cyclically check the connection status.
Client status requests	clientstatus.xml	If the server wishes to check the status of the client, it sends a ClientStatusRequest (ClientStatusAnfrage) to the client and waits for an answer, ClientStatusReply (ClientStatusAntwort).
Managing data subscriptions	subscription.xml (aboverwalten.xml)	This request is used to poll on-line data from the respective control system or to delete existing subscriptions. Acceptance of the request is confirmed in return or in the case of error a corresponding error message is returned.

Name	Request code	Description
Report new data	dataready.xml (datenbereit.xml)	This request can be used to tell a partner system that there is data ready to be retrieved. The partner system then initiates the data connection with a 'connection data' request. Acceptance of the request is confirmed as a response, or a corresponding error message is sent.
Polling data	polldata.xml (datenabrufen.xml)	This request is used to poll on-line data. The available data is returned or a corresponding error message.

Table 4 - HTTP request URL

The URL to which the above requests are directed is defined as follows:

```
HTTP_URL = "http:" "//" host [ ":" port ] abs_path
```

`"http:"` "//" Name of the protocol used

host Denotes the HTTP server to which the request is sent.

':'port Specifies the port via which the TCP/IP connection is made. This specification is not required if the default port (80) is used.

abs_path Specifies the request path.

Definition of abs_path for the on-line interface:

```
abs_path = "/" leitstellenkennung "/" dienstkennung "/" anfragekennung
```

The control centre code must be specified by the interface partner. Specification of the third-party control centre code within the full path facilitates specific implementations of the interface depending on the partner. The service code is specified when a service is created and the request code can be taken from the above table.

The requests for the various services differ in terms of the URL path. The requests are always interpreted by the service to which they are directed. Cross-service requests are not possible.

Example of a status request (port 8080) of system A to the connection protection service of system B:

http://serverhostsystemb:8080/leitstellenkennungsystema/ans/status.xml

5.2.5 Error handling

Protocol-specific status codes are supported at the level of the HTTP protocol. The most important codes are listed in the following tables:

Code	Short description	Description
200	OK	Request successful
400	Bad Request	The server did not understand the request. The request should not be repeated by the client.
401	Unauthorized	User name and password are required for the request.
403	Forbidden	The server 'understands' the request but cannot fulfil it.
404	Not Found	The requested URL was not found.
408	Request Timeout	Server is not responding.

Table 5 - HTTP error messages

For the on-line interface it is sufficient if status code 200 (OK) differs from the other codes on the client side. This indicates the successful reception of the request. All other codes supplied by the HTTP server must be interpreted as a rejection of the request. The supplied status code provides additional information on the cause of the rejection.

If the server fails to respond within the pre-specified time-out, this must also be interpreted as a rejection of the request.

Errors on a technical level are reported within the scope of the message acknowledgement using defined fields (ErrorNumber (Fehlernummer), ErrorText (Fehlertext)).

5.3 Security

AVLC systems are generally incorporated into a company network. As the connection of internal systems via the internet poses a security risk, we recommend using a VPN (virtual private network) for managing the data traffic. This connects the computers or partial networks of different company networks, resulting in a virtual secure network, to which only the authorised computers have access. The data within the virtual network is encoded before transmission, which means it can be carried via unprotected connections (i.e. the internet).

6 “Technical services" interface description

The technical services are based on the generic subscription concept. They complement the requests and responses with the technical data that is necessary to carry out the services. With regard to the subscriptions, message processing is standard.

The following chapters describe the various dedicated services - connection protection (CP), dynamic passenger information (DPI), visualisation of third-party vehicles (VIS) and the general message service (GMS).

6.1 General specifications

There is a basic problem when exchanging data between different operations: How do we generate a common understanding of a particular object (e.g. a stop)? How do we create a common reference basis? What is the 'life-span' of these objects, or how must the data be managed in terms of time? The following chapter explains the fundamental considerations on which the modelling is based. These considerations are reflected as data definitions in the data management systems of the AVLC systems. This data is subsequently referred to as meta data.

6.1.1 Operational days

The entire data exchange is based on the operational data that is currently stored in the AVLC system. As the validity time for planning data can vary from operation to operation, this specification is restricted to the exchange of daily data.

The problem here is that AVLC operating times generally differ from agency to agency. Some systems are shut down periodically, after which they are re-initialised and data is copied from the journey planning. Other agencies work round the clock without a break. It should be noted however that it is not always necessary to have an accurately defined switch-over point. Several timetable versions can co-exist within a defined passenger connection time. From this we can deduce that there is no clearly defined time frame in which it is possible to guarantee that both participating systems have full data availability. The reality is partial availability in which a system is unable to provide all requested data. This affects the exchange of reference data in particular. However, availability cannot be guaranteed even for last minute requests for process data.

For this reason, a request time frame is specified and a response, which allows the end of the data horizon of the data producer to be detected. This means the consumer knows that the request cannot be answered (completely), or that it must be repeated at a later time.

6.1.2 Date and time formats

Each time entry relates to the so-called UTC (Coordinated Universal Time). Deviations from this time zone are coded in accordance with ISO 8601.

Example: 2000-04-07T18:39:00+01:00.

Without a specification of time difference, the time is already in UTC. In this case there may be a Z at the end.

Example: 2002-04-30T12:00:00 corresponds to 2002-04-30T12:00:00Z.

In other words, the first 19 characters are obligatory and correspond to local time or UTC.

No further time units beyond the second are used, e.g. 1/10th, 1/100th second etc. If so, they are ignored at import.

This method of time representation avoids any problems with summer/winter changeover.

6.1.3 Control centre code

In order to be able to differentiate between messages from different communication partners within a service, each message contains a unique control centre code ('Sender' attribute) relating to the requesting system. All additional bilaterally agreed meta data are based on the control centre code.

6.1.4 Local references

In addition to the daily planning data, there must be extra planning data between two systems using inter-operational connection protection, which allows a common understanding to be created via the respective local references. The internal local references (stop references, DPI sign references, etc.) can then be managed on both sides.

There is however a major disadvantage with this approach: The direct linking of the local location reference with that of the third-party operation links the databases. Changes in the underlying database can quickly lead to inconsistencies.

For this reason, an independent location code has been introduced, which effectively decouples the data management systems. The basic idea is as follows: Any direct linking is rendered obsolete by the agreement of 'common' names and codes. There is only a partial link between the local data and the 'meta data'. The reference can be made for location points (e.g. stops, masts etc.) or location areas (stopping zones). The suitability of the reference model depends on the data model of the AVLC as well as the local conditions (interchange times).

In the case of the message service there are no local references, but the exchange of data is still based on bilaterally agreed IDs, which effectively unlink the systems. These are termed channels and split the volume of possible messages into classes. The channel therefore represents the equivalent of a local reference in the message service.

The following references must be defined in the respective services:

Service	Location code	Code name
Connection protection	Connection area	CPIID (ASBID)
Dynamic passenger information	Display area	DISID (AZBID)
Visualisation	Visualisation area	VISID (VISID)
Message service	Channel	ChannelID (KanalID)

Table 6 - Local references in the dedicated services

The agreement of common names can be discussed in planning conferences or similar committees. The agreement should have mid to long term validity, in order to minimise the need for data modifications.

Agreements can be exclusively bilateral. Within a data exchange process, the reference to the communication partner is always recognisable. For this reason, every label need only be related to one specific communication partner. A regional agreement of labels between several agencies may be useful but is not necessary.

All reference data within this specification must be linked using this concept with the internal agency data. There is no need for any common data supply beyond the meta data.

Note: In order to be able to manage any changes in advance, the data management requires a version structure for the meta data suitable for every communication partner.

6.1.5 Trip reference (*TripID*)

The referencing of third-party trips presents similar challenges to that of the local references. Operation-specific terms such as route / block / run do not have common semantics. Again, the required uniqueness is not guaranteed.

Within an AVLC there is therefore an internal label that references the trip uniquely (within the local data horizon). As the data horizon of the third-party operation is unknown, it is not possible to exclude the situation in which two trips of a third-party operation have the same internal trip label within the local operational day (there are different operational days in the third-party AVLC).

In order to be able to use the internal AVLC trip label as a universal meta datum, it must be extended with the operational day code in the interface:

Definition of TripID (FahrtID):

TripName (FahrtBezeichner):	Unique label for a trip within the operational day.
OperationalDay (Betriebstag):	Date of the operational day on which the trip is travelling.

It is irrelevant how the operational day is described internally. Of significance is the uniqueness of the combined TripID (FahrtID) element.

6.1.6 Route and direction references

In order to be able to separate the data management with regard to route and direction references between different agencies, non-specific route (LineID) and DirectionIDs (RichtungsID) are used in the interface. These are agreed bilaterally.

It is necessary to create a reference between a route or direction ID and a local route or direction within the data management. Labels of third-party companies are used in the connection definition.

6.1.7 Product types

Product types are used within the interface to classify the quality features of trips. Possible products are 'Intercity', 'regional train', 'express bus' etc. There are currently no common specifications regarding product classification.

In order to be able to present common product labels (text and icons etc.) to the passengers, in the case of cross-operation connection protection and dynamic passenger information, meta data management is necessary here too. It is necessary to model the inter-operational ProductID (ProduktID) and local quality classes. In the same way as all other meta data, the agreement is bilateral.

As the use of ProductID (ProduktID) is optional, it is also possible to avoid the agreement of inter-operational product types.

6.1.8 Branch trips

The announcement of planning and process data requires the unique referencing of the approach of a feeder to a stop. However, the combination of TripID (FahrtID) and CPIID (ASBID) is not always unique. If a trip serves a particular stop more than once, it is not possible to differentiate between the first and second approach. The so-called stop sequence counter (StopSeqCount (HstSeqZaehler)) is used to solve this problem.

The stop sequence counter is a positive integer, which increases with each approach of a trip to a connection area. The StopSeqCount (HstSeqZaehler) need not be sequential but must increase in a strictly monotone way.

Data producing systems that use other methods to count multiple approaches (planned arrival times, pattern point counter) can use this data directly, coded as an integer, as long as the monotony is observed.

Branch trips of the fetcher vehicles are not taken into consideration.

6.1.9 Service characteristics

In order to be able to reproduce service characteristics such as school trips, wheelchair access trips etc. there must be references in both systems to common cross-operational meta data. This is achieved within the scope of operational data management:

Local service characteristic	Inter-operational service characteristic	Third-party agency code
School trip	SCHOOL	AVLC B
Suitability for disabled passengers	WHE	AVLC B
Intermitting trip	INTERMIT	AVLC B
Museum trip	MUS	AVLC B

Table 7 - Definition of inter-operational service attributes

6.1.10 Errors at the technical level

In addition to the general errors that arise as a result of missing availability of the client, server or network and with that occur at the HTTP level, there are other errors that arise within the service information. If, for example, a meta data reference is unavailable (e.g. the CPIID (AS-BID)), the required service cannot be carried out and an error message must be returned to the client. This is achieved with the Acknowledge element, which is a sub-element of every response message. This contains the optional ErrorNumber (Fehlernummer) element, which specifies an error category:

ErrorNumber (Fehlernummer)	Meaning / cause
0	OK: (no error)
100-199	XML error Examples: XML document format incorrect, error checking against schema
200-299	Reference data violation Example: Invalid 'Sender' attribute in a request
300-399	All other errors All other errors that can be traced back to faulty requests are reported in this category. The request should not be repeated as it was.
400-499	Other replies Error messages, which result from reasons other than a faulty request, e.g. the requested data is currently being processed or is temporarily unavailable. Subsequent repetition of the request can prove successful.

Table 8 - Technical error types

If communication is transmitted via a DDS, when it comes to trouble shooting it is important to know whether the error was triggered by a peripheral system or by the DDS itself. This

case differentiation helps the operator of the requesting system to decide whether to contact the operator of the DDS or the operator of the requested system for an analysis of the malfunctioning.

Errors that are triggered by the peripheral systems (of the requested system) should, as before, be passed on by the DDS transparently with an error code in the range of 100-499.

The subsequent number range of 500-599 should only be used in cases where the error has been triggered by the DDS. Possible error statuses are:

500-529	Errors in the request, i.e. as 100-499 but this time 'detected' by the DDS. • XML errors • Reference data violation • Other errors in the request (The statuses only occur if the DDS validates the messages before they are passed on).
530-559	Errors in connection with connection to a peripheral system. • Peripheral system is unknown / cannot be established • Peripheral system unavailable / not responding • XML error in the peripheral system reply
560-599	Other errors in the data fetcher, • e.g. the database is currently unavailable etc.

The ErrorText (Fehlertext) element is available for a more exact specification of the error, permitting a written explanation. The written explanation must contain the faulty element (tag) including its value. In general, the message should be as detailed as possible, as it represents the only possibility for troubleshooting.

If an error occurs within a SubscriptionRequest (AboAnfrage), the entire request is invalid and must be rejected. Subscriptions can neither be created nor deleted if either the syntax or semantics check fails. Checking the meta data should therefore be carried out in advance or via an intermediate system.

6.1.11 Optional fields

Much of the information within the messages is optional. This means that the recipient cannot bank on the availability of this data, if the data producing system cannot prepare the data for technical or operational reasons. This means it is not possible to use default settings for optional fields.

Exceptions to this are mentioned explicitly (see Trainset (Traktion), PendingData (Weitere-Daten)).

6.1.12 Texts for publication

Within the messages, specific information texts are intended explicitly for publication for passengers. The affected elements are labelled with the suffix “**Text**” in the element name (e.g. “Line**Text**” (Linien**Text**)).
The information contained in these elements was created explicitly as passenger information by the data producers and must be used unaltered in cases of communication.

6.2 Connection protection (REF-CP, CP)

6.2.1 Introduction

Inter-operational connection protection demands that both operations have an AVLC system with internal connection functionalities. The data exchange described in this chapter enables the systems to receive all necessary data concerning the feeder vehicles to allow connection monitoring and dispatch to take place. The operational dispatch methods remain untouched.

Two services are made available via the interfaces. The connection protection reference data service (REF-CP) provides functions for transmitting planned arrival times at connection areas.

The connection protection process data service (CP) offers functions for exchanging actual data (timetable statuses, effects of manual interactions).

The process data service can be used in one of two versions:

1. Trip-specific exchange of process data

   Corresponds to the procedures in VDV453 Version 1.0. This involves the exchange of process data relating to previously known trips. Notification occurs via common journey planning or use of the connection protection reference data service (REF-CP). With that, the connection agreement can be started directly in connection with the connection of the daily timetables into the AVLC system. The amount of connections is therefore known in advance.

2. Time-based connection protection

   This procedure has been introduced into version 2.0 of VDV453. It represents connection protection without any prior exchange of reference data. Connections are agreed on a 'last-minute' basis.

Within the connection protection service, messages are transmitted from the feeder to the fetcher (FeederMessage) as well as - for the process data service - from the fetcher to the feeder (FetcherMessage).

1. Feeder messages

   All messages that are transferred from the feeder to the fetcher are encapsulated within a *FeederMessage (Zubringernachricht*) element (see 6.2.4.3). The feeder message is used in both the reference and process data services.

   Within the context of the connection protection reference data service, the feeder message consists exclusively of timetable information (*CPISchedule* (*ASBFahrplan)* 6.2.3.3.).

In the process data service, the feeder message can consist of any combination of time-table status information (*CPIDeviation (ASBFahrplanlage*), *6.2.4.3.1*) and trip cancellation messages (*CPITripDelete (ASBFahrtLoeschen*), *6.2.4.3.2*).

2. FetcherMessages

   In addition to the information that is conveyed from the feeder to the fetcher, there are further messages within the process data service (*FetcherMessage (Abbringer-Nachricht)*, 6.2.4.4) that are transmitted in the opposite direction, from the fetcher to the feeder. This data provides passenger information in the feeder trips. All reverse channel messages are optional. Whether or not a system sends these messages is subject to an inter-agency agreement. Differences between the various parties are managed using meta data.

6.2.2 Operational data supply and management

All that is required for the simplest case of transferring the connection protection reference data is the pre-specified supply with common connection area codes (CPIID (ASBIDs)). A table is created which maps the abstract location codes (connection areas) onto the internal location codes (stops or stopping areas), see 6.1.4. On the fetcher system side the interchange time must also be managed.

A definition of this type may look as follows in the two systems:

CPIID	Third-party agency code	Internal stop code
12345	AVLC B	3642
12346	AVLC B	4564
35678	AVLC B	7765

Table 9 - Representation of the connection locality labels for the feeder (AVLC A)

CPIID	Third-party agency code	Internal stop code	Interchange time
12345	AVLC A	2345	0
12346	AVLC A	3687	0
35678	AVLC A	7566	1

Table 10 - Representation of the connection locality labels for the fetcher vehicle (AVLC B)

This yields the possibility of creating requests for feeder data. It is not possible, however, to make selections based on specific connection relationships between routes, directions or time-

of-day periods. This demands an extended data management, which additionally defines the connection relationships. Data management is carried out from the point of view of the fetcher.

Third-party agency code	CPIID	LineID (F)	DirectionID (F)	LineID (R)	DirectionID (R)	Time
AVLC A	12345	10	Zoo	12	CSt.	10:00-12:00
AVLC A	12345	10	Station	12	CSt.	10:00-14:00

Table 11 – Definition of interchange relationships for the fetcher

This definition states that the trips of two directions (Zoo, Central Station) of a route (10) of third-party agency AVLC A could be potential feeders, but only between 10:00am and 12:00pm or between 10:00am and 2:00pm and only if the fetcher vehicle of the local agency is travelling on route 12 in the direction of Central Station.

It is clear that in addition to the CPIID (ASBID), the third-party route / direction must also be specified as a meta datum. The local route and direction information must also be managed as meta data, because fetcher information must be made available for the reverse channel messages or the subscriptions.

On the feeder side, there is no need for meta data relating to the route or direction of the fetcher.

If any additional data is to be implemented as criteria (service attribute, product type) for finding connections, it must be defined within the context of the connection relationship definitions. It should be noted however that this information is optional.

The product types (ProductID (ProduktID)) must also be agreed by all participating agencies.

6.2.3 Reference data service (REF-CP)

The reference data service is used to exchange planning data for potential feeder trips. The service is localised, i.e. all requests and replies relate to specific connection areas.

The exchange of reference data is only useful in connection with the trip-specific connection protection. The trip-specific connection protection requires a knowledge of the feeder trips. This can be achieved by means of a common timetable supply or by an exchange of data using the reference data service.

The fetcher initiates the reference data service. It prompts the feeder for trips in a specific connection area. Assuming operational meta data management is in place, the request can be restricted to third-party routes and directions.

The feeder fulfils this request by returning a departure board for the desired connection areas. The TripID (FahrtID) supplied with every trip is used for subsequent referencing in the subscriptions of the connection protection process data service.

With that, the reference data service fulfils two objectives:

- Preparing the retrieval of process data (trip-specific)
- Advance connection determination

Changes in the timetable brought about by dispatch actions are also communicated within the framework of the subscription process (6.2.3.1.2). This function is optional on the server side. Clients must however always be able to formally accept and acknowledge any updates.

6.2.3.1 Data exchange

The fetcher is responsible for initiating the reference data service. For every defined interchange relationship (Table 11), the fetcher system generates a *SubscriptionRequest (Abo-Anfrage)* (and with that a *SubscriptionID (AboID)* that is unique within the service). The request comprises the CPIID (ASBID) of the connection location and the additional (optional) filters of Route (Linie) and Direction (Richtung). It is possible to define several requests within one SubscriptionRequest (AboAnfrage).

The subject data of the request is encapsulated in a CPIRefSubscription (AboASBRef) element, which represents the service-specific framework. The request time frame generally extends along the entire availability window of the actual system (to the end of the operating day or the end of the operating day on the fetcher route).

After receiving the message and checking the validity of the meta data, the feeder system responds with an Acknowledge (Bestaetigung).

Afterwards the feeder system searches its own planning database (including all current dispatch actions) and compiles the departure boards. The readiness of this data is made known to the fetcher by means of a DataAvailable (DatenBereit) message.

The fetcher can now request all modified data (with a DataSupplyRequest (DatenAbrufenAnfrage) message). The feeder system responds with a DataSupplyAnswer (DatenAbrufenAntwort) message, which contains the relevant information (departure boards).

6.2.3.1.1 Availability horizon

As the end of an OperationalDay (Betriebstag) of the local AVLC does not always coincide with the end of the OperationalDay (Betriebstag) of a third-party AVLC, it must be presumed that any provision of data cannot be valid for the entire request period. To ensure that the data consumer (fetcher) is aware of this, the optional DataValidUntil (DatenGueltigBis) element is defined in the Acknowledge (Bestaetigung) message of the CPIRefSubscription (AboASBRef) request. If the feeder receives a request, which goes beyond its data availability horizon, it can enter the end of this horizon in this optional element. If the DataValidUntil (DatenGueltigBis) element is missing, the fetcher can assume that its request can be fulfilled in its entirety.

If the request time frame lies completely outside the data horizon, this is signalled by a value in the DataValidUntil (DatenGueltigBis) element that is before the time point of the request.

6.2.3.1.2 Updates

After sending a request, the fetcher promptly receives a response in the form of departure boards. If these change, further DataReadyRequest (DatenBereitAnfragen) are sent to the fetchers in order to indicate that the data has changed.

The only change to be reported is the addition of extra trips. The dispatch reason behind the change is of no consequence (reinforcement trips, diversions of other routes via the connection area).

Changes to the trip itself (start stop, departure time, means of transport) remain hidden or are communicated within the context of the process data exchange. Trips that will no longer arrive at the connection area due to a manual interaction are removed from the connection protection with a specific *CPITripDelete (ASBFahrtLoeschen)* message.

Replacement trips for any cancellations should be exchanged with transparency (recommended implementation). This represents the same trip with a different vehicle.

Updates on the server side are optional. Clients must always be able to accept any updates.

6.2.3.2 Requesting timetables (CPIRefSubscription)

The polling of timetable data involves a SubscriptionRequest (AboAnfrage) with one or more embedded CPIRefSubscription (AboASBRef) elements.

The CPIRefSubscription (AboASBRef) element specifies the connection area, optional filter criteria as well as the request time frame in which data is to be supplied.

Definition of CPIRefSubscription (AboASBRef):

SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription of visualisation data. The SubscriptionID (AboID) is given by the fetcher system.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the subscription is valid.
CPIID (ASBID):	Reference to the connection area.
LineID (LinienID):	(optional) Filter for the feeder route whose data is to be supplied.
DirectionID (RichtungsID):	(optional) Filter for the feeder direction whose data is to be supplied.
EarliestArrivalTime (FruehesteAnkunftszeit):	Defines the start of the time frame for which data is to be supplied. The reference is the arrival time of the feeder at the connection area.
LatestArrivalTime (SpaetesteAnkunftszeit):	Defines the end of the time frame for which data is to be supplied. The reference is the arrival time of the feeder at the connection area.

LineID (LinienID) and DirectionID (RichtungsID) are independently optional. It is therefore possible to specify a request that only has one direction filter and no route filter.

The EarliestArrivalTime (FruehesteAnkunftszeit) element should not be set so that it lies before the local data horizon and LatestArrivalTime (SpaetesteAnkunftszeit) should not fall after the end of the horizon. The *ValidUntilTimeStamp (**VerfallZst)*** attribute should be the same time or later than *LatestArrivalTime (SpaetesteAnkunftszeit)*.

The following example describes a reference data request ('AVLC A' is the fetcher) for connection area '12345'. Only the feeder data of route 10 in the direction of 'Zoo' is to be supplied. The fetcher AVLC which has created the request has a data horizon from 5:00am to 11:00pm, 8.8.2001. The feeder AVLC only has one data horizon up to 10:00pm.

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T05:00:00">
      <AboASBRef AboID="25" VerfallZst="2001-08-09T00:00:00">
      <ASBID>12345</ASBID>
      <LinienID>10</LinienID>
      <RichtungsID>Zoo</RichtungsID>
      <FruehesteAnkunftszeit>
            2001-08-08T05:00:00
      </FruehesteAnkunftszeit>
      <SpaetesteAnkunftszeit>
            2001-08-08T23:00:00
      </SpaetesteAnkunftszeit>
      </AboASBRef>
</AboAnfrage>
```

The feeder responds with an Acknowledge (Bestaetigung) within a SubscriptionReply (AboAntwort) message. The restricted data horizon is signalled in DataValidUntil (DatenGueltigBis):

```
<AboAntwort>
      <Bestaetigung Zst="2001-08-08T05:00:05" Ergebnis="ok"
            Fehlernummer="0">
            <Datengueltigbis>2001-08-08T22:00:00</Datengueltigbis>
      </Bestaetigung>
</AboAntwort>
```

6.2.3.3 Transferring timetables (CPISchedule)

After the reference data subscriptions have been set up (6.2.3.2), the feeder system establishes the departure boards and signals this with the initial *DataReadyRequest (DatenBereitAnfragen)* message:

```
<DatenBereitAnfrage Sender="AVLC B" Zst="2001-08-08T05:01:00">
</DatenBereitAnfrage>
```

Reception is reported by the fetcher with an Acknowledge (Bestaetigung) within the DataReadyReply (DatenBereitAntwort) message:

```
<DatenBereitAntwort>
      <Bestaetigung Zst="2001-08-08T05:01:01" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
</DatenBereitAntwort>
```

The fetcher then polls the data with a *DataSupplyRequest (DatenAbrufenAnfrage):*

```
<DatenAbrufenAnfrage Sender="AVLC A" Zst="2001-08-08T05:01:05">
</DatenAbrufenAnfrage>
```

The feeder replies to this message with the desired data within CPISchedule (ASBFahrplan) elements. In turn, the *CPISchedule (ASBFahrplan)* element is a sub-element of the so-called *FeederMessage (Zubringernachricht)* (6.2.4.3)) and corresponds to a concrete arrival at a connection area. A FeederMessage (Zubringernachricht) is directly assigned to a subscription.

The arrivals board is therefore derived from a list of several CPISchedule (ASBFahrplan) elements.

Definition of CPISchedule (ASBFahrplan):

TimeStamp (Zst)	(attribute) Time at which the timetable data changed.
CPIID (ASBID):	Reference to the connection area.
TripID (FahrtID):	(sub-element) References a feeder trip.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips. Value increases with each subsequent passage.
LineID (LinienID):	Route code of the feeder
LineText (LinienText):	Description (name) of the route (relevant to the passenger)
DirectionID (RichtungsID):	Direction code of the feeder
DirectionText (RichtungsText):	Description of the direction: Passenger-relevant destination text of the feeder (vehicle signage on the trip section at approach trip to connection area. This corresponds to the end of the trip or the next intermediate destination, e.g. for circular routes).
FromDirectionText (VonRichtungsText):	(optional) Text for the passenger informing them where the feeder trip originated. This corresponds to the origin of the trip or the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination, e.g. for circular routes). No specification: The source text from the annual timetable is used (per LineID and DirectionID).
ScheduledCPIArrivalTime (AnkunftszeitASBPlan):	Planned arrival time of the feeder in the connection area.
StopID (HaltID):	(optional) References the stopping position within the connection area.
ArrivalPlatformText (AnkunftssteigText)	(optional) Specification of the platform or stand descriptor for the arrival of the trip at the stop.

StopPositionText (HaltepositionsText):	(optional) Planned stopping position of the fetcher vehicle in the connection area.
ArrivalSectorText (AnkunftsSektoren-Text)	(optional) Information on the sectors for arrival at a platform/track
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the feeder trip.

6.2.3.3.1 Additional information about the trip (TripInfo)

Information in the TripInfo (FahrtInfo) element has no functionality within the context of the connection protection. It is therefore entirely optional and serves primarily as additional information for the dispatcher, possibly for the purpose of telephone conversations with other control centres or logging purposes.

Definition of TripInfo (FahrtInfo):

VehicleID (FahrzeugID):	(optional) Internal AVLC label of the vehicle.
LineNumber (LinienNr):	(optional) Internal AVLC route number of the trip. Can deviate from the LineID (LinienID) as it is a meta datum.
BlockNumber (UmlaufNr):	(optional) Internal AVLC block number of the trip.
RunNumber (KursNr):	(optional) Internal AVLC run number of the trip.
DepartureStopLong (StartHstLang):	(optional) Full name of the start stop.
DepartureStop (StartHst):	(optional) Code name of the start stop.
DestinationStopLong (ZielHstLang):	(optional) Full name of the last stop.
DestinationStop (ZielHst):	(optional) Code name of the last stop.
PatternID (LinienfahrwegID):	(optional) The unique pattern code of the current pattern sequence (in association with a given route) stored in the on-board computer of a vehicle corresponds to the "ROUTE_NO (ROUTEN_NR)" element of the "LINE (REC_LID)" table in the VDV452 standard or a generated code in the case of a pattern that has been created spontaneously in the control centre.
DepartureTimeStartStop (AbfahrtszeitStartHst):	(optional) Planned departure time of the trip from the first stop.
ArrivalTimeDestinationStop (AnkunftszeitZielHst):	(optional) Planned arrival time of the trip at the destination stop.
ProductID (ProduktID):	(optional) Unique reference to the product.

OperatorID (BetreiberID):	(optional) This value is a meta datum. The OperatorID (BetreiberID) is a code describing the managing transport agency. This code allows routes from specific operators to be filtered out. With the OperatorID (BetreiberID) it is also possible to assign responsibilities for additional functions such as, for example, bookings and seat reservations.
Operator (Betreiber):	(Optional) Name of the (sub) operator running the trip. (Element no longer in use, please replace with *OperatorID (BetreiberID)*.)
ServiceFeature (ServiceMerkmal):	(optional, multiple) Specifies that the trip has the agreed service feature.
DirectCall (Direktruf):	(sub-element, optional) Information for direct communication with the vehicle

The specified ServiceFeatures (ServiceMerkmal) allow an operation to distinguish special trips, such as school buses and wheelchair access vehicles, amongst others.

A typical response to the request 6.2.3.2could read as follows:

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T05:01:07"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Zubringernachricht AboID="25">
            <ASBFahrplan Zst="2001-08-08T05:01:05">
                  <ASBID>12345</ASBID>
                  <FahrtID>
                        <FahrtBezeichner>64356</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <HstSeqZaehler>1</HstSeqZaehler>
                  <LinienID>10</LinienID>
                  <LinienText>X10</LinienText>
                  <RichtungsID>Zoo</RichtungsID>
                  <RichtungsText>Zoologischer Garten</RichtungsText>
                  <AnkunftszeitASBPlan>
                        2001-08-08T06:10:00
                  </AnkunftszeitASBPlan>
                  <FahrtInfo>
                        <VehicleID>34567</VehicleID>
                        <LinienNr>10</LinienNr>
                        <ServiceMerkmal>Schuelerfahrt</ServiceMerkmal>
                        <ServiceMerkmal>
                              behindertengerecht
                        </ServiceMerkmal>
                        <ServiceMerkmal>klimatisiert</ServiceMerkmal>
                        <ProduktID>Expressbus</ProduktID>
                        <Betreiber>BVG</Betreiber>
                        <Direktruf>
                              <Telefonnummer>
                                    +4917633445566
                              </Telefonnummer>
                        </Direktruf>
```

```
                  </FahrtInfo>
            </ASBFahrplan>
            <ASBFahrplan Zst="2001-08-08T05:01:06">
                  <ASBID>12345</ASBID>
                  <FahrtID>
                        <FahrtBezeichner>44347</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <HstSeqZaehler>1</HstSeqZaehler>
                  <LinienID>10</LinienID>
                  <LinienText>X10</LinienText>
                  <RichtungsID>Zoo</RichtungsID>
                  <RichtungsText>Zoologischer Garten</RichtungsText>
                  <AnkunftszeitASBPlan>2001-08-08T06:20:00</AnkunftszeitAS-
BPlan>
            </ASBFahrplan>
      </Zubringernachricht>
</DatenAbrufenAntwort>
```

In this situation exactly two trips are supplied, both on route X10 in the direction of the 'Zoological Garden' with arrival times 6:10 am and 6:20 am.

6.2.3.3.2 Information for direct communication (DirectCall)

The DirectCall (Direktruf) element contains information concerning the possibilities of direct communication with the vehicle or driver. This allows direct communication with the vehicle, e.g. at the end of business hours of the responsible control centre or as a result of last minute requirements of a third-party control centre.

Definition of DirectCall (Direktruf):	
PhoneNumber (Telefonnummer):	(optional) Telephone number (with international dialling code) with which the vehicle can be called directly.
IP address:	(optional) IP address or IP number for direct calls to the vehicle.

6.2.4 Process data service (CP)

The connection protection process data service is used for the exchange of current timetable statuses and events that are relevant to the passenger in connection with the connection management. Within the process data service, a differentiation must be made between the

- trip-based connection protection and
- time-based connection protection

procedures.

The trip-based connection protection facilitates the request and exchange of data for exactly one, previously defined feeder.

The time-based connection protection is used to request and exchange data from one or more vehicles which approach a connection area as potential feeders. The restriction of the data is

achieved via route, direction and a time window, in which the arrivals of the approaching trips must lie.

Both methods are implemented within the service with different filter requests. The exchange of data occurs using identical messages and data structures. Differences in behaviour and in the number of results stem from the different requirements of the two connection protection methods.

6.2.4.1 Data exchange

Data exchange in the process data service is initiated by the fetcher system. It creates a subscription, which specifies the trip data that it wishes to retrieve. This can be an individual trip (trip-based connection protection) or a group of trips that reach the connection area with a time window based on the predictions.

If the subscription is set up with the feeder, it answers immediately with the current subscription data (CPIDeviation (ASBFahrplanlage)). This data includes predictions or planning data in the absence of available prediction data.

If the process data relating to the trips assigned to the subscription changes (feeder timetable status, dispatch actions etc.), the feeder declares the existence of new data (DataReadyRequest (DatenBereitAnfrage)).

The fetcher can now explicitly request (DataSupplyRequest (DatenAbrufenAnfrage)) the updated data (only this data is sent). Following the loss of data, it is possible to re-request the entire data set (*AllData (DatensatzAlle*), please refer to 0 for more information).

Furthermore, there is also the possibility of initiating the flow of information in the opposite direction and communicate feedback messages from the fetcher to the feeder within the context of a subscription (6.2.4.4).

6.2.4.1.1 Updating / hysteresis

A process data subscription is considered to have been changed in connection with the connection protection if at least one of the following properties has changed:

- Timetable deviation
- Connection cannot take place (the feeder has not or cannot reach the connection area in time)
- Arrival at the connection area

A change remains active until the explicit retrieval of the data assigned to the subscription resets this status.

In order to avoid constant updates on the basis of minimal changes to the predicted arrival time, a desired hysteresis value (threshold) can be defined within the subscription and passed onto the feeder. This defines the time span after which a change is considered significant enough by the system to warrant communication. With a threshold value of 120 seconds, for example, any changes of 2 minutes or more with regard to the last transmitted arrival time are

reported. The threshold value however is only a recommendation and can be ignored by the data producing system.

Updating ends after the validity period has expired (ValidUntilTime is exceeded) or after the subscription is deleted.

6.2.4.1.2 Preview time

In addition to the hysteresis value (threshold), the preview time parameter also further restricts the transmission of subscription data in connection with trip-based connection protection. It restricts the data transmission to the time frame in which the connection monitoring is active for the fetcher. This is particularly important for longer trips (e.g. national travel).

The preview time defines the time window before the actual arrival of the feeder at the connection area within which its timetable status information is to be sent. If the vehicle is not yet on the approach at this point in time, planning information is sent.

The preview time is optional. If it is not specified, the time points relating to the subscription set-up are used (trip-based) or EarliestArrivalTime (FruehesteAnkunftszeit).

6.2.4.2 Retrieving the connection data (CPISubscription)

The subscription of process data generally occurs as soon as possible after receiving the timetable data from the reference data service.

The fetcher initiates communication. It creates a SubscriptionRequest (AboAnfrage) with an embedded CPISubscription (AboASB) element: Within the CPISubscription (AboASB) element there are two sub-elements, TripFilter (FahrtFilter) and TimeFilter (ZeitFilter), which define the corresponding filters. Within a CPISubscription (AboASB) it is possible to define a single TimeFilter (ZeitFilter) or one or more TripFilter (FahrtFilter). Both together are not permitted. It is necessary to specify at least one filter.

Definition of CPISubscription (AboASB):

SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription for feeder data created by the request. The SubscriptionID (AboID) is given by the fetcher system.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the subscription is valid.
CPIID (ASBID):	Reference to the connection area.
TripFilter (FahrtFilter):	(sub-element, alternative, multiple) Defines the data request for a feeder trip.
TimeFilter (ZeitFilter):	(sub-element, alternative) Defines a time window for the data request.
Hysteresis (Hysterese):	Desired change in seconds above which an update is to be communicated to the fetcher. The deviation must be greater than or equal to the specified value before deviations are transferred.
FetcherInfo (AbbringerInfo):	(optional) References the fetcher trip.

6.2.4.2.1 Trip-based data ((TripFilter)

The *TripFilter (Fahrtfilter)* element itself only specifies the feeder, as well as the planned arrival time (branch trip problems, 6.1.8). Once this preview time is reached, the timetable statuses of the requested vehicles are sent and then retransmitted after every change (CPIDeviation (ASBFahrplanlage)).

Definition of TripFilter (FahrtFilter):

TripID (FahrtID):	(sub-element) Unique code for the trip.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips. Value increases with each subsequent passage. Not sequential.
ScheduledCPIArrivalTime (AnkunftszeitASBPlan):	Planned arrival time of the trip in the connection area.
PreviewTime (Vorschauzeit):	Time in minutes before the planned arrival time at which the transmission of feeder predictions is begun.

In the following example, an operation (AVLC A) requests data from feeder vehicle '7748' at connection area '12345' for fetcher trip '2208'. The expiry time is then set to the planned departure time of the fetcher (4:00pm) plus the maximum waiting time (10 minutes). This guarantees a data supply even in the case of severely delayed fetchers. The transmission of timetable statuses should be started 30 minutes before the arrival of the trip in the connection area.

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T05:10:00">
      <AboASB AboID="25" VerfallZst="2001-08-08T16:10:00">
            <ASBID>12345</ASBID>
            <FahrtFilter>
                  <FahrtID>
                        <FahrtBezeichner>7748</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <HstSeqZaehler>1</HstSeqZaehler>
                  <AnkunftszeitASBPlan>
                        2001-08-08T16:00:00
                  </AnkunftszeitASBPlan>
                  <Vorschauzeit>30</Vorschauzeit>
            </FahrtFilter>
            <Hysterese>120</Hysterese>
            <AbbringerInfo>
                  <FahrtID>
                        <FahrtBezeichner>2208</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <LinienID>3</LinienID>
                  <LinienText>3</LinienText>
                  <RichtungsID>HBF</RichtungsID>
                  <RichtungsText>Hauptbahnhof</RichtungsText>
                  <AbfahrtszeitASBPlan>
                        2001-08-08T16:00:00
                  </AbfahrtszeitASBPlan>
            </AbbringerInfo>
      </AboASB>
</AboAnfrage>
```

The feeder confirms reception of the message with an Acknowledge (Bestaetigung) element in the SubscriptionReply (AboAntwort).

6.2.4.2.2 Time-based data (TimeFilter)

The requesting of time-based connection subscriptions is analogous to the trip-based subscription process. The only difference is in the specification of exactly one TimeFilter (Zeitfilter). This defines the feeder routes / directions for which the trips are to supply their timetable statuses and messages. The filter criteria are optional, i.e. it is possible to request all directions of a route, all routes in a particular direction or all trips approaching a particular connection area. The two times within the filter define the time widow, within which the trips must reach the connection area in order to be reported. Trips that have been sent once already, continue to be transmitted even if they no longer fall within the time window.

PreviewTime:

Without specification of a preview time, the transmission of the timetable statuses that match the given time period is begun as soon as the subscription is received in the feeder system. If a receiver system is subscribing to a very large time period (e.g. daily from 3.30am for 25 hours), it is debatable as to when exactly the data supplier should commence with the transmission of the feeder data. Many manufacturers send predictions as soon as they are available in the system.

With the transmission of the preview time, this defines the time in minutes prior to the planned or predicted arrival time of the feeder trip at which transmission of feeder predictions is to start. In this situation, timetable statuses are not sent until the preview time is reached and thereafter upon every change to the feeder trip.

Implementation notes:

An initial feeder message should be sent when one of the following conditions is satisfied:

- if the planned arrival of the feeder lies within the subscription time period,
- if the planned arrival time of the feeder falls into the preview window, if the subscription contains a preview time.

In order to be able to protect connections that arise spontaneously as a direct result of delays, an initial feeder message should be sent when the same conditions for the predicted arrival are satisfied.

Definition of the TimeFilter (Zeitfilter):	
LineID (LinienID):	(optional) Code of the feeder route, for which feeder data is to be provided.
DirectionID (RichtungsID):	(optional) Direction code of the feeder route, for which feeder data is to be provided.

EarliestArrivalTime (FruehesteAnkunftszeit):	Start of the arrival time window for which feeder data is to be supplied.
LatestArrivalTime (SpaetesteAnkunftszeit):	End of the arrival time window for which feeder data is to be supplied.
PreviewTime (Vorschauzeit):	(optional) Time in minutes before the planned arrival time of the feeder at which the transmission of feeder predictions is begun.

In the following example, feeder data of the trips on route 2, in the direction of 'Central Station' is to be retrieved for a trip approaching a connection area (AVLC A). Data should only be sent for vehicles that arrive in the connection area between 3:50pm and 4:10pm in accordance with the current predictions.

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T15:45:00">
      <AboASB AboID="25" VerfallZst="2001-08-08T16:10:00">
            <ASBID>12345</ASBID>
            <ZeitFilter>
                  <LinienID>2</LinienID>
                  <RichtungsID>Bahnhof</RichtungsID>
                  <FruehesteAnkunftszeit>
                        2001-08-08T15:50:00
                  </FruehesteAnkuntszeit>
                  <SpaetesteAnkunftszeit>
                        2001-08-08T16:10:00
                  </SpaetesteAnkuntszeit>
            </ZeitFilter>
            <Hysterese>120</Hysterese>
      </AboASB>
</AboAnfrage>
```

Timetable statuses should only be sent if the change in the delay is at least 120 seconds.

6.2.4.2.3 Additional information on the fetcher (FetcherInfo)

The optional FetcherInfo (AbbringerInfo) element is used to supply the interior signs in the feeder vehicles with the planning data (fetcher displays).

Definition of FetcherInfo (AbbringerInfo):	
TripID (FahrtID):	(sub-element) Unique reference to the fetcher trip.
StopSeqCount (HstSeqZaehler):	(optional) Passage counter of the feeder for branch trip detection. Value increases with each subsequent passage.
LineID (LinienID):	Route code of the fetcher
LineText (LinienText):	Route description of the fetcher (relevant for the passengers)
DirectionID (RichtungsID):	Direction code of the fetcher

DirectionText (RichtungsText):	Description of the direction: Destination text of the fetcher vehicle that is relevant to the passenger (corresponds to the vehicle signage along this stretch of route at departure from the connection area, valid until the end of the trip or in the case of circular routes until the next intermediate destination).
FromDirectionText (VonRichtungsText):	(optional) Text for the passengers informing them where the trip originated. In the case of circular routes with intermediate destinations, it is recommended to indicate the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination) from the stop after the first intermediate destination until the terminal stop. No specification: The text from the annual schedule (per LineID (LinienID) and DirectionID (RichtungsID)) is used.
ScheduledCPIDepartureTime (AbfahrtszeitASBPlan):	Planned departure time of the fetcher in the connection area.
StopID (HaltID):	(optional) References the stopping position within the connection area.
DeparturePlatformText (AbfahrtssteigText)	(optional) Specification of the platform or stand descriptor for the departure of the trip from the stop
StopPositionText (HaltepositionsText):	(optional) Planned stopping position of the fetcher vehicle in the connection area.
DepartureSectorText (AbfahrtsSektorenText)	(optional) Information on the sectors for departure at a platform/track
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the fetcher trip.

6.2.4.3 Feeder messages (FeederMessage)

All messages that are transferred from the feeder to the fetcher are encapsulated within the FeederMessage (Zubringernachricht) element. This element does not contain any data elements. Instead, it logically groups all message elements that are sent from the feeder to the fetcher.

The following messages can be reported within the FeederMessage (Zubringernachricht):

In the reference data service:

- Connection area timetable data (CPISchedule (ASBFahrplan))

In the process data service:

- Current timetable statuses (CPIDeviation (ASBFahrplanlage))
- Cancellation of a trip (CPITripDelete (ASBFahrtLoeschen))

Definition of FeederMessage (Zubringernachricht):

SubscriptionID (AboID):	(attribute) Unique reference to the subscription within the service.
CPISchedule (ASBFahrplan):	(sub-element, optional, multiple) Contains the data of the area timetable of the feeder.
CPIDeviation (ASBFahrplanlage):	(sub-element, optional, multiple) Contains the data relating to the current timetable status of the feeder.
CPITripDelete (ASBFahrtLoeschen):	(sub-element, optional, multiple) Reports the cancellation of the feeder trip.

6.2.4.3.1 Transmit the connection data (CPIDeviation)

Once the subscription is set up, the feeder initially gathers the data for the desired trip (trip-based subscription) or for the desired time frame (time-based subscription). It signals the readiness to transmit data with a *DataReadyRequest (DatenBereitAnfrage*). The fetcher confirms reception by means of an *Acknowledge (Bestaetigung)* element in *DataReadyReply (DatenBereitAntwort*), see 6.2.3.3. With time-based connection protection, the time of the initial signalling occurs just after the subscription has been set up. With the trip-based subscriptions the first notification occurs after reaching the PreviewTime (Vorschauzeit).

The fetcher now specifically requests all data of all subscriptions that have changed since the last DataReadyRequest (DatenBereitAnfrage).

```
<DatenAbrufenAnfrage Sender="AVLC A" Zst="2001-08-08T05:10:00">
</DatenAbrufenAnfrage
```

In return, the fetcher gets a DataSupplyAnswer (DatenAbrufenAntwort), which contains a FeederMessage (Zubringernachricht) element for every modified subscription. The FeederMessage (ZubringerNachricht) contains a CPIDeviation (ASBFahrplanlage) element with the status information of the trip:

Definition of CPIDeviation (ASBFahrplanlage):

TimeStamp (Zst)	(attribute) Timestamp of the timetable status recording.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the data is valid.
CPIID (ASBID):	Reference to the connection area.
TripID (FahrtID):	(sub-element) References the feeder trip.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips. Value increases with each subsequent passage. Not sequential.
LineID (LinienID):	Route code of the feeder
LineText (LinienText):	Route description of the feeder (relevant for the passengers)
DirectionID (RichtungsID):	Direction code of the feeder

DirectionText (RichtungsText):	Description of the direction: Passenger-relevant destination text of the feeder (vehicle sign on the trip section at approach trip to connection area. This corresponds to the end of the trip or the next intermediate destination, e.g. for circular routes).
FromDirectionText (VonRichtungsText):	(optional) Text for the passenger informing them where the feeder trip originated. This corresponds to the origin of the trip or the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination, e.g. for circular routes). No specification: The source text from the annual timetable is used (per LineID and DirectionID).
AtCPIPoint (AufASB):	(optional) Flag that indicates the vehicle has entered the connection area.
ScheduledCPIArrivalTime (AnkunftszeitASBPlan):	Planned arrival time of the feeder in the connection area.
ExpectedCPIArrivalTime (AnkunftszeitASBPrognose):	(optional) Predicted arrival time of the feeder at the stop of the connection area. This entry is mandatory if TripStatus (FahrtStatus) = 'actual'. This can be omitted if TripStatus (FahrtStatus) = 'planned' and there is no prediction for the arrival time.
TripStatus (FahrtStatus):	Specifies whether actual (real-time) information can be supplied for the vehicle (“actual”) or not (“planned”).
TransferPassengers (Umsteigewillige):	(optional) Number of transfer passengers. Negative if the information cannot be supplied.
FeederStopLong (ZubringerHstLang):	(optional) Full name of the feeder stop.
LatestFetcherInfoTime (SpaetesteAbbringerInfo):	(optional) Time to which the fetcher information is to be transmitted.
StopID (HaltID):	(optional) References the stopping position within the connection area.
ArrivalPlatformText (AnkunftssteigText)	(optional) Specification of the platform or stand descriptor for the arrival of the trip at the stop
StopPositionText (HaltepositionsText):	(optional) Describes the new stopping position.
ArrivalSectorText (AnkunftsSektorenText)	(optional) Information on the sectors for arrival at a platform/track
QueueIndicator (Stauindikator):	(optional) States whether the vehicle is stuck in traffic (true) or not (false).
TripInfo (FahrtInfo):	(sub-element, optional) Supplies additional information relating to the trip

The LatestFetcherInfoTime (SpaetesteAbbringerInfo) element serves to inform the fetcher system when the feeder system wants to use the information from the reverse channel for the passenger information. This allows the fetcher system to provide the information required for dispatcher decisions, in order for example, to establish the latest time for manual intervention by the dispatcher.

Example of a trip-based subscription

The following example shows the response to a trip-based subscription for a feeder with a planned arrival time of 3:58 pm (for the associated subscription example please see 6.2.4.2.1). Upon reaching the 30-minute preview time, a timetable status message is sent for the first time. Predictions are not possible at the time of sending the timetable status (3:28pm), as the vehicle is not yet on the path (TripStatus = planned, Prediction = planned).

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:28:00" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Zubringernachricht AboID="25">
      <ASBFahrplanlage  Zst="2001-08-08T05:10:05" >
                        VerfallZst="2001-08-08T16:00:00">
            <ASBID>12345</ASBID>
            <FahrtID>
                  <FahrtBezeichner>7748</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
            </FahrtID>
            <HstSeqZaehler>1</HstSeqZaehler>
            <AnkunftszeitASBPlan>
                  2001-08-08T15:58:00
            </AnkunftszeitASBPlan>

            <FahrtStatus>Soll</FahrtStatus>
            <AufASB>false</AufASB>
            <LinienID>10</LinienID>
            <LinienText>X10</LinienText>
            <RichtungsID>Zoo</RichtungsID>
            <RichtungsText>Zoologischer Garten</RichtungsText>
      </ASBFahrplanlage>
      </Zubringernachricht>
</DatenAbrufenAntwort>
```

Once the trip has begun and the timetable deviation exceeds 120 seconds (desired hysteresis value in the CPISubscription (AboASB)), this is signalled by the feeder (DataReadyRequest (DatenBereitAnfrage)). The request is made again via DataSupplyRequest (DatenAbrufenAnfrage) initiated by the fetcher. A possible reply could read as follows:

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:40:00" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Zubringernachricht AboID="25">
      <ASBFahrplanlage  Zst="2001-08-08T15:39:00">
                        VerfallZst="2001-08-08T16:05:00">
            <FahrtID>
                  <FahrtBezeichner>7748</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
            </FahrtID>
```

```
            <HstSeqZaehler>1</HstSeqZaehler>
            <AnkunftszeitASBPlan>
                  2001-08-08T15:58:00
            </AnkunftszeitASBPlan>
            <AnkunftszeitASBPrognose>
                  2001-08-08T16:00:00
            </AnkunftszeitASBPrognose>
            <HaltID>B3</HaltID>
            <Haltepositionstext>Bussteig B</Haltepositionstext>
            <FahrtStatus>Ist</FahrtStatus>
            <AufASB>false</AufASB>
            <LinienID>10</LinienID>
            <LinienText>X10</LinienText>
            <RichtungsID>Zoo</RichtungsID>
            <RichtungsText>Zoologischer Garten</RichtungsText>
      </ASBFahrplanlage>
      </Zubringernachricht>
</DatenAbrufenAntwort>
```

This procedure is repeated until the vehicle has reached the stop (last message with AtCPIPoint (AufASB) = true), or the fetcher has deleted the subscription.

Implementation notes:

Once the ValidUntilTimeStamp (VerfallZst) has been reached, the trips are to be removed from the connection protection, even if this time point lies before the ExpectedCPIArrivalTime.

Implementation notes:
The feeder system is always obliged to send the last message with AtCPIPoint (AufASB) = true. If, for any reason, it is not technically possible to detect that the vehicle has reached the stopping position (e.g. in railway operation), a robust prediction should be made and transmitted by the feeder system, as it is in a better position than the fetcher system to make a realistic estimate.

Example of a time-based subscription

In the following example, the feeder responds within the frame of a time-based subscription to the *DataSupplyRequest (DatenAbrufenAnfrage)* with the timetable status of two trips (for an associated subscription example please refer to 6.2.4.2.2):

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:45:30" Ergebnis="ok" Fehlernum-
mer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Zubringernachricht AboID="25">
      <ASBFahrplanlage  Zst="2001-08-08T15:45:10">
                        VerfallZst="2001-08-08T16:00:00">
            <ASBID>12345</ASBID>
            <FahrtID>
                  <FahrtBezeichner>7748</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
            </FahrtID>
```

```
            <HstSeqZaehler>1</HstSeqZaehler>
            <AnkunftszeitASBPlan>
                  2001-08-08T15:48:00
            </AnkunftszeitASBPlan>
            <AnkunftszeitASBPrognose>
                  2001-08-08T15:53:00
            </AnkunftszeitASBPrognose>
            <HaltID>B3</HaltID>
            <Haltepositionstext>Bussteig 3</Haltepositionstext>
            <FahrtStatus>Ist</FahrtStatus>
            <AufASB>false</AufASB>
            <LinienID>2</LinienID>
            <LinienText>2</LinienText>
            <RichtungsID>Bahnhof</RichtungsID>
            <RichtungsText>Bahnhof</RichtungsText>
      </ASBFahrplanlage>
      <ASBFahrplanlage Zst="2001-08-08T15:45:10">
            <ASBID>12345</ASBID>
            <FahrtID>
                  <FahrtBezeichner>6611</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
            </FahrtID>
            <HstSeqZaehler>1</HstSeqZaehler>
            <AnkunftszeitASBPlan>
                  2001-08-08T16:05:00
            </AnkunftszeitASBPlan>
            <AnkunftszeitASBPrognose>
                  2001-08-08T16:06:00
            </AnkunftszeitASBPrognose>
            <FahrtStatus>Ist</FahrtStatus>
            <AufASB>false</AufASB>
            <LinienID>2</LinienID>
            <LinienText>2</LinienText>
            <RichtungsID>Bahnhof</RichtungsID>
            <RichtungsText>Bahnhof</RichtungsText>
      </ASBFahrplanlage>
      </Zubringernachricht>
</DatenAbrufenAntwort>
```

Trip 7748 is an example of a trip that only falls within the time window as a result of it's severe delay.

Change messages are processed in a similar way to trip-based connection protection. It should be noted that once a trip has fallen within the time range, it remains there for the entire life-span of the subscription, which means it is constantly updated and transmitted. The number of reported timetable statuses can therefore increase throughout the life of the subscription (3.1.9).

6.2.4.3.2 Cancellation of a feeder (CPITripDelete)

Feeders may be cancelled for several reasons:

- Breakdown of the feeder vehicle

- Diversion of the feeder route/direction
- Obstruction of a route section
- Short turn action

A feeder cancellation therefore denotes the cancellation or lateness of the scheduled stop at the connection stop. This can lead to the fetcher abandoning the connection.

Notification occurs in a similar way to reporting timetable statuses (6.2.4.3.1). The occurrence of one of the above-mentioned events is interpreted as a change. The associated message is called CPITripDelete (ASBFahrtLoeschen) and is a sub-element of the FeederMessage (Zubringernachricht).

Definition of CPITripDelete (ASBFahrtLoeschen):

TimeStamp (Zst)	(attribute) Time stamp of the dispatch action
CPIID (ASBID):	Reference to the connection area.
TripID (FahrtID):	(sub-element) References the cancelled feeder trip.
StopSeqCount (HstSeqZaehler):	Passage counter of the feeder for branch trip detection. Value increases with each subsequent passage.
LineID (LinienID):	References the feeder route
LineText (LinienText):	Route description of the feeder (relevant for the passengers)
DirectionID (RichtungsID):	Direction code of the feeder
DirectionText (RichtungsText):	Direction of the feeder (relevant to the passenger)
FromDirectionText (VonRichtungsText)	(optional) Passenger-relevant origin text
ScheduledCPIArrivalTime (AnkunftszeitASBPlan)	(optional) Planned arrival time of the feeder in the connection area
StopID (HaltID)	(optional) References the stopping position within the connection area.
HaltepositionsText	(optional) Describes the new stopping position.
TripInfo (FahrtInfo)	(sub-element, optional) Supplies additional information relating to the trip.
Reason (Ursache):	(optional) Description of the cause of the error

Since it is not always possible to reference the trip using the TripID (FahrtID), the optional fields *FromDirectionText (VonRichtungsText), ScheduledCPIArrivalTime (Ankunftszeit-ASBPlan), StopID (HaltID), StoppingPositionText (HaltepositionsText) and TripInfo (FahrtInfo)* shall be filled in if possible.

The following example indicates a possible message reported by a feeder in case of cancellation of trip 6611 at connection area 12345 at 3:55pm on route 12:

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:56:00" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Zubringernachricht AboID="25">
      <ASBFahrtLoeschen Zst="2001-08-08T15:55:00">
            <ASBID>12345</ASBID>
            <FahrtID>
                  <FahrtBezeichner>6612</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
            </FahrtID>
            <HstSeqZaehler>1</HstSeqZaehler>
            <LinienID>12</LinienID>
            <LinienText>12</LinienText>
            <RichtungsID>TRPK</RichtungsID>
      <RichtungsText>Tierpark</RichtungsText>
            <Ursache>Fahrzeugausfall</Ursache>
      </ASBFahrtLoeschen>
      </Zubringernachricht>
 </DatenAbrufenAntwort>
```

It should be noted that with the cancellation message from the feeder, the trip itself cannot be deleted from the subscription. After re-initialising the fetcher following a loss of data (DataSupplyRequest (DatenAbrufenAnfrage)), the cancellation message must be re-sent in place of a CPIDeviation (ASBFahrplanlage).

6.2.4.4 Fetcher messages (FetcherMessage)

In addition to the information that is conveyed from the feeder to the fetcher, there are further messages that are transmitted in the opposite direction, from the fetcher to the feeder.

The messages of the reverse channel are provided with the code of one or more feeder trips to facilitate targeted notification.

These fetcher messages need not be retrieved explicitly; instead they represent the reverse channel of an existing subscription.

The following fetcher messages are defined:

- Change to the departure location in the connection area
- Change to the fetcher dispatch status (waiting time)
- Cancellation of the fetcher

Definition of the FetcherMessage (Abbringernachricht):

SubscriptionID (AboID):	(attribute) Unique reference to the subscription within the service.
StopPositionChange (HaltepositionsAenderung):	(sub-element, alternative, multiple) Reports a change in the departure location of the fetcher in the connection area.
WaitUntil (WartetBis):	(sub-element, alternative, multiple) Reports a change in the departure time of the fetcher.
FetcherTripDelete (AbbringerFahrtLoeschen):	(sub-element, alternative, multiple) Reports the cancellation of the fetcher trip in the connection area.

The fetcher signals a change to the process data associated with the subscription via a DataReadyRequest (DatenBereitAnfrage). The feeder confirms this and then requests the data with a DataSupplyRequest (DatenAbrufenAnfrage).

Within the DataSupplyAnswer (DatenAbrufenAntwort) / FetcherMessage (Abbringer-Nachricht), the fetcher now supplies one or more elements of the above listed fetcher messages.

6.2.4.4.1 Change to the stopping position (StopPositionChange)

If the fetcher changes its planned stopping position within the connection area, this can be relevant to the passenger. This function is used to convey information relating to the new connection path to the passengers.

The basis is an existing subscription (CPISubscription (AboASB)), which retrieves feeder data. Using the TripIDExt (FahrtIDExt), the message is addressed to one or more feeders. The TripID (FahrtID) of the fetcher serves as a reference. This guarantees that the interior signs in the feeders can also be updated with the new stopping position of the fetcher.

Definition of StopPositionChange (HaltepositionsAenderung):

TimeStamp (Zst):	(attribute) Time stamp of the dispatch action.
CPIID (ASBID):	References the connection area
FetcherInfo (AbbringerInfo):	(sub-element) Unique reference to the fetcher.
TripIDExt (FahrtIDExt):	(sub-element, multiple) References the feeder trip(s).

6.2.4.4.2 Referencing trips (TripIDExt)

In fetcher messages that are valid for one or more feeder trips, the *TripIDExt (FahrtIDExt)* element uniquely identifies the relevant feeder trips via *TripID (FahrtID)* or *TripName*

(FahrtBezeichner) as well as *OperationalDay (Betriebstag)* and *StopSeqCount (HstSeqZaehler)*.

Definition of TripIDExt (FahrtIDExt):	
TripName (FahrtBezeichner):	Unique label for a feeder trip within the operational day.
OperationalDay (Betriebstag):	Date of the operational day on which the trip is travelling.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips (feeder). Value increases with each subsequent passage.

The following example shows the message sent by the fetcher vehicle in the context of the subscription with SubscriptionID 25, stating that the fetcher vehicle (TripID 6612, route 8, in direction 'Schlossplatz') has changed its stopping position in the connection area (CPIID 12345) to 'platform 3'. The message is addressed to feeder trip 5467.

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:56:00" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Abbringernachricht AboID="25">
      <HaltepositionsAenderung Zst="2001-08-08T15:55:00">
            <ASBID>12345</ASBID>
            <AbbringerInfo>
                  <FahrtID>
                        <FahrtBezeichner>6612</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <LinienID>8</LinienID>
                  <LinienText>8</LinienText>
                  <RichtungsID>SCHLP</RichtungsID>
                  <RichtungsText>Schlosspark</RichtungsText>
                  <AbfahrtszeitASBPlan>
                        2001-08-08T16:00:00
                  </AbfahrtszeitASBPlan>
                  <HaltepositionsText>
                        Steig 3
                  </HaltepositionsText>
            </AbbringerInfo>
            <FahrtIDExt>
                  <FahrtBezeichner>5467</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
                  <HstSeqZaehler>1</HstSeqZaehler>
            </FahrtIDExt>
      </ HaltepositionsAenderung >
      </Abbringernachricht>
 </DatenAbrufenAntwort>
```

6.2.4.4.3 Prolonging the wait (WaitUntil)

An essential function of the connection protection is to hold back the fetcher vehicle in order to allow passengers on delayed feeders the chance to catch their connection. This so-called connection dispatch decides the duration of an unscheduled wait for the fetcher in a connection area.

This information should be communicated to the feeder in order to be able tell the passengers the new departure time of the waiting fetcher (via interior signs in the vehicle).

Any extension to the waiting time is reported to the feeder by means of the WaitUntil (Wartet-Bis) message within the FetcherMessage (Abbringernachricht).

Definition of WaitUntil (WartetBis):

TimeStamp (Zst):	(attribute) Time stamp of the wait decision
CPIID (ASBID):	References the connection area
FetcherInfo (AbbringerInfo):	(sub-element) References the fetcher trip.
TripIDExt (FahrtIDExt):	(sub-element, multiple) References the feeder trip(s).
ExpectedDepartureTime (AbfahrtszeitASBPrognose):	Predicted departure time of the fetcher vehicle at the connection area stop.
Reliability (Verlaesslichkeit):	(optional) Classifies the reliability / probability of the prediction (in stages from 1 to 5 where 1 is the highest and 5 is the lowest value).

Implementation notes:

If a fetcher is waiting for several feeders, a WaitUntil (WartetBis) is sent containing several TripIDExt (FahrtIDExt) sub-elements. If one of these feeders becomes so severely delayed that the fetcher is no longer able to wait, a FetcherTripDelete (AbbringerFahrtLoeschen) is sent with the corresponding TripIDExt (FahrtIDExt).

If an AVLC systems works partially without a planning data supply, then it can be useful for reasons of uniqueness to additionally send a WaitUntil (WartetBis) with the current ExpectedDepartureTime (AbfahrtszeitASBPrognose) to those fetcher vehicles that do not need to be held back. The same also goes for the situation in which a connection can be held back due to the fact that both participant vehicles have the same delay.

In the example below, fetcher 6612 of route 8 direction Schlosspark reports a new departure time of 4:02pm at connection area 12345 to feeder 5467.

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:56:00" Ergebnis="ok"
            Fehlernummer="0">
```

```
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Abbringernachricht AboID="25">
      <WartetBis Zst="2001-08-08T15:55:00">
            <ASBID>12345</ASBID>
            <AbbringerInfo>
                  <FahrtID>
                        <FahrtBezeichner>6612</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <LinienID>8</LinienID>
                  <LinienText>8</LinienText>
                  <RichtungsID>SCHLP</RichtungsID>
                  <RichtungsText>Schlosspark</RichtungsText>
                  <AbfahrtszeitASBPlan>
                        2001-08-08T16:00:00
                  </AbfahrtszeitASBPlan>
            </AbbringerInfo>
            <FahrtIDExt>
                  <FahrtBezeichner>5467</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
                  <HstSeqZaehler>1</HstSeqZaehler>
            </FahrtIDExt>
            <AbfahrtszeitASBPrognose>
                  2001-08-08T16:02:00
            </AbfahrtszeitASBPrognose>
      </WartetBis>
      </Abbringernachricht>
</DatenAbrufenAntwort>
```

6.2.4.4.4 Fetcher cancellation (FetcherTripDelete)

The absence of a fetcher simultaneously means the failure of the associated connection. The *FetcherTripDelete (AbbringerFahrtLoeschen*) message is used to signal this event to passengers in the feeder vehicle (via the interior signs), similar to (6.2.4.3.2), but in the opposite direction.

In order to prevent the further transmission of the timetable status of feeders, the associated subscription on the fetcher side must also be deleted.

Implementation notes:

If the fetcher trip changes within the context of dynamic connection protection, it is necessary to send a FetcherTripDelete (AbbringerFahrtLoeschen) together with the information about the new fetcher (WaitUntil (WartetBis)) in a FetcherMessage (Abbringernachricht) in order to avoid any unnecessary cancellation of the connection.

The semantics of the elements in *FetcherTripDelete (AbbringerFahrtLoeschen*) only differ slightly from *CPITripDelete (ASBFahrtLoeschen)* of the *FeederMessage* (*Zubringernachricht*):

Definition of FetcherTripDelete (AbbringerFahrtLoeschen):

TimeStamp (Zst):	(attribute) Time stamp of the dispatch action
CPIID (ASBID):	References the connection area
FetcherInfo (AbbringerInfo):	(sub-element) References the missing fetcher trip.
TripIDExt (FahrtIDExt):	(sub-element, multiple) References the feeder trip(s).
Ursache (Reason):	(optional) Description of the cause of the error

The following example indicates the message relating to cancellation of a fetcher (7712) of route 8 at 4:00pm at connection area 12345 for feeder 7748, whose data has been retrieved via SubscriptionID (AboID) 25.

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:56:00" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <Abbringernachricht AboID="25">
      <AbbringerFahrtLoeschen Zst="2001-08-08T15:55:00">
            <ASBID>12345</ASBID>
            <AbbringerInfo>
                  <FahrtID>
                        <FahrtBezeichner>7712</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <LinienID>8</LinienID>
                  <LinienText>8</LinienText>
                  <RichtungsID>SCHLP</RichtungsID>
                  <RichtungsText>Schlosspark</RichtungsText>
                  <AbfahrtszeitASBPlan>
                        2001-08-08T16:00:00
                  </AbfahrtszeitASBPlan>
            /AbbringerInfo>
            <FahrtIDExt>
                  <FahrtBezeichner>7748</FahrtBezeichner>
                  <Betriebstag>2001-08-08</Betriebstag>
                  <HstSeqZaehler>1</HstSeqZaehler>
            </FahrtIDExt>
      </AbbringerFahrtLoeschen>
      </Abbringernachricht>
 </DatenAbrufenAntwort>
```

6.3 Dynamic passenger information (REF-DPI, DPI)

6.3.1 Introduction

The dynamic passenger information service is used to match and transmit data for the operation of common passenger information signs at the stops. The following specification describes the exchange of full text information with regard to supplying third-party DPI signs. (Optimised data exchange using codes instead of full text is possible with the corresponding inter-agency agreements.) It is assumed in this context that one of the two participating agencies owns and controls the signs (sign owner).

The agency that wishes to display information on the third-party signs is referred to as the sign user.

Data exchange within the interface is based on full texts (destination, via stations). If the signs are to be controlled using codes, due to restrictions on the transmission capacities, the codes must be agreed by all parties and exchanged within the context of the data management. The display owner system must define and map the full texts and corresponding codes (6.3.3).

As some systems only permit the processing of process data if the corresponding planning data already exists, there is the option of a reference data service for the passenger information (REF-DPI). This permits, where necessary, the exchange of localised timetables for the passenger information.

6.3.2 Operational data supply and management

In a similar way to the connection protection function, the location codes must also be agreed between the agencies (6.1.3), here referred to as *display areas* (*DIS (AZB*)). Unique bilateral codes are agreed, so-called DISIDs (AZBID). The display owner system maps these onto the corresponding codes of the local DPI signs (or sign groups):

DISID (AZ-BID)	Third-party agency code	Internal DPI sign code
12345	AVLC B	2345
12346	AVLC B	3687
35678	AVLC B	7566

Table 12 - Definition of DPI location codes by the sign owner

The sign user maps the codes onto the local stops rather than the signs. This links a third-party stop with the associated DPI sign. When defining the *DISIDs (AZBIDs*), possible use within the log-off messages for the rapid cleardown function must be taken into consideration (6.3.5)

If a sign serves more routes than those to be displayed, a table must be created which assigns the relevant third-party routes to each particular sign.

DISID	Third-party agency code	Third-party route	Direction on the third-party route
12345	AVLC B	8	Zoo
12346	AVLC B	10	Station

Table 13 - Definition of the routes to be displayed

This requires an agreement on the product types. This can be relevant in connection with the use of symbols to represent operating areas or vehicle types on the DPI signs. The modelling of the product types should take this into consideration.

Differing field lengths (e.g. for the destination or 'via' information texts) create problems for the signs. It cannot be assumed that it is sufficient to manage a single field length. For each type of DPI sign, there must be a corresponding supply with full texts. The interface however provides a function that allows the user to specify a maximum length (MaxTextLength (MaxText-Laenge)), which supplies the optimum length where there are various different lengths to be accommodated. This function however does not solve all the related issues as some signs use proportional character sets. A sign does not therefore have a standard text length per line, but a number of points per line, which can represent texts of variable lengths. The management of full texts therefore demands an intensive consultation process and verification of the actual texts.

6.3.3 DPI systems with coded control

DPI systems for which there is limited transmission capacity for data supply via radio, often use predefined texts with associated codes instead of using full texts (direction, via stations). Only the shorter code is then used during communication between the controlling control centre and the DPI sign.

Full texts are however used exclusively within the interface. If coded control is also to be used for third-party vehicles, all predefined texts of the sign users must first be imported and then assigned corresponding codes. These must then be made known to the DPI sign systems during the data supply process. The conversion into full texts occurs in the sign owner system by means of a text comparison between the full text and the text/code table of the third-party operation.

6.3.4 DPI systems with autonomous sign predictions

The physical message channel between the AVLC and the DPI signs is often a very limited resource (e.g. narrow bandwidth radio network). The reference data service for DPI provides information that ensures optimum use of this limited resource. The idea is to eliminate any redundant information that arises from the targeted control of each individual DPI sign. To achieve this, the DPI signs are supplied with timetable information at the start of each operational day. During operation, instead of the departure time predictions for the signs served by a particular trip only the delay information is sent in broadcast mode. The DPI signs filter out the relevant information and from the planning data and the current timetable status are able to determine the predicted arrival time. In this situation we refer to 'intelligent DPI signs'.

The control of autonomous signs requires trip-based planning data as well as real-time data. The DPI interface however only provides location-based planning data or real-time data, which means it is not able to serve systems controlling autonomous signs.

It is possible to use the timetable information interface (VDV Recommendation 454) for such DPI systems, as this interface is capable of providing trip or route-specific data.

6.3.5 Rapid cleardown

The so-called 'rapid cleardown' function is used in order to be able to remove trip information from a sign very quickly after the vehicle has departed. To enable this, the vehicle must be able to communicate directly with the stop / sign. TLP radio technology is generally used.

Once the vehicle leaves the stop it sends a radio message with the trip code. The sign then deletes the associated trip information.

The code used within the interface to identify the trip (TripID (FahrtID)) cannot usually be used here: This is an internal AVLC code, which is not stored in the vehicle's data supply.

In order to be able to implement rapid cleardown a so-called DepartureNoticeID (AbmeldeID) is added to the TripID (FahrtID) in every relevant message. This is registered in the vehicles of the sign user system and once received by the sign owner system is made known to the sign within the context of supplying the timetable information.

Contrary to the TripID (FahrtID), the DepartureNoticeID (AbmeldeID) is only unique on the operational day of the sign user system. The sign must compensate the limiting cases using additional logics. The signs can then assign trips with the same DepartureNoticeID (AbmeldeID) according to the current time and the planned departure time.

In addition to the DepartureNoticeID (AbmeldeID), which identifies the vehicle or trip, a location reference is also often used. This defines the sign from which the trip is to be deleted. This differentiation is necessary if, for example, there are several signs in the radio area of the vehicle. The DISID (AZBID) is used as standard for this purpose. If it is not possible to use this due to the restrictions of existing systems (type, format, dimensions) then in addition to the *DISID (AZBID)*, the sign reference of the sign owner that is used in the cleardown procedure must also be stored in the operational database of the sign user (Table 14). This must also be loaded into the on-board computers within the framework of the vehicle data supply. When defining the DISID (AZBID) therefore, it is advisable to consider the physical data format that is used for the rapid cleardown function. Otherwise it would be necessary to implement additional operational data management - as described above.

Third-party agency code	DISID (AZBID)	Third-party sign reference
AVLC B	12345	XK3
AVLC B	45678	ZZ4

Table 14 - Data management of third-party sign references for rapid cleardown

6.3.6 Traction / trailers / coupled trips

The term traction describes the connection of individual trips to form a composite trip. These composite trips can be connected along the route or operated separately (coupled trips), but can also travel the entire journey together.

From the point of view of the passenger and depending on the given situation, these composite trips should be represented in the DPI as one combined trip or several individual trips. Furthermore, the position within the composite trip is relevant for some systems in order to send the necessary information of the individual trip to the destination displays according to the associated stopping section.

In order to allow this, a reference to an existing traction (TrainsetID (TraktionsID)) is included in the DPI messages. This allows the individual parts of a composite trip to be identified. Additional optional elements allow the total number and position of the individual trips to be determined (NumOfTrips, (AnzahlFahrten), Position).

The specification of traction data is optional. Departing from the standard behaviour of optional elements, the existence of traction data indicates that a trip at a stopping point is involved within a composite trip (traction).

The transmission of traction data begins inclusive of the stopping point at which the composite trip is formed and ends inclusive of the stopping point at which the composite trip separates or the trip leaves the composite. Trips can change composite trips and several composite trips can be combined into a single new one.

6.3.7 Reference data service (REF-DPI)

6.3.7.1 Data exchange

The exchange of planning data for the DPI is used to create reference data within the data consuming system (sign owner). Not all systems require reference data in order to be able to process the associated real-time (process) data. In these cases the exchange of reference data for the DPI system can be eliminated.

The REF-DPI reference data service is largely identical to the REF-CP reference data service. In both cases it involves an exchange of the departure times of vehicles at a third-party stop / DPI sign. The differences arise in the nomenclature (display area / connection area) as well as in the structure of the data that is exchanged.

6.3.7.1.1 Updates

As is the case for the REF-CP service, the data is updated for additional trips. Cancelled or modified trips are not communicated or reported as additional trips.

6.3.7.2 Polling DPI reference data (DISRefSubscription)

The polling of timetable data involves a SubscriptionRequest (AboAnfrage) with one or more embedded DISRefSubscription (AboAZBRef) elements. The DISRefSubscription (AboAZBRef) element specifies the display area, optional filter criteria as well as the polling time frame.

Definition of DISRefSubscription (AboAZBRef):

SubscriptionID (AboID):	(attribute) This references the subscription for planned DPI data. The SubscriptionID (AboID) is given by the sign owner system.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the subscription is valid.
DISID (AZBID):	Reference to the display area.
LineID (LinienID):	(optional) Filter for the third-party route whose data is to be supplied.
DirectionID (RichtungsID):	(optional) Filter for the third-party direction whose data is to be supplied.
EarliestDepartureTime (FruehesteAbfahrtszeit):	Defines the start of the time frame for which data is to be supplied. The reference is the departure time of the third-party vehicle from the connection area.
LatestDepartureTime (SpaetesteAbfahrtszeit):	Defines the end of the time frame for which data is to be supplied. The reference is the departure time of the third-party vehicle from the connection area.

LineID (LinienID) and DirectionID (RichtungsID) are independently optional. It is therefore possible to specify a request that only has one direction filter and no route filter.

The EarliestDepartureTime (FruehesteAbfahrtszeit) element should not be set so that it lies before the local data horizon and LatestDepartureTime (SpaetesteAbfahrtszeit) should not fall after the end of the horizon. The *ValidUntilTimeStamp (VerfallZst)* attribute should be the same time or later than the *LatestDepartureTime (SpaetesteAbfahrtszeit)* element.

The following example describes a reference data request (AVLC A is the sign owner) for display area '12345'. Only the data of route 10 in the direction of 'Zoo' is to be supplied. The sign owner AVLC, which has created the request, has a data horizon from 5:00am to 11:00pm, on 8.8.2001. The third-party AVLC only has one data horizon up to 10:00pm.

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T05:00:00">
      <AboAZBRef AboID="25" VerfallZst="2001-08-09T00:00:00">
      <AZBID>12345</AZBID>
      <LinienID>10</LinienID>
      <RichtungsID>Zoo</RichtungsID>
      <FruehesteAbfahrtszeit>
            2001-08-08T05:00:00
      </FruehesteAbfahrtszeit>
      <SpaetesteAbfahrtszeit>
            2001-08-08T23:00:00
```

```
      </SpaetesteAbfahrtszeit>
      </AboAZBRef>
</AboAnfrage>
```

The feeder responds with an Acknowledge (Bestaetigung) within a SubscriptionReply (Abo-Antwort) message. The restricted data horizon is signalled in DataValidUntil (DatenGueltigBis):

```
<AboAntwort>
      <Bestaetigung Zst="2001-08-08T05:00:05" Ergebnis="ok"
            Fehlernummer="0">
            <Datengueltigbis>2001-08-08T22:00:00</Datengueltigbis>
      </Bestaetigung>
</AboAntwort>
```

6.3.7.3 Transmitting DPI reference data (DISSchedule)

After the reference data subscriptions have been set up (6.3.7.2), the sign user system establishes the arrival boards in the display area of the local vehicles and signals this with the initial *DataReadyRequest (DatenBereitAnfragen)* message.

After receiving and confirming the 'data ready' message, the sign owner system polls the data with a DataSupplyRequest (DatenAbrufenAnfrage).

The sign user system replies to this message with the desired data within DISSchedule (AZBFahrplan) elements. In turn, the *DISSchedule (AZBFahrplan)* element is a sub-element of the so-called *DISMessage (AZBNachricht)* (*6.3.8.3*) and corresponds to a concrete arrival at a display area. A DISMessage (AZBNachricht) is directly assigned to a subscription.

The arrivals board is therefore derived from a list of several DISSchedule (AZBFahrplan) elements.

Definition of DISSchedule (AZBFahrplan):

TimeStamp (Zst):	(attribute) Time at which the timetable data changed.
DISID (AZBID):	Reference to the display area.
TripID (FahrtID):	(sub-element) References a third-party trip.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips. Value increases with each subsequent passage.
LineID (LinienID):	Route code of the third-party trip
LineText (LinienText):	Route name of the third-party trip (for the passenger)
TripNameText (FahrtBezeichnerText)	(optional, multiple, alphanumerical) Indication of the passenger-relevant trip name (*not to be confused with the TripName (FahrtBezeichner) in TripID (FahrtID)!*).
DirectionID (RichtungsID):	Direction code of the third-party trip

DirectionText (RichtungsText):	Description of the direction: Passenger-relevant destination text for the trip (vehicle signage on the next trip section as departure from the display area. This corresponds to the end of the trip or the next intermediate destination, e.g. for circular routes).
FromDirectionText (VonRichtungsText):	(optional) Passenger-relevant origin text for the trip. This corresponds to the origin of the trip or the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination, e.g. on circular routes). No specification: The source text from the annual timetable is used (per LineID and DirectionID).
ScheduledDISArrivalTime (AnkunftszeitAZBPlan):	(optional) Planned arrival time of the third-party trip in the display area.
StopID (HaltID):	(optional) References the stopping position within the connection area
ArrivalPlatformText (AnkunftssteigText):	(optional) Specification of the platform or stand descriptor for the arrival of the trip at the stop
DeparturePlatformText (AbfahrtssteigText):	(optional) Specification of the platform or stand descriptor for the departure of the trip from the stop
StopPositionText (HaltepositionsText):	(optional) Planned stopping position of the fetcher vehicle in the connection area.
ArrivalSectorText (AnkunftsSektorenText):	(optional) Information on the sectors for arrival at a platform/track
DepartureSectorText (AbfahrtsSektorenText):	(optional) Information on the sectors for departure at a platform/track
ScheduledDISDepartureTime (AbfahrtszeitAZBPlan):	(optional) Planned departure time of the third-party trip from the display area.
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the third-party trip.
NoBoarding (Einsteigeverbot):	(optional) Vehicle only stops for alighting. No specification: No boarding restrictions.
NoAlighting (Aussteigeverbot):	(optional) Vehicle only stops for boarding. No specification: No alighting restrictions.
PassThru (Durchfahrt):	(optional) Vehicle does not stop here. No specification: Vehicle does stop here.

In order to be able to represent starting or finishing trips correctly, the ScheduledDISArrivalTime (AnkunftszeitAZBPlan) and ScheduledDISDepartureTime (AbfahrtszeitAZBPlan) elements are specified as optional extras. It is necessary however to specify at least one element.

6.3.8 Process data service (DPI)

6.3.8.1 Data exchange

The exchange of process data for the DPI provides the sign owner system with information about third-party trips approaching the signs. Information is provided within a specific preview time window. The amount of trip information to be supplied can be restricted to the information that is to be represented on the signs.

The data exchange process is represented in Fig. 6-1.

The sign owner system polls data for specific routes and directions of the approaching trips (DISSubscription (AboAZB)). It can specify a so-called PreviewTime (Vorschauzeit), which specifies how long before the vehicles arrive at the sign, at which the system should start sending the prediction timetable data. Once this preview time is reached, the timetable statuses of the approaching vehicles are sent and then retransmitted after every change (DISDeviation (AZBFahrplanlage)). The detection of a change to the timetable status can be controlled by a Hysteresis (threshold) specification. Reaching the display area is also considered as a change (DataReadyRequest2 (DatenBereitAnfrage2)). The transmission of data for a subscription is started once the PreviewTime (Vorschauzeit) has been reached.

The subscription also defines the maximum number of vehicles that can be reported. The data volume can, for example, be restricted to only include the trips the sign is capable of displaying.

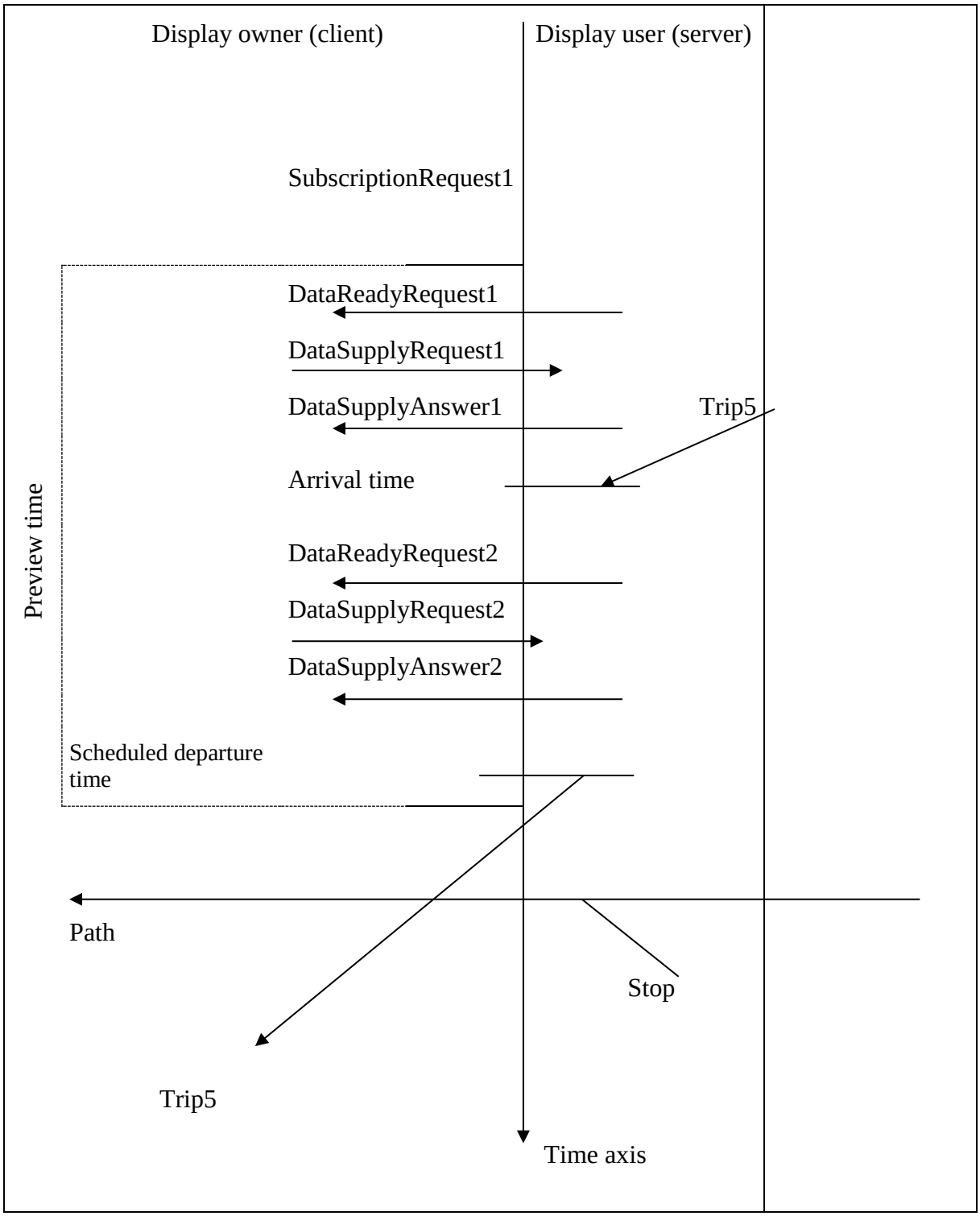

Fig. 6-1: Data exchange process for DPI

Data polling begins as soon as possible (at the start of the local operational day) and ends with it. As with the connection protection reference data service it must be assumed that it is not possible to cover the entire time period. For this reason, the Acknowledge (Bestaetigung) element again provides notification of the end of the time horizon of the system supplying the data.

6.3.8.2 Polling the DPI data (*DISSubscription*)

The retrieval of DPI data is initiated by the sign owner. It creates a SubscriptionRequest (Abo-Anfrage) which must contain one or more DISSubscription (AboAZB) sub-elements.

Definition of DISSubscription (AboAZB):	
SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription of DPI sign data. The SubscriptionID (AboID) is given by the sign owner system.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the subscription is valid.
DISID (AZBID):	Reference to the display area.
LineID (LinienID):	(optional) References the route of the sign user for which data is to be supplied.
DirectionID (RichtungsID):	(optional) References the direction of the sign user for which data is to be supplied.
PreviewTime (Vorschauzeit):	Specifies the lead time before the entry of an approaching trip at the DPI sign, from which it is possible to start with the transmission of the timetable status predictions (in minutes).
MaxNumOfTrips (MaxAnzahlFahrten):	(optional) Defines the maximum number of timetable statuses simultaneously reported in the DISSubscription (AboAZB) messages. The first n vehicles in accordance with the sequence of their arrival at the display area are supplied.
Hysteresis (Hysterese):	Desired change in seconds above which an update to the timetable deviation is communicated to the sign owner. The deviation must be greater than or equal to the specified value before deviations are transferred.
MaxTextLength (MaxTextLaenge):	(optional) Desired maximum text length of the destination and via texts. This value is only a recommendation. Longer texts can be supplied.
OnlyUpdate (NurAktualisierung):	(optional) When this element is set, the subscription informs the data provider that the ValidUntilTimeStamp (VerfallZst) is merely being extended. For this reason, it is not necessary for all trips corresponding to the subscription to be sent again in the next message, only the trips with a new trigger (e.g. hysteresis, entry into preview time need to be sent). If the same subscription as the previous one is not available on the data provider side, or if this element is missing in the SubscriptionRequest (AboAnfrage), or if the client did not implement the element, all data corresponding to the subscription must sent in the initialising message after the SubscriptionRequest (AboAnfrage) (or a portion of the data if PendingData (WeitereDaten) = true is used).

LineID (LinienID) and DirectionID (RichtungsID) are independently optional. It is therefore also possible to create a request without any filters. MaximumStopVisits (MaxAnzahlFahrten) is used to optimise the data traffic. The sign owner can use it to restrict the number of trips sup-

plied in a DISDeviation (AZBFahrplanlage) message. This means that the number of messages can be tailored to match the number of messages that can be shown on the signs. The trips currently approaching the display area are supplied. As with time-based connection protection, once a timetable status has been sent once for a particular trip, it must continue to be transmitted. Any additional dispatch actions therefore can result in a situation in which there are more timetable predictions than that defined in the MaxNumOfTrips (MaxAnzahlFahrten) message. The following tables highlight this procedure:

Table 15 shows a departure board (of the sign user) with the trips that are approaching the display area. In this case, MaxNumOfTrips (MaxAnzahlFahrten) has been defined with the value 3 in the subscription. This means that once the preview time has been reached, only the trips in grey are reported.

TripID (FahrtID)	Predicted departure time
123	13:00
124	13:10
125	13:20
126	13:30
127	13:40
128	13:50

Table 15 - Departure board limited by MaxNumOfTrips

If reinforcement trip 566 with a predicted departure time of 1:05pm is now dispatched, this 'pushes' trip 125 from third place. Despite this, the details relating to trip 125 continue to be sent (see Table 16), as it has already been communicated once before.

There can now be up to four DISDeviation (AZBFahrplanlagen) elements within one DISMessage (AZBNachricht).

TripID (FahrtID)	Predicted departure time
123	13:00
566	13:05
124	13:10
125	13:20
126	13:30
127	13:40
128	13:50

Table 16 - Exceptions to the constraints of reinforcement trips

Implementation notes:

If there are fixed areas on the sign for specific routes / directions or if the sign is to show the next trip for each specified route / direction, the subscriptions should also be set up on a route / direction specific basis.

If the specified hysteresis value (threshold) has only been reached or exceeded by some of the transmitted trips, only their data needs to be transmitted.

In the following example, data is polled for a sign (owned by AVLC A) with DISID (AZBID) '12345'. Data should be transmitted for the time period up to 11:00pm with a PreviewTime of 30 minutes. The text fields should contain no more than 30 characters (MaxTextLength (Max-TextLaenge)). The sign user should report a maximum of 3 trips of route 8 in all directions:

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T05:00:00">
      <AboAZB AboID="25" VerfallZst="2001-08-08T23:00:00">
            <AZBID>12345</AZBID>
            <LinienID>8</LinienID>
            <Vorschauzeit>30</Vorschauzeit>
            <MaxAnzahlFahrten>3<MaxAnzahlFahrten>
            <Hysterese>120</Hysterese>
            <MaxTextLaenge>30</MaxTextLaenge>
      </AboAZB>
</AboAnfrage>
```

The sign user confirms acceptance of the subscription and reports back that it can only supply data up to 10:00pm:

```
<AboAntwort>
      <Bestaetigung Zst="2001-08-08T05:00:05" Ergebnis="ok"
            Fehlernummer="0">
            <Datengueltigbis>2001-08-08T22:00:00</Datengueltigbis>
      </Bestaetigung>
</AboAntwort>
```

6.3.8.3 Messages from the sign users (DISMessage)

If there are updates available, the sign user responds with a so-called DISMessage (AZBNachricht) once the preview time has been reached. With reference to a subscription, this element forms the framework for the results of dispatch actions or timetable statuses.

The following messages can be reported within the DISMessage (AZBNachricht):

Reference data service:

- Timetable data (DISSchedule (AZBFahrplan))

Process data service:

- Timetable status predictions (DISDeviation (AZBFahrplanlage))
- Cancellation of a trip, departure from the sign (DISTripDelete (AZBFahrtLoeschen))
- Creation and deletion of a special route text (DISLineSpecialText / DISLineSpecialTextDelete)

Definition of DISMessage (AZBNachricht):

SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription of DPI data. The SubscriptionID (AboID) is given by the displaying system
DISSchedule (AZBFahrplan):	(sub-element, optional, multiple) Timetable information relating to a trip.
DISDeviation (AZBFahrplanlage):	(sub-element, optional, multiple) Prediction information relating to a trip.
DISTripDelete (AZBFahrtLoeschen):	(sub-element, optional, multiple) Instructs the recipients to delete the trip from the sign.
DISLineSpecialText (AZBLinienSpezialtext):	(sub-element, optional, multiple) Transmission of free text for a trip or direction.
DISLineSpecialTextDelete (AZBLinienSpezialtextLoeschen):	(sub-element, optional, multiple) Deletion of the text information

As many sub-elements as necessary can be specified for every DISMessage (AZBNachricht).

6.3.8.3.1 Transmitting prediction data (DISDeviation)

Once the preview time has been reached, the system starts to transmit the prediction data for the relevant trips.

In the above example, this is 30 minutes before the vehicle (route 8) reaches the display area. An initial timetable prediction is created by the sign user and signalled ready for retrieval (DataReadyRequest (DatenBereitAnfrage)). The sign owner confirms this (DataReadyReply (DatenBereitAntwort) / Acknowledge (Bestaetigung)) and polls the subscription data (DataSupplyRequest (DatenAbrufenAnfrage)). The sign user responds with a DISMessage (AZBNachricht) associated with the subscription.

This groups the timetable predictions (DISDeviation (AZBFahrplanlage)) of all approaching trips associated with the subscription.

Definition of DISDeviation (AZBFahrplanlage):

TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the data.
ValidUntilTimeStamp (VerfallZst):	(attribute) Time to which the data in this message is valid.
DISID (AZBID):	Identifies the display area
TripID (FahrtID):	(sub-element) Uniquely identifies the trip to be displayed.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips. Value increases with each subsequent passage.
Trainset (Traktion):	(sub-element, optional) Contains data relating to the composite trip in which the current trip is located.
VehicleNumber (BetrieblicheFahrzeugnummer):	*(optional, multiple) References vehicles that are linked together within a train set or composite unit.*
LineID (LinienID):	Reference to the route associated with the trip
LineText (LinienText):	Route name of the third-party trip (relevant for the passenger)
TripNameText (FahrtBezeichnerText):	(optional, multiple, alphanumerical) Indication of the passenger-relevant trip name (*not to be confused with TripName (FahrtBezeichner) in TripID (FahrtID)!*).
DirectionID (RichtungsID):	Reference to the direction of the trip
DirectionText (RichtungsText):	Description of the direction: Passenger-relevant destination text of the trip (vehicle signage on the next trip section at departure from the display area. This corresponds to the end of the trip or the next intermediate destination, e.g. for circular routes).
FromDirectionText (VonRichtungsText):	(optional) Passenger-relevant origin text for the trip. This corresponds to the origin of the trip or the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination, e.g. on circular routes). No information: The source text from the annual timetable is used (per LineID (LinienID) and DirectionID (RichtungsID)).

DepartureNoticeID (AbmeldeID):	(optional) Internal trip number for the rapid cleardown function.
DestinationStop (ZielHst):	Code name of the destination stop.
AtDISPoint (AufAZB):	(optional) Specifies that the vehicle is at the display area stop.
ViaStop1 (ViaHst1Lang):	(optional) Full name of the first 'via' stop. (Element no longer in use, please replace with Via.)
ViaStop2 (ViaHst2Lang):	(optional) Full name of the second 'via' stop. (Element no longer in use, please replace with Via.)
ViaStop3 (ViaHst3Lang):	(optional) Full name of the third 'via' stop. (Element no longer in use, please replace with Via.)
Via (Via):	(sub-element, optional, multiple) New Via structure, additionally includes priorities.
TripStatus (FahrtStatus):	Specifies whether actual (real-time) information can be supplied for the vehicle (“actual”) or not (“planned”).
ScheduledDISArrivalTime (AnkunftszeitAZBPlan):	(optional) Planned arrival time at the display area stop.
ExpectedDISArrivalTime (AnkunftszeitAZBPrognose):	(optional) Predicted arrival time at the display area stop if TripStatus = “actual”. Otherwise the scheduled time.
AimedDepartureTime (AbfahrtszeitAZBPlan):	(optional) Planned departure time from the display area stop.
ExpectedDISDepartureTime (AbfahrtszeitAZBPrognose):	(optional) Predicted departure time from the display area stop if TripStatus (FahrtStatus) = 'actual'. Otherwise the scheduled time.
AimedDISDepartureTime (AbfahrtszeitAZBDisposition):	(optional) Foreseen departure time (based on current dispatch status) from the display area stop.
StopVisitNote (Fahrtspezialtext):	(optional) Text that is to be shown on the sign in place of the current trip data. A trip-specific text must be deleted if this element is not included in the transmission.
SpeechOutput (Sprachausgabe):	(optional, multiple) Indicates whether the trip-specific text is also to be announced via the voice unit and in which language.
StopID (HaltID)	(optional) References the stopping position within the connection area.
ArrivalPlatformText (AnkunftssteigText)	(optional) Specification of the platform or stand descriptor for the arrival of the trip at the stop
DeparturePlatformText (AbfahrtssteigText)	(optional) Specification of the platform or stand descriptor for the departure of the trip from the stop
StopPositionText (HaltepositionsText):	(optional) Text describing the arrival/departure location of the trip in the display area.

QueueIndicator (Stauindikator):	(optional) States whether the vehicle is stuck in traffic (true) or not (false).
ArrivalSectorText (AnkunftsSektoren-Text):	(optional) Information on the sectors for arrival at a platform/track
DepartureSectorText (AbfahrtsSektoren-Text):	(optional) Information on the sectors for departure at a platform/track
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the current trip.
NoBoarding (Ein-steigeverbot):	(optional) Vehicle only stops for alighting. No specification: No boarding restrictions.
NoAlighting (Aus-steigeverbot):	(optional) Vehicle only stops for boarding. No specification: No alighting restrictions.
PassThru (Durch-fahrt):	(optional) Vehicle does not stop here. No specification: Vehicle does stop here.

Definition of Via (Via):

StopName (Haltestel-lenName):	Stop name
StopPriority (Hal-testellenPriorität):	(optional) The smaller the number the higher the priority.

Implementation notes:

Once the ValidUntilTimeStamp (VerfallZst) has been reached, the respective trip must be deleted from the sign, even if this time precedes the ExpectedDISDepartureTime (AbfahrtszeitAZBPrognose) or the ExpectedDISArrivalTime (AnkunftszeitAZBPrognose).

In order to be able to represent starting or finishing trips correctly, the ScheduledDISArrivalTime (AnkunftszeitAZBPlan) and ScheduledDISDepartureTime (AbfahrtszeitAZBPlan) or ExpectedDISArrivalTime (AnkunftszeitAZBPrognose) and ExpectedDISDepartureTime (AbfahrtszeitAZBPrognose) elements are specified as optional extras. It is necessary however to specify at least one element.

All information relating to the productive trips must be made available to the third-party control centres for the representation of the arrivals and departures:

- For the intermediate stops, DISDeviation (AZBFahrplanlage) contains both the arrival and the departure times.
- Finishing trips or trips with unproductive departures do not contain any departure times. Arrival times are transmitted for these trips, as long as the arrival is productive and its direction matches the subscription specifications.
- Starting trips or trips with unproductive arrivals do not contain any arrival times. Departure times are transmitted for these trips, as long as the departure is productive and its direction matches the subscription specifications.

In a similar way to the connection protection, the AimedDISDepartureTime (AbfahrtszeitAZBDisposition) element provides information on holding back the fetcher to the DPI service. As shown in the following examples, AimedDISDepartureTime (AbfahrtszeitAZBDisposition) ensures this information is separated from the ExpectedDISDepartureTime (AbfahrtszeitAZBPrognose) information.

If this hold back were to be incorporated into the ExpectedDISDepartureTime (AbfahrtszeitAZBPrognose), as shown in the following examples, a suitably adjusted departure time prediction would be shown on the stationary signs, which could lead the passengers to infer a guaranteed later departure giving them time to maybe leave the stop. If the connection is cancelled in the meantime however, by means of a dispatch action, the fetcher departs immediately and the passengers would miss the train.

Example 1: Only internal connections for trains, no reverse channel to the bus operators possible

It can be seen in the example that when the bus arrives, the stationary sign shows

- 'board now', if the train is already standing at the stop

or

- a countdown showing time to departure (without any hold-back delay), if the train has not yet arrived at the stop.

The resulting inconsistency between the information in the (delayed) vehicle and that on the sign is intentional:

The passengers in the bus are informed that despite their expectations they will probably still catch the train (this is an essential element of passenger information from the point of view of the passenger). The fixed signs on the platforms however inform the waiting passengers there to board immediately, in order to prevent anyone missing the train should the connection be cancelled in the meantime.

The relaying of false information to the passenger in the bus in the case where the connection has failed (a rare but possible situation) must be accepted on the grounds of freedom to initiate suitable dispatch actions.

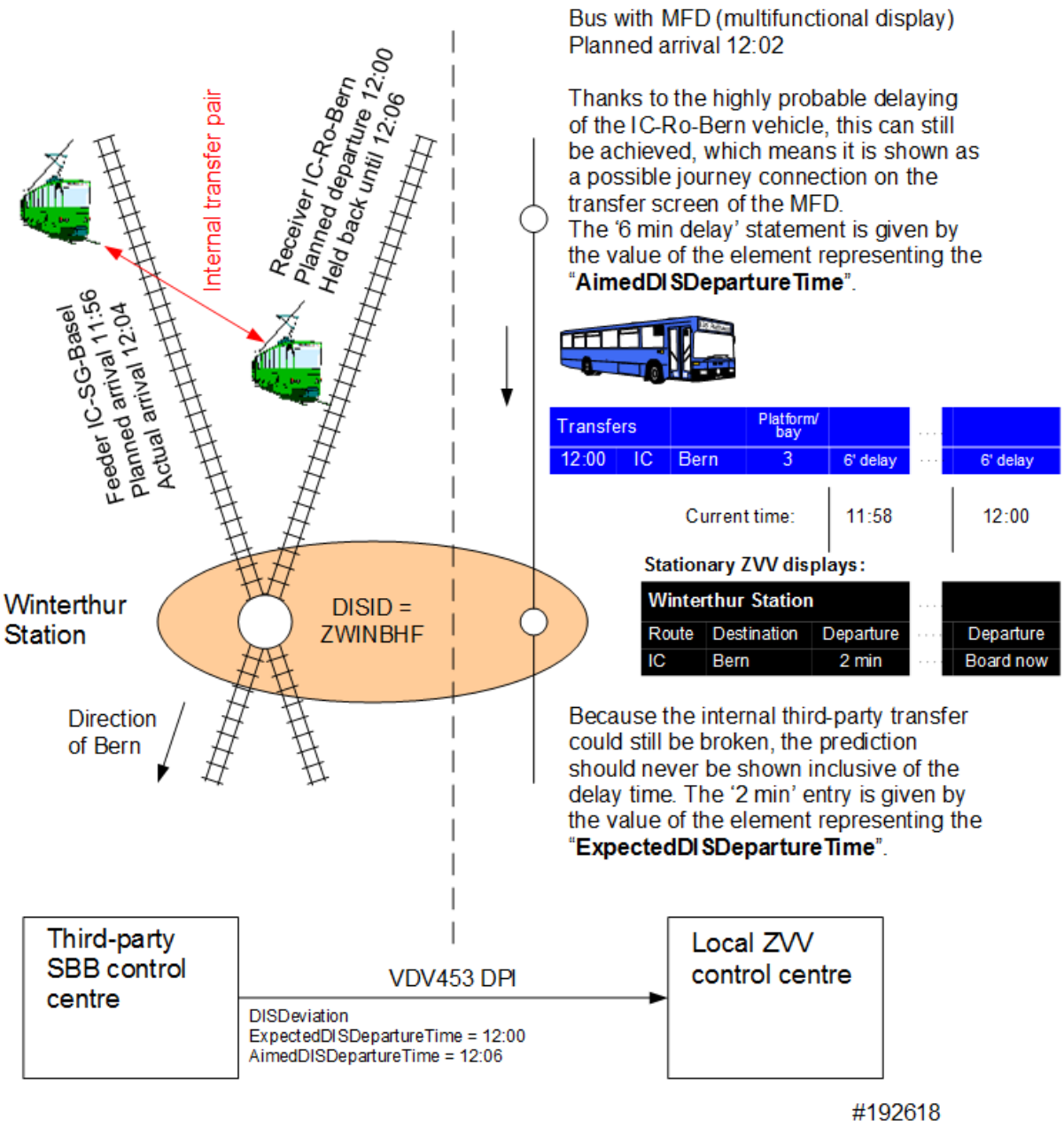

Fig. 6-2: Example 1: Internal connections for the fetchers

Example 2: Connections at the preceding stop only, no reverse channel possible

Example 2 shows the significant advantage of the additional transmission of the hold-back (delay) time. From 11:58am the bus indicates that route 14 will depart at 12:11pm The count-down on the sign remains at 5 minutes between 12:00pm and 12:06pm, as route 14 has been held back at the previous stop during this time in order to wait for the delayed route 2. If the bus on route 5 arrives at the stop at 12:09pm, then 2 minutes is shown correctly on the sign.

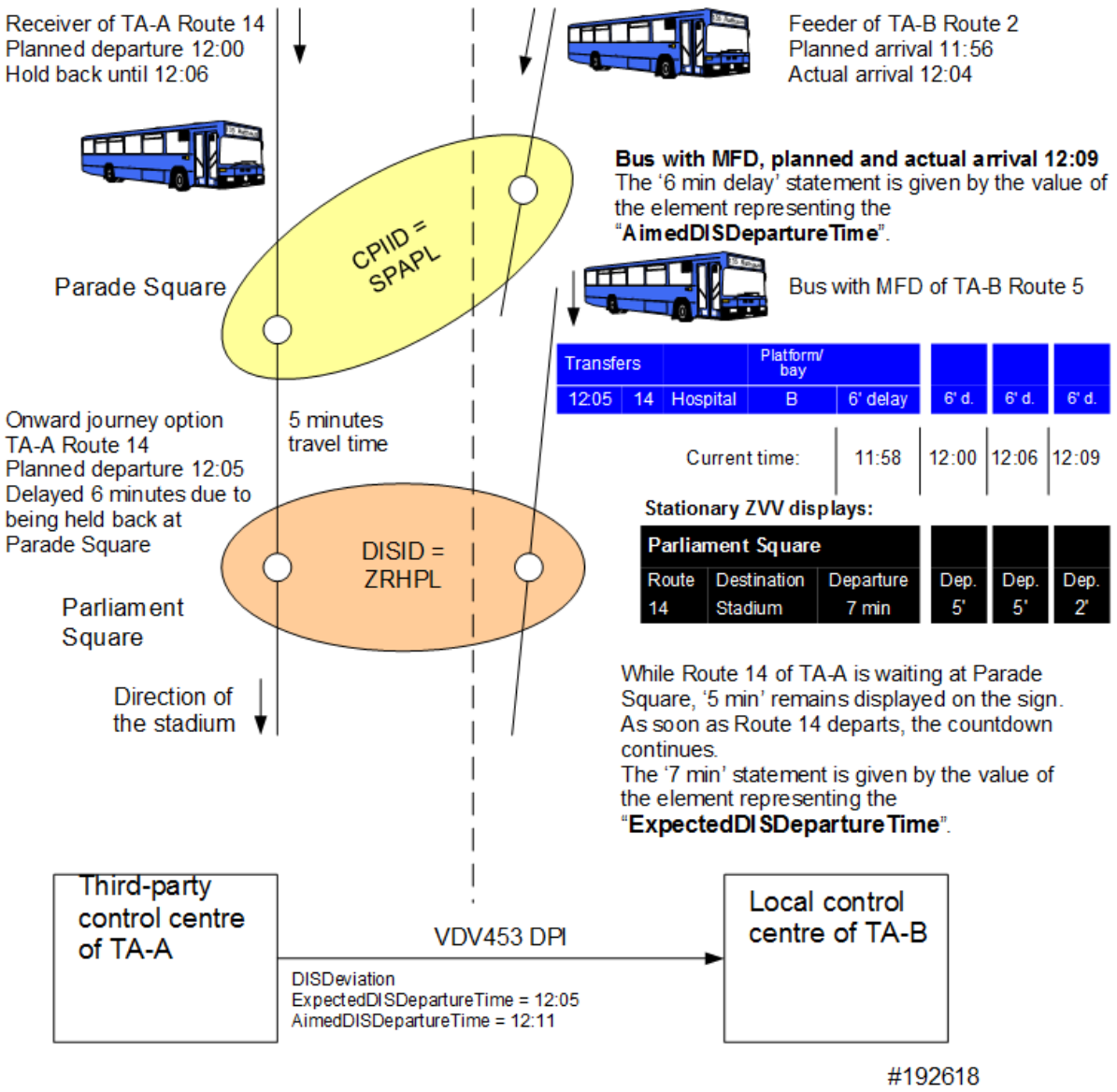

Fig. 6-3: Example 2: Hold back at the previous stop

The above examples show that even though it is possible to transmit additional, passenger relevant information, it can prove problematic in certain situations and even be misleading. It remains the responsibility of the fetcher / client to use and pass on this information but may need to be agreed in advance in individual cases.

6.3.8.3.2 Composite units (traction)

Definition of Trainset (Traktion):	
TrainsetID (Traktion-sID):	Unique reference to the composite unit (traction).

NumOfTrips (AnzahlFahrten):	(optional) Number of individual trips in the composite unit. If not specified there is no information concerning the number of trips in the composite unit.
Position (Position):	(optional) Position of the trip within a composite trip (1 = first position in the direction of travel). If not specified, it is not possible to give any positional information within the traction unit.

In the following example, the sign user replies to the above subscription (25) with the timetable deviation of exactly one vehicle (2367). The trip has a one-minute delay and is positioned in first place within the composite unit, with a further two trips behind.

If several trips fall within the preview time of a subscription, they must all be reported in separate DISDeviation (AZBFahrplanlage) elements.

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T07:30:15" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <AZBNachricht AboID="25">
            <AZBFahrplanlage Zst="2001-08-08T07:30:00
                              VerfallZst="2001-08-08T08:06:00">
                  <FahrtID>2367</FahrtID>
                  <HstSeqZaehler>1</HstSeqZaehler>
                  <AZBID>12345</AZBID>
                  <Traktion>
                        <TraktionsID>234234</TraktionsID>
                        <AnzahlFahrten>3</AnzahlFahrten>
                        <Position>1</Position>
                  </Traktion>
                  <LinienID>8</LinienID>
                  <LinienText>8</LinienText>
                  <RichtungsID>HBF</RichtungsID>
                  <RichtungsText>Hauptbahnhof</RichtungsText>
                  <AbmeldeID>3426</ AbmeldeID >
                  <AufAZB>false</AufAZB>
                  <ZielHst>Hauptbahnhof</ZielHst>
                  <ViaHst1Lang>Marktplatz</ViaHst1Lang>
                  <ViaHst2Lang>Zoo</ViaHst2Lang>
                  <ViaHst3Lang></ZielHst3Lang>
                  <AnkunftszeitAZBPlan>
                        2001-08-08T08:00:00
                  </AnkunftszeitAZBPlan>
                  <AnkunftszeitAZBPrognose>
                        2001-08-08T08:01:00
                  </AnkunftszeitAZBPrognose>
                  <AbfahrtszeitAZBPlan>
                        2001-08-08T08:01:00
                  </AbfahrtszeitAZBPlan>
                  <AbfahrtszeitAZBPrognose>
                        2001-08-08T08:02:00
                  </AbfahrtszeitAZBPrognose>
            <FahrtStatus>Ist</FahrtStatus>
                  <HaltepositionsText>Steig 3</HaltepositionsText>
                  <FahrtInfo>
                   ...
                  </FahrtInfo>
```

```
            </AZBFahrplanlage>
            ... weitere zwei Fahrplanlagen
      </AZBNachricht>
</DatenAbrufenAntwort>
```

This process is repeated, as soon as there is any change in the subscription.

No more updates are sent once the display area has been reached. The subscription must be specifically deleted.

6.3.8.3.3 Transmitting special route texts (DISLineSpecialText)

The special route text is used to inform the passengers of special events relating to a specific route (e.g. 'traffic jam following an incident')

Displayed special route texts must be explicitly deleted again from the signs by the sign user, or they are removed after the subscription is deleted.

The sign user reports special route texts with the DISLineSpecialText (AZBLinienSpezialtext) element within the DISMessage (AZBNachricht).

Definition of DISLineSpecialText (AZBLinienSpezialtext):

TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the data.
ValidUntilTimeStamp (VerfallZst):	(attribute) Time to which the data in this message is valid.
DISID (AZBID):	Identifies the display area
LineID (LinienID):	References the route of the sign user for which a special text is to be displayed.
LineText (Linien-Text):	(optional) Name of the route for which a special route text is to be displayed.
DirectionID Richtungs-ID):	References the direction for displaying the special text.
Special route text:	Text that is to be displayed.
Priority (Prioritaet):	(optional) Classifies the relevance of the information (in stages from 1 to 3 where 1 is the highest and 3 is the lowest priority). The representation method can be derived from this information (permanent display, alternating display, scrolling text etc.).
SpeechOutput (Sprachausgabe):	(optional, multiple) Indicates whether the trip-specific text is also to be announced via the voice unit and in which language.

In the following example and according to the above subscription, a text is to be displayed for route 8, direction Central Station which provides information on the suitability for disabled passengers:

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T07:30:15" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <AZBNachricht AboID="25">
```

```
            <AZBLinienSpezialtext   Zst="2001-08-08T07:30:00
                                    VerfallZst="2001-08-08T09:00:00">
                  <AZBID>12345</AZBID>
                  <LinienID>8</LinienID>
                  <RichtungsID>HBF</RichtungsID>
                  </LinienSpezialtext>
                        All vehicles on Route 8 travelling towards Central
Station are low-floor buses offering easy access to disabled passengers
                  </LinienSpezialtext>
            </AZBLinienSpezialtext>
      </AZBNachricht>
</DatenAbrufenAntwort>
```

It is possible to send several special route texts within one message (in accordance with the definition of DISMessage (AZBNachricht)). It is necessary to ensure that the special route texts do not overlap.

6.3.8.3.4 Delete the line special texts *(DISLineSpecialTextDelete)*

The special route texts are deleted in a similar way to that in which they are created. The actual LineSpecialText (Linienspezialtext) element is missing:

Definition of DISLineSpecialTextDelete (AZBLinienSpezialtextLoeschen):	
TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the data.
DISID (AZBID):	Identifies the display area
LineID (LinienID):	References the route of the sign user for which a special text is to be deleted.
DirectionID (RichtungsID):	References the direction for which a special text is to be deleted.

The following example deletes the previously created special text:

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T07:30:15" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <AZBNachricht AboID="25">
            <AZBLinienSpezialtextLoeschen Zst="2001-08-08T07:30:00>
                  <AZBID>12345</AZBID>
                  <LinienID>8</LinienID>
                  <RichtungsID>HBF</RichtungsID>
            </AZBLinienSpezialtextLoeschen>
      </AZBNachricht>
</DatenAbrufenAntwort>
```

6.3.8.3.5 Trip cancellation / departure (DISTripDelete)

If a sign user trip is cancelled or has departed from the display area, the sign user system must send a corresponding deletion message. To achieve this, the sign user sends one or more

DISTripDelete (AZBFahrtLoeschen) elements within a DISMessage (AZBNachricht). The sign user must delete the trip from the sign or show a corresponding message for the passengers.

Definition of DISTripDelete (AZBFahrtLoeschen):

TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the data.
DISID (AZBID):	Reference to the display area.
TripID (FahrtID):	(sub-element) References the departed/cancelled trip.
StopSeqCount (HstSeqZaehler):	Passage counter for branch trips. Value increases with each subsequent passage.
LineID (LinienID):	Reference to the route associated with the cancelled trip.
LineText (LinienText):	Route name of the third-party trip (relevant for the passenger)
DirectionID (RichtungsID):	Reference to the direction of the cancelled trip.
DirectionText (RichtungsText):	Direction of the third-party trip (relevant for the passenger)
FromDirectionText (VonRichtungsText):	(optional) Passenger-relevant origin text for the trip.
ScheduledDISArrivalTime (AnkunftszeitAZBPlan):	(optional) Planned arrival time at the display area stop.
ScheduledDISDepartureTime (AbfahrtszeitAZBPlan):	(optional) Planned departure time from the display area stop.
StopID (HaltID):	(optional) References the stopping position within the connection area
StopPositionText (HaltepositionsText):	(optional) Text describing the arrival/departure location of the trip in the display area.
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the current trip.
DepartureNoticeID (AbmeldeID):	(optional) Used for the rapid cleardown function.
Reason (Ursache)	(optional) Description of the cause of the error Absent under normal conditions (departure from display area).

Because it is not always possible to uniquely reference the trip using *TripID (FahrtID)*, the optional fields *FromDirectionText (VonRichtungsText), ScheduledDISArrivalTime (AnkunftszeitAZBPlan), ScheduledDISDepartureTime (AbfahrtszeitAZBPlan), StopID (HaltID), StoppingPositionText (HaltepositionsText)* and *TripInfo (FahrtInfo)* shall be filled in if possible.

The following example indicates the cancellation of a trip (6612) at 3:55pm in display area '12345'.

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T15:56:00" Ergebnis="ok"
            Fehlernummer="0">
```

```
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <AZBNachricht AboID="25">
            <AZBFahrtLoeschen Zst="2001-08-08T15:55:00">
                  <FahrtID>
                        <FahrtBezeichner>6612</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <HstSeqZaehler>1</HstSeqZaehler>
                  <AZBID>12345</AZBID>
                  <LinienID>8</LinienID>
                  <LinienText>8</LinienText>
                  <RichtungsID>HBF</RichtungsID>
                  <RichtungsText>Hauptbahnhof</RichtungsText>
                  <Ursache>Motorschaden</Ursache>
            </AZBFahrtLoeschen>
      </AZBNachricht>
</DatenAbrufenAntwort>
```

Following the cancellation message, no further updates are reported for this trip.

6.4 Visualising the third-party vehicles (VIS)

6.4.1 Introduction

The visualisation of third-party vehicles service (VIS) is used to transmit trip information to signs in a third-party AVLC system. The following information is then available in the third-party AVLC system:

- Trip information (ID, internal number, service features)
- Geographical position (longitude, latitude), optional
- Current position on the pattern, optional

The following tables and diagrams are therefore possible:

- Vehicle list (table)
- GIS map diagram
- Route ladder

The exact tables and diagrams that can be implemented depend on the type of data that is provided by the data supplying system. As geographical position and current pattern section are optional, they simultaneously represent prerequisites for the corresponding representations.

The route ladder also demands the availability of pattern models for the vehicles to be displayed. The exchange of this data is not included in this interface.

The visualisation service only transmits process data.

6.4.2 Operational data supply and management

Visualisation data must be polled. The subscription is achieved via so-called visualisation areas, which specify the amount of data to be transmitted.

In a similar way to the other location codes (6.1.4), the visualisation areas *(VISIDs*) must be agreed between the participating agencies and incorporated into the local data management.

The term 'visualisation area' should not be taken exclusively as a geographically defined area, but as a synonym for a section of all available data (see 3.3.2).

Visualisation areas can depict successive individual routes, where each route is assigned a separate VISID (VISID).

A visualisation area however can also comprise several routes. In this situation, unique assignment is then guaranteed by means of further stipulations.

It is also possible, of course, to define a visualisation area in simple geographical terms, e.g. a selected number of stops for which all relevant routes must supply data.

6.4.3 Process data service

6.4.3.1 Procedure

The exchange of visualisation data starts with the creation of a subscription by the displaying system. It creates subscriptions with reference to the desired visualisation areas, for which data is to be supplied.

Once a subscription has been created, trip information is sent from the data producers to the visualising system for the entire time it remains valid. The type and frequency of repetition is left to the data producer, but can be specified by the displaying system within the context of a recommendation. The information that is sent relates exclusively to vehicles that are currently en route.

Only complete records are communicated. Information on individual trips may be missing from consecutive messages if there have been no changes to position, delay status or any other data.

Once a vehicle has left a visualisation area, the data producer sends a corresponding message that allows the displaying system to remove the vehicle in question. The act of departing from a visualisation area depends on the given definitions, but in operational terms can be reduced to the following points:

- Trip has ended.
- Trip has been cancelled.
- Trip has left the area zone (spatial definition)

In general, the transmitted information should provide as complete and accurate a representation of the given operation as possible.

6.4.3.2 Poll visualisation data (VISSubscription)

The polling of visualisation data starts with the transmission of a subscription request by the displaying system. This SubscriptionRequest (AboAnfrage) contains one or more VISSubscription (AboVIS) elements. The VISSubscription (AboVIS) contains the visualisation area, the requested time period as well as the desired update cycle (recommendation).

Definition of VISSubscription (AboVIS):

SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription of visualisation data. The SubscriptionID (AboID) is given by the displaying system
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the subscription is valid.
VISID (VISID):	Reference to the visualisation area.
LineID (LinienID):	(optional) Restricts the data volume to the specified route.
DirectionID (RichtungsID):	(optional) Restricts the data volume to the specified direction.
Cycle (Zyklus):	Time interval in seconds, after which new data is sent.

The following example indicates the creation of a visualisation subscription for visualisation area 12345 in the time period between 5:00am and 10:00pm. New data is required every 120 seconds.

```
<AboAnfrage spaetesteAbfarhtszeit="AVLC A" Zst="2001-08-08T05:00:00">
      <AboVIS AboID="25" VerfallZst="2001-08-08T22:00:00">
      <VISID>12345</VISID>
      <Zyklus>120</Zyklus>
      </AboVis>
</AboAnfrage>
```

The data producer confirms reception and creation of the subscription with an Acknowledge (Bestaetigung) element within the SubscriptionReply (AboAntwort).

6.4.3.3 Messages from the visualisation service (*VISMessage)*

All messages of the data producing system are encapsulated within the so-called VISMessage (VISNachricht). The VISMessage (VISNachricht) is the reference frame for the subscription. The VISMessage (VISNachricht) can contain the following sub-elements:

- Trip information (VISDeviation (VISFahrplanlage))
- Trip leaves the visualisation area (VISTripDelete (VISFahrtLoeschen))

Definition of VISMessage (VISNachricht):	
SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription of visualisation data. The SubscriptionID (AboID) is given by the displaying system
VISDeviation (VIS-Fahrplanlage):	(sub-element, optional, multiple) Information relating to a trip.
VISTripDelete (VIS-FahrtLoeschen):	(sub-element, optional, multiple) Instructs the recipients to delete the trip from the displays.

6.4.3.3.1 Sending visualisation data (VISDeviation)

When new visualisation data becomes available, this is signalled by the data producer with a DataReadyRequest (DatenBereitAnfrage) message. The displaying system acknowledges this with a DataReadyReply (DatenBereitAntwort).

It then requests the data with the transmission of a DataSupplyRequest (DatenAbrufenAnfrage):

```
<DatenAbrufenAnfrage Sender="AVLC B" Zst="2001-08-08T05:05:00">
      <DatensatzAlle>true</DatensatzAlle>
</DatenAbrufenAnfrage>
```

The data producer provides the trip information in its response.

Definition of VISDeviation (VISFahrplanlage):	
TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the data.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time at which the trip is to be removed from the visualisation in the absence of any further data.
VISID (VISID):	Unique reference to the visualisation area.
TripID (FahrtID):	(sub-element) References a trip to be visualised.
LineID (LinienID):	Route of the trip.
LineText (Linien-Text):	Description (name) of the route (relevant to the passenger)
DirectionID (Rich-tungsID):	Direction of the trip.
DirectionText (Rich-tungsText):	Description of the direction: Passenger-relevant destination text of the trip (vehicle signage on the next trip section at departure from the display area. This corresponds to the end of the trip or the next inter-mediate destination, e.g. for circular routes).

FromDirectionText (VonRichtungsText):	(optional) Passenger-relevant origin text for the trip. This corresponds to the origin of the trip or the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination, e.g. on circular routes). No specification: The source text from the annual timetable is used (per LineID and DirectionID).
TripStatus (FahrtStatus):	Specifies whether actual (real-time) information can be supplied for the vehicle (“actual”) or not (“planned”). If it is only possible to supply planning data, positions and delays are considered outdated.
Delay (Verspaetung):	(optional) Current delay in seconds. Early times are indicated by negative values.
DepartureStop (StartHst):	(optional) Short description of the start stop.
EndStop (EndHst):	(optional) Short description of the last stop.
ActStop (AktHst):	(optional) Short description of the current or last stop.
StopSeqCount (HstSeqZaehler):	(optional) Related to ActStop (AktHst) and specifies the passage sequence in branch trips.
AtStop (AufHst):	(optional) Flag that indicates whether the vehicle is currently located at the ActStop ('true').
NextStop (NachHst):	(optional) Short description of the stop the vehicle will next arrive at.
Distance (Distanz):	(optional) Travelled distance as a percentage of the section between ActStop (AktHst) and NextStop (NachHst).
Velocity (Geschwindigkeit):	(optional) Last known (average) speed of the vehicle (usually trains) at a specific measurement time point (TimeStamp (Zst)). This definition must be agreed bilaterally: e.g. actual or average speed.
Bearing (KompassRichtung):	(optional) This is the compass reading defined as an integer between 0-359 degrees
Longitude (Longitude):	(optional) Geographical longitude in WGS-84 format (milliseconds).
Latitude (Latitude):	(Optional) Geographical latitude in WGS-84 format (milliseconds).
QueueIndicator (Stauindikator):	(optional) States whether the vehicle is stuck in traffic (true) or not (false).
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the trip.

The following example shows a possible reply to the above subscription. Exactly one vehicle is reported.

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T05:05:10"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <VISNachricht AboID="25">
            <VISFahrplanlage Zst="2001-08-08T05:05:05"
```

```
                        VerfallZst="2001-08-08T06:05:05">
                  <VISID>12345</VISID>
                  <FahrtID>
                        <FahrtBezeichner>64356</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <HstSeqZaehler>1</HstSeqZaehler>
                  <LinienID>10</LinienID>
                  <LinienText>X10</LinienText>
                  <RichtungsID>Zoo</RichtungsID>
                  <RichtungsText>Zoologischer Garten</RichtungsText>
                  <FahrtStatus>Ist</FahrtStatus>
                  <AktHst>PARISSTR</AktHst>
                  <NachHst>NBGRSTR</NachHst>
                  <Distanz>50</Distanz>
                  <Longitude>46800123</Longitude>
                  <Latitude>93355177</Latitude>
                  <Stauindikator>false</Stauindikator>
            </VISFahrplanlage>
      </VISNachricht>
</DatenAbrufenAntwort>
```

6.4.3.3.2 Deleting visualisation data (VISTripDelete)

If a trip leaves a visualisation area (usually once the trip has ended) the data producer must send a deletion message which prompts the displaying system to remove the trip from the corresponding displays.

To achieve this it sends a VISMessage (VISNachricht) containing the VISTripDelete (VISFahrtLoeschen) element within a DataSupplyAnswer (DatenAbrufenAntwort). Afterwards it does not send any more VISDeviations (VISFahrplanlage).

Definition of VISTripDelete (VISFahrtLoeschen):

TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the data.
VISID (VISID):	Unique reference to the visualisation area.
TripID (FahrtID):	(sub-element) References the trip to be deleted.
LineID (LinienID):	Route of the trip.
LineText (LinienText):	Description (name) of the route (relevant to the passenger)
DirectionID (RichtungsID):	Direction of the trip.
DirectionText (RichtungsText):	Description of the direction (relevant to the passenger)
FromDirectionText (VonRichtungsText):	(optional) Passenger-relevant origin text for the trip. This corresponds to the origin of the trip or the last intermediate destination (corresponds to the vehicle signage before the last intermediate destination, e.g. on circular routes). No specification: The source text from the annual timetable is used (per LineID and DirectionID).

DepartureStop (StartHst):	(optional) Short description of the departure stop
EndStop (EndHst):	(optional) Short description of the terminal stop
TripInfo (FahrtInfo):	(sub-element, optional) Additional information on the trip.
Reason (Ursache):	(optional) Reason for the unexpected departure from the visualisation area (e.g. early cancellation of the trip). Usually omitted under normal circumstances (end of the trip).

Because it is not always possible to uniquely reference the trip using *TripID (FahrtID)*, the optional fields *DepartureStop (StartHst)*, *EndStop (EndHst)* and *TripInfo (FahrtInfo)* shall be filled in if possible.

A message that deletes the above trip (65356) could look as follows:

```
<DatenAbrufenAntwort>
      <Bestaetigung Zst="2001-08-08T05:25:10" Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <VISNachricht AboID="25">
            <VISFahrtLoeschen Zst="2001-08-08T15:25:00">
                  <VISID>12345</VISID>
                  <FahrtID>
                        <FahrtBezeichner>64356</FahrtBezeichner>
                        <Betriebstag>2001-08-08</Betriebstag>
                  </FahrtID>
                  <LinienID>10</LinienID>
                  <LinienText>X10</LinienText>
                  <RichtungsID>Zoo</RichtungsID>
                  <RichtungsText>Zoologischer Garten</RichtungsText>
                  <Ursache>Motorschaden</Ursache>
            </VISFahrtLoeschen>
      </VISNachricht>
</DatenAbrufenAntwort>
```

6.5 General message service (GMS)

6.5.1 Introduction

The message service is used to connection general text information between the participating control centres. As with the other services, it is based on the client-server model. The server is the system that wishes to send data. The client is the recipient of these text messages. The message service differentiates between various information channels. These can be assigned different message types (errors, messages, warnings, traffic info, operational messages, etc.). Each channel can be subscribed to independently.

Messages can be revoked as well as sent. This makes sense in the situation where a message loses its validity before the scheduled end. If, for example, a message has been sent to set up

a train replacement bus service with a specific duration, the message can be revoked with a deletion message if the train replacement bus service ends early.

The message service consists solely of process data exchange. There is no foreseeable need for any exchange of reference data.

6.5.2 Message formats

The message service can convey three main types of data:

- Simple text
- Text with structuring features
- Freely defined XML content

Whilst XML contents have their own structures, simple texts can have additional formatting that allows subsequent structuring. CSV (comma separated values) files are an example of such a format. This represents a table format, where the column values are separated by commas and the lines by carriage return.

In order to be able to use different formats at run time, a so-called FormatID is used in order to identify the format in every message. For XML content it is necessary to use the URL of the schema as the FormatID.

Note: The VDV453 structure definition defines XML as an xsd:comlextype element, consisting of a sequence of any number of xsd:any elements. Verification of the structure definition can be activated as required by setting the processcontents attribute to "lax".

The structure definition, against which the content of XML is checked, is specified within the actual document (attribute xmlns="innernamespace" xsi:schemaLocation=" innernamespace locationinnernamespace.xsd"). Please note: When checking this function it became clear that most parsers currently only partially support the W3C standard.

Alternatively, the content of XML can be extracted and verified in a second stage.

If no FormatID is specified, the default of simple text without formatting is assumed (can be displayed directly).

Every implementation of a message service must at the very least support simple text format.

6.5.3 Operational data supply and management

In order to match the different classes of messages between the different agencies, it is necessary to define the so-called ChannelIDs (KanalID). These define a universal understanding of the message classes. The ChannelIDs (KanalIDs) must be incorporated into the data management of the participating systems. The matching process is bilateral.

The following example shows the definition of two message channels for communicating the transmission of messages to agency AVLC B:

ChannelID (KanalID)	Third-party agency code	Message class (internal)
1	AVLC B	Incidents
2	AVLC B	Other messages

Table 17 - Definition of the message channels

FormatIDs (FormatIDs) must also be agreed between the participating agencies if structured data is to be exchanged between the control centres.

6.5.4 Process data service

As in the other services, communication begins with the creation of a subscription. In the message service this is achieved by sending a GMSSubscription (AboAND) message contained in a SubscriptionRequest (AboAnfrage).

One or more channels can be specified in the GMSSubscription (AboAND), whose data is then to be retrieved.

If there is a message ready to be transmitted, this is signalled by the server by means of a DataReadyRequest (DatenBereitAnfrage). After reception and confirmation the messages are then retrieved by means of a DataSupplyRequest (DatenAbrufenAnfrage) by the client. The server supplies its information within the GMSMessage (ANDNachricht). The actual information is transported in the GMSNotification (ANDMeldung) elements. These have a separate ID and validity (ValidUntilTime (VerfallsZst)). Targeted deletion can be achieved via the ID of an individual GMSNotification (ANDMeldung) element (GMSNotificationDelete (ANDMeldungLoeschen) message).

6.5.4.1 Polling messages (GMSSubscription)

The retrieval of messages is initiated by the client. It creates a SubscriptionRequest (AboAnfrage) with one or more GMSSubscription (AboAND) sub-elements.

Definition of GMSSubscription (AboAND):

SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription for messages created by the request. The SubscriptionID (AboID) is given by the requesting system.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the time to which the subscription is valid.
ChannelID (KanalID):	(optional, multiple) Message channel; the system retrieves the messages of this channel.

The following example shows a request, which subscribes to two data channels in the time between 5:00 am and 10:00 pm:

```
<AboAnfrage Sender="AVLC A" Zst="2001-08-08T05:00:00">
      <AboAND AboID="25" VerfallZst="2001-08-08T22:00:00">
            <KanalID>Stoerungen</KanalID>
            <KanalID>Fahrgastunfaelle</KanalID>
      </AboAND>
</AboAnfrage>
```

The server confirms the successful set-up of the subscription with an Acknowledge (Bestaetigung) element in a SubscriptionReply (AboAntwort).

6.5.4.2 Messages in the message service (*GMSMessage*)

All messages that are sent from the server to the client are encapsulated within a GMSMessage (ANDNachricht). With that, the GMSMessage (ANDNachricht) defines the scope of the subscription.

The GMSMessage (ANDNachricht) can contain the following sub-elements:

- Sending a message GMSNotification (ANDMeldung)
- Cancelling a text message GMSNotificationDelete (ANDMeldungLoeschen)

Definition of GMSMessage (ANDNachricht):

SubscriptionID (AboID):	(attribute) The SubscriptionID (AboID) references the subscription for messages created by the request. The SubscriptionID (AboID) is given by the requesting system.
GMSNotification (ANDMeldung):	(sub-element, optional, multiple) Contains the useful data of a message.
GMSNotificationDelete (ANDMeldungLoeschen):	(sub-element, optional, multiple) Instructs the recipient to delete the message or to mark it as invalid or expired.

6.5.4.2.1 Transferring messages (GMSNotification)

If the server has messages that are ready to send, it sends a DataReadyRequest (DatenBereitAnfrage) to the client. This confirms receipt with an Acknowledge (Bestaetigung) element within a DataReadyReply (DatenBereitAntwort).

The client now polls the data via a DataSupplyRequest (DatenAbrufenAnfrage). The server responds with a DataSupplyAnswer (DatenAbrufenAntwort), which contains one or more GMSMessage (ANDNachricht) elements.

The message information is contained within GMSMessage (ANDNachricht) in one or more GMSNotification (ANDMeldung) elements. Messages contain the actual (useful) data. They have a unique ID (NotificationID (MeldungsID)) and a lifetime (ValidUntilTime (VerfallsZst)). They can be updated (overwritten) or declared invalid before they expire.

The NotificationID (MeldungsID) is unique for the entire GMS service (across channel boundaries).

Definition of GMSNotification (ANDMeldung):	
TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the message.
ValidUntilTimeStamp (VerfallZst):	(attribute) Specifies the planned time to which the message has operational validity (e.g. end of an action).
ChannelID (KanalID):	Unique reference to the message channel.
NotificationID (MeldungsID):	Uniquely references the message within the service. Used to delete or overwrite messages.
FormatID (FormatID):	(optional) Unique reference to the format used in the text.
MessageXML (MessageXML):	(optional) Contains the message in XML format.

The following example shows the transmission of two messages for the above subscription. Simple text without formatting is used:

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T10:00:00"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <ANDNachricht AboID="25">
            <ANDMeldung Zst="2001-08-08T9:59:30"
                        VerfallZst="2001-08-08T11:00:00">
                  <KanalID>Stoerungen</KanalID>
                  <MeldungsID>4711</MeldungsID>
                  Beidseitige Sperrung der Oberbaumbrücke wegen
                  Straßenarbeiten.
            </ANDMeldung>
            <ANDMeldung  Zst="2001-08-08T9:58:00"
                         VerfallZst="2001-08-11T12:00:00">
                  <MeldungsID>4712</MeldungsID>
                  <KanalID>Stoerungen</KanalID>
                  Dreitägige Sperrung des Innenstadtrings.
            </ANDMeldung>
      </ANDNachricht>
</DatenAbrufenAntwort>
```

The following example illustrates the use of formatted text (CSV):

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T10:00:00"
```

```
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <ANDNachricht AboID="25">
            <ANDMeldung Zst="2001-08-08T9:59:30"
                        VerfallZst="2001-08-08T11:00:00">
                  <KanalID>Stoerungen</KanalID>
                  <MeldungsID>4711</MeldungsID>
                  <FormatID>CSV</FormatID>
                  Beidseitige Sperrung,Oberbaumbrücke,Straßenarbeite,
            </ANDMeldung>
            <ANDMeldung Zst="2001-08-08T9:58:00"
                        VerfallZst="2001-08-11T12:00:00">
                  <MeldungsID>4712</MeldungsID>
                  <KanalID>Stoerungen</KanalID>
                  <FormatID>CSV</FormatID>
                  Sperrung, Innenstadtring, ,3 Tage
            </ANDMeldung>
      </ANDNachricht>
</DatenAbrufenAntwort>
```

And the same message again in XML format:

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T10:00:00"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <ANDNachricht AboID="25">
            <ANDMeldung Zst="2001-08-08T9:59:30"
                        VerfallZst="2001-08-08T11:00:00">
                  <KanalID>Stoerungen</KanalID>
                  <MeldungsID>4711</MeldungsID>
                  <FormatID>www.myschema.com/ANDSchema</FormatID>
                  <MessageXML>
                        <Vorkommnis> Beidseitige Sperrung</Vorkommnis>
                        <Ort>Oberbaumbrücke</Ort>
                        <Ursache>Straßenarbeiten</Ursache>
                  </MessageXML>
            </ANDMeldung>
            <ANDMeldung Zst="2001-08-08T9:58:00"
                        VerfallZst="2001-08-11T12:00:00">
                  <MeldungsID>4712</MeldungsID>
                  <KanalID>Stoerungen</KanalID>
                  <FormatID>www.myschema.com/ANDSchema</FormatID>
                  <MessageXML>
                        <Vorkommnis>Sperrung</Vorkommnis>
                        <Ort>Innenstadtring</Ort>
                        <Dauer>3 Tage</Dauer>
                  </MessageXML>
            </ANDMeldung>
      </ANDNachricht>
</DatenAbrufenAntwort>
```

If a GMSNotification (ANDMeldung) is resent, it overwrites the contents of the old message. This means it is possible to subsequently alter the validity of messages.

6.5.4.2.2 Deleting messages (GMSNotificationDelete)

If a message loses it's validity before it expires, the server can inform the client by sending a GMSNotificationDelete (ANDMeldungLoeschen) message. Updating (overwriting) a previously deleted message is not possible.

Referencing a message to be deleted is achieved via the NotificationID (MeldungsID).

Definition of GMSNotificationDelete (ANDMeldungLoeschen):

TimeStamp (Zst):	(attribute) The time stamp establishes the time of creation of the deletion message.
NotificationID (MeldungsID):	References the message to be deleted within the service.

The following example revokes the notification with ID 4711 of the above message:

```
<DatenAbrufenAntwort>
      <Bestaetigung
            Zst="2001-08-08T10:00:00"
            Ergebnis="ok"
            Fehlernummer="0">
      </Bestaetigung>
      <WeitereDaten>false</WeitereDaten>
      <ANDNachricht AboID="25">
            <ANDMeldungLoeschen     Zst="2001-08-08T9:59:30">
                  <MeldungsID>4711</MeldungsID>
            </ANDMeldungLoeschen>
      </ANDNachricht>
</DatenAbrufenAntwort>
```

7 Glossary

The glossary provides a description of the terms used within this project. All definitions required by the operation are based on the stipulations within the glossary. Any necessary amendments are carried out in accordance with the project management.

Term	Description
Actual timetable	Timetable generated from the scheduled timetable, which is supplemented with up-to-date information.
AVLC	Automatic Vehicle Location and Control System
Connection planning	Determination of the connections to be monitored (on the basis of the daily timetable).
Connection point (stop)	Stop or station at which several means of transport stop for the purpose of connections.
Control centre	Set-up to control and regulate the traffic-based and operational processes of a transport authority.
CP	Connection protection: Service for the operational exchange of connection protection data
Delay	Positive deviation from the scheduled timetable
Dispatch	Operative management to control traffic and operation
Display text	Displayed information concerning route, trip destination, departure time as well as special and service information
DPI sign	Dynamic passenger information: Service for the operational exchange of passenger information data.
DPI sign owner	AVLC system that controls the signs
DPI sign user	AVLC which displays its trips on the signs of another AVLC system
Early time	Negative deviation from the scheduled timetable
Feeder (trip)	Trip that brings passengers to a connection point
Fetcher (trip)	Trip that has taken on passengers from a feeder at the connection point.
GMS	General message service: Service for the exchange of operational messages between two control centres
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
Interchange time	Time required to change between vehicles at a connection point.
Meta data	The definitions and stipulations agreed between two transport authorities as a basis for data exchange.

Term	Description
OperationalDay	Time frame for the validity of timetables within an AVLC (can be different in different AVLC systems).
Pattern	Path along a route (from A to B); a route can have several patterns (e.g. from A to B via C).
Planned timetable	Scheduled timetable, planned in advance
Process data exchange	Exchange of current information between two AVLC systems.
Published timetable	Timetable that is posted at stops and stations; it depicts the transport service on offer
REF, reference data exchange	Exchange of the planned timetables between two AVLC systems
Route trip	Trip on a route
Run vehicle	Section of a train which separates from the remainder of the train assembly (traction) during the course of the trip and then continues as a single vehicle or as part of a new composite unit.
Special trip	Trip on one route or spanning several routes in which not all stops are necessarily served.
Stop name	Unique identification feature of a stop (e.g. stop number)
Subscription method	Communication method in the interface for the purpose of data exchange.
Timetable prediction	Preview of the actual timetable of a subsequent time section
Traction (composite train set)	Composite unit of coupled individual trips (see run vehicle).
Trip announcement	Generation and display of trip information based on the timetable data
TripName	Unique identification of a trip (e.g. trip number, trip ID)
Validity time frame of timetables	Fixed time frame for which a timetable is valid, e.g. a timetable period. Different transport authorities do not generally have matching validity time frames for timetables.
Via texts	Texts that inform passengers about the route taken by a particular trip.
VM	Visualisation: Service for exchanging process data for visualising third-party vehicles in a control centre
XML	Extended Markup Language

8 References

HTTP 22/06/2001

http://www.w3.org/Protocols/
Welcome page of the W3 Consortium on HTTP. Overview page for W3C documents concerning HTTP, with references to other sources.

HTTP/1.1 specification June 1999

http://www.ietf.org/rfc/rfc2616.html
Specification RFC2616 of the Hypertext-Transport-Protocol (HTTP) of the IETF.

XML 27/10/2001

http://www.w3.org/XML/
Welcome page of the W3 Consortium on XML. Overview page for W3C documents concerning XML, with references to other sources.

XML in 10 points 27/10/2001

http://www.w3.org/XML/1999/XML-in-10-points
Brief description of the meaning of XML. German, when using a browser with a German user interface.

XML structure definition part 0: Primer (Recommendation) 25/05/2002

http://www.w3.org/TR/xmlschema-0/
Non-normative, explanatory introduction to the capabilities of the XML structure definition, an XML-based Hypertext Mark-up Language, through which along with DTDs it is possible to define XML data structures that can be validated.

http://www.edition-w3c.de/TR/2001/REC-xmlschema-0-20010502/

XML structure definition part 1: Structures 25/05/2002

http://www.w3.org/TR/xmlschema-1/
Part 1 of the normative description of the XML structure definition. This section describes the layout and components of the XML structure definition.

http://www.edition-w3c.de/TR/2001/REC-xmlschema-1-20010502/

XML structure definition part 2: Data types 25/05/2002

http://www.w3.org/TR/xmlschema-2/
Part 2 of the normative description of the XML structure definition. This section describes the elementary data types of the XML structure.

http://www.edition-w3c.de/TR/2001/REC-xmlschema-2-20010502/

9 English aliases

The following tables contain the SIRI equivalents of the services and elements used in the VDV453 Recommendation, which are then employed in all international implementations of this standard.

9.1 Services

VDV 453		SIRI Equivalents		VDV 453 English Aliases	
Name	Abbr	Name	Abbr		
Anschluss-Sicherung Referenzdatendienst	REF-ANS	Connection Timetable Service	CT	reference data connection protection service	REF-CP
Anschluss-Sicherung Prozessdatendienst	ANS	Connection Monitoring Service	CM	process data connection protection service	CP
Dynamische Fahrgastinformation Referenzdatendienst	REF-DFI	Stop Timetable Service	ST	Reference Data Service	REF-DPI
Dynamische Fahrgastinformation Prozessdatendienst	DFI	Stop Monitoring Service	SM	Process Data Service	DPI
Visualisierung	VIS	Vehicle Monitoring Service	VM	Visualisation of Foreign Vehicles	VIS
Allgemeiner Nachrichtendienst	AND	General Message Service	GM	General Message Service	GMS

9.2 Root elements and complex sub-elements

VDV453 name	SIRI equivalent	VDV 453 English Aliases
AbbringerFahrtLoeschen	DistributorDepartureCancellation	FetcherTripDelete
AbbringerInfo	DistributorInfoGroup	FetcherInfo
Abbringernachricht	Connection MonitoringDistributorDelivery	FetcherMessage
AboAND	GeneralMessageSubscriptionRequest mit GeneralMessageRequest	GMSSubscription
AboAnfrage	SubscriptionRequest	SubscriptionRequest
AboAntwort	SubscriptionResponse	SubscriptionReply
AboASB	ConnectionMonitoringSubscriptionRequest	CPISubscription
AboASBRef	ConnectionTimetableSubscriptionRequest mit ConnectionTimetableRequest	CPIRefSubscription

VDV453 name	SIRI equivalent	VDV 453 English Aliases
AboAZB	StopMonitoringSubscriptionRequest mit StopMonitoringRequest	DISSubscription
AboAZBRef	StopTimetableSubscriptionRequest mit StopTimetableRequest	DISRefSubscription
AboLoeschen	TerminateSubscriptionRequest	DeleteSubscription
AboVIS	VehicleMonitoringSubscriptionRequest mit VehicleMonitoringRequest	VISSubscription
ANDMeldung	InfoMessage	GMSNotification
ANDMeldungLoeschen	InfoMessageCancellation	GMSNotificationDelete
ANDNachricht	GeneralMessageDelivery	GMSMessage
ASBFahrplan	ConnectionTimetableDelivery (mit TimetabledFeederArrival)	CPISchedule
ASBFahrplanlage	MonitoredFeederArrival	CPIDeviation
ASBFahrtLoeschen	MonitoredFeederArrivalCancellation	CPITripDelete
AZBFahrplan	StopTimetableDelivery (mit TimetabledStopVisit)	DISSchedule
AZBFahrplanlage	MonitoredStopVisit	DISDeviation
AZBFahrtLoeschen	MonitoredStopVisitCancellation	DISTripDelete
AZBLinienspezialtext	StopLineNotice	DISLineSpecialText
AZBLinienspezialtextLoeschen	StopLineNoticeCancellation	DISLineSpecialTextDelete
AZBNachricht	StopTimetableDelivery (ST Service) StopMonitoringDelivery (SM Service)	DISMessage
Bestaetigung	ResponseStatus	Acknowledge
DatenAbrufenAnfrage	DataSupplyRequest	DataSupplyRequest
DatenAbrufenAntwort	ServiceDelivery	DataSupplyAnswer
DatenBereitAnfrage	DataReadyNotification	DataReadyRequest
DatenBereitAntwort	DataReadyAcknowledgement	DataReadyReply
Direktruf	./. (to be added in VehicleJourney-Info)	DirectCall
FahrtFilter	ConnectingJourneyFilter	TripFilter
FahrtID	FramedVehicleJourneyRef	TripID
FahrtInfo	VehicleJourneyInfo	TripInfo
HaltepositionsAenderung	StoppingPositionChangedDeparture	StopPositionChange
Status	Status	Status
StatusAnfrage	CheckStatusRequest	StatusRequest
StatusAntwort	CheckStatusResponse	StatusReply
Traktion	TrainBlockPart	Trainset
VISFahrplanlage	VehicleActivity	VISDeviation
VISFahrtLoeschen	VehicleActivityCancellation	VISTripDelete
VISNachricht	VehicleMonitoringDelivery	VISMessage
WartetBis	WaitProlongedDeparture	WaitUntil

VDV453 name	SIRI equivalent	VDV 453 English Aliases
ZeitFilter	ConnectingTimeFilter	TimeFilter
Zubringernachricht	ConncetionTimetableDelivery (CT Service) ConnectionMonitoringFeederDelivery (CM Service)	FeederMessage

9.3 Additional elements

The equivalents for the additional elements are listed in the table below. It is recommended that the SIRI documentation is obtained for the purpose of comparison in order to identify any deviations in the definitions.

VDV453 name	SIRI equivalent	VDV 453 English Aliases
AbfahrtszeitASBPlan	AimedDepartureTime (CM Service)	ScheduledCPIDepartureTime
AbfahrtszeitASBPrognose	ExpectedDepartureTime (CM Service)	ExpectedCPIDepartureTime
AbfahrtszeitAZBPlan	AimedDepartureTime (SM Service)	ScheduledDISDepartureTime
AbfahrtszeitAZBPrognose	ExpectedDepartureTime (SM Service)	ExpectedDISDepartureTime
AbfahrtszeitAZBDisposition	./. (element to be added in MonitoredStopVisit)	AimedDISDepartureTime
AbfahrtszeitStartHst	OriginAimedDepartureTime	DepartureTimeStartStop
AbmeldeID	CleardownRef	DepartureNoticeID
AboID	SubscriptionRef mit SubscriptionIdentifier	SubscriptionID
AboLoeschenAlle	All (in TerminateSubscriptionRequest)	DeleteSubscriptionsAll
AktHst	StopPointRef (in MonitoredCall)	ActStop
AnkunftszeitASBPlan	AimedArrivalTime (CM Service)	ScheduledCPIArrivalTime
AnkunftszeitASBPrognose	ExpectedArrivalTime (CM Service)	ExpectedCPIArrivalTime
AnkunftszeitAZBPlan	AimedArrivalTime (SM Service)	ScheduledDISArrivalTime
AnkunftszeitAZBPrognose	ExpectedArrivalTime (SM Service)	ExpectedDISArrivalTime
AnkunftszeitZielHst	Destination AimedArrivalTime	ArrivalTimeDestinationStop
AnzahlFahrten	NumOfBlockParts	NumOfTrips
ASBID	ConnectionLinkRef	CPIID
AufASB	VehicleAtStop (CM Service)	AtCPIPoint
AufAZB	VehicleAtStop (SM Service)	AtDISPoint
AufHst	VehicleAtStop	AtStop
AZBID	MonitoringRef	DISID
Betreiber	OperatorRef	Operator
BetrieblicheFahrzeugnummer	VehicleRef (in EstimatedVehicleJourney /OperationalInfoGroup)	VehicleNumber
Betriebstag	DataFrameRef	OperationalDay
DatenBereit	Ref. 9.2: Status	DataAvailable
Datengueltigbis	ValidUntil	DataValidUntil
DatensatzAlle	AllData (in DataSupplyRequest)	AllData
Distanz	Percentage	Distance
EndHst	DestinationShortName	EndStop
Ergebnis	Status (in SubscriptionResponse)	Result
FahrtBezeichner	DatedVehicleJourneyRef	TripName
Fahrtspezialtext	StopVisitNote	TripSpecialText
FahrtStatus	Monitored	TripStatus
FahrzeugID	VehicleRef	VehicleID
Fehlernummer	ErrorCondition	ErrorNumber
Fehlertext	Description (in ResponseStatus)	ErrorText

VDV453 name	SIRI equivalent	VDV 453 English Aliases
FormatID	FormatRef	FormatID
FruehesteAbfahrtszeit	StartTime (DepartureWindow in StopTimetableRequest)	EarliestDepartureTime
FruehesteAnkunftszeit	StartTime (ArrivalWindow in ConnectionTimetableRequest) EarliestArrivalTime (ConnectingTimeFilter in ConnectionTimetableMonitoring)	EarliestArrivalTime
HaltepositionsText	ChangeNote, NewLocation	StopPositionText
HaltID	StopPointRef	StopID
HstSeqZaehler	VisitNumber	StopSeqCount
Hysterese	ChangeBeforeUpdate	Hysteresis
KanalID	InfoChannelRef	ChannelID
KuerzMoeglicherZyklus	ShortestPossibleCycle	ShortestPossibleCycleTime
KursNr	CourseOfJourneyRef	RunNumber
Latitude	Latitude (in VehicleLocation)	Latitude
LinienID	LineRef	LineID
LinienNr	ExternalLineRef	LineNumber
Linienspezialtext	LineNote	LineSpecialText
LinienText	PublishedLineName	LineText
Longitude	Longitude (in VehicleLocation)	Longitude
MaxAnzahlFahrten	MaximumNumberOfCalls	MaxNumOfTrips
MaxTextLaenge	MaximumTextLength	MaxTextLength
MeldungsID	InfoMessageIdentifier	NotificationID
NachHst	StopPointRef (in OnwardCall)	NextStop
Position	PositionOfTrainBlockPart	Position
ProduktID	ProductCategoryRef	ProductID
Prognose	Estimation	Prediction
Prioritaet	./. (new element to be added in	Priority
RichtungsID	DirectionRef	DirectionID
RichtungsText	DirectionName	DirectionText
Sender	SubscriberRef	Sender
ServiceMerkmal	ServiceFeatureRef	ServiceAttribute
SpaetesteAbbringerInfo	./. (new element to be added in MonitoredFeederArrival)	LatestFetcherInfoTime
SpaetesteAbfahrtszeit	EndTime (DepartureWindow in StopTimetableRequest)	LatestDepartureTime
SpaetesteAnkunftszeit	EndTime (ArrivalWindow in ConnectionTimetableRequest LatestArrivalTime (ConnectingTimeFilter in ConnectionTimetableMonitoring)	LatestArrivalTime
StartDienstZst	ServiceStartedTime	StartServiceTimeStamp
Sprachausgabe	./. (New element to be added in StopLineNotice)	SpeechOutput
StartHst	OriginRef (als Identifier) OriginShortName (als Kurzbezeichner)	DepartureStop
StartHstLang	OriginName	DepartureStopLong
Stauindikator	InCongestion	QueueIndicator
TraktionsID	TrainPartRef	TrainsetID
UmlaufNr	BlockRef	BlockNumber
UmsteigeWillige	NumberOfTransferPassengers	TransferPassengers
Ursache	Reason	Reason

VDV453 name	SIRI equivalent	VDV 453 English Aliases
VerfallZst	InitialTerminationTime, ValidUntilTime	ValidUntilTimeStamp
Verlaesslichkeit		Reliability
Verspaetung	Delay	Delay
ViaHst	Vianame	ViaStop
VISID	VehicleMonitoringRef	VISID
Vorschauzeit	PreviewIntervall	PreviewTime
WeitereDaten	MoreData	PendingData
ZielHst	DestinationShortName	DestinationStop
ZielHstLang	DestinationName	DestinationStopLong
Zst	RecordedAtTime	TimeStamp
ZubringerHstLang	StopPointName	FeederStopLong
Zyklus	UpdateIntervall	Cycle