MobilityBroker Interface with Smartcar Extension based on IXSI - Interface for X-Sharing Information Version 4

Markus Beutel	**RWTH Aachen University**
Sevket Gökay	**RWTH Aachen University**
Peter von Grumbkow	**HaCon Ing.-Ges. mbH**
Dirk Hillbrecht	**Cantamen GmbH**
Karl-Heinz Krempels	**RWTH Aachen University**
Christian Samsel	**RWTH Aachen University**
Christoph Terwelp	**RWTH Aachen University**
Heike Twele	**HaCon Ing.-Ges. mbH**
Herhard Wagner	**Cantamen GmbH**

Date: January 31, 2017 Document Version: 0.20

The respective XML schema is available in the github repository https://github.com/RWTH-i5-IDSG/ixsi/.

econnect Germany

mobility broker

Gefördert durch:

Bundesministerium
für Wirtschaft
und Technologie

aufgrund eines Beschlusses
des Deutschen Bundestages

Contents

1 Overview

Aim of this interface specification is to couple information systems for rental vehicles with travel information systems. The reason for the coupling is to serve the trend driven requirement of creating intermodal travel chains in order to integrate rental systems to travel information systems.

The specification consists of:

- A Role Model of the involved partners,
- A recommendation for a service hierarchy of information exchange with different qualities,
- Interaction sequences to depict message ordering between the partners in order to serve the information coupling based on the specified service levels,
- The specification of the Data Model for messages of the interaction protocol,
- Specification of suited technologies to visualize data, the communication between: the information systems, the implementation of the interaction protocol and the concluding information handling which are to be linked
- Tables of allowed values for enumerations.

This version of IXSI contains the following - mobility broker specific - extensions:

- Vehicle activation
- Exchange of consumption data
- Subscription of external bookings
- Creation and locking of VRS (Vehicle Rental System) users
- Synchronisation of vehicle settings

This version of IXSI contains the following - smartcar specific - extensions:

- Synchronisation of vehicle settings
- Remote configuration navigation system and route monitoring
- Dialog public transport alternative (TODO)

2 Role Model

The Role Model describes the occuring roles for the integration scenario of a vehicle rental system (e.g. CarSharing) into a travel information system.

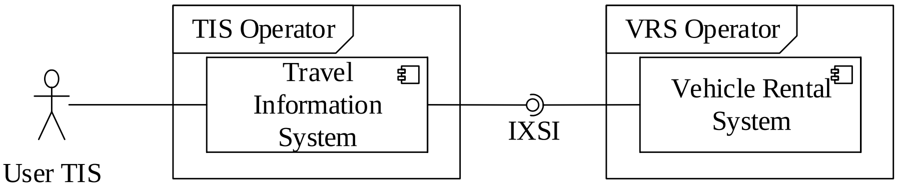


Figure 2.1: Overview Roles.

Travel Information System

The Travel Information System (TIS) is a information system, responsible for travel inquiries and covers: the unification of travel options, the construction of travel chains, the calculation of a total price for a travel chain, the reservation of travel options for separated elements of the travel chain, the processing of data for representations in user interfaces and finally the presentation of information.

Use Cases

- User inquiries the TIS concerning preferred mobility options and search conditions, using search filters and preferences. Search filters and preferences might be starting- and target- location, departure- and arrival- time, transportation modes, amount of transfer flights, price range, etc. The results are provided via user-interfaces of the TIS in form of travel chains.

Vehicle Rental System

The Vehicle Rental System (VRS) is a information system, responsible for managing and booking of rental vehicles. Vehicles may vary in type or might be attached or unattached to stations.

Use Cases

- A user books a vehicle via VRS for contractual prices, times and stations and uses it.
- A user inquiries the availability of a vehicle via VRS.

User TIS

User TIS – represents a legal person, which is authorized to book and utilize a travel chain under use of selected modes of transportation.

Use Cases

- User sets up a travel inquiry towards the TIS.
- User books a travel via TIS.

User VRS

User VRS - represents a legal person, which is authorized to rent and use a vehicles.

Use Cases

- User sets up a travel inquiry towards the VRS.
- User books a vehicle via VRS.

Operator TIS

Operator TIS – provides the TIS as a service for transportation service providers.

Operator VRS

Operator VRS - provides the VRS as a service for vehicle rental companies.

Vehicle

Vehicle - provided to the customer via the VRS on a station as a mobility offer.

3 Hierarchical Model

The hierarchical model describes different qualities of information coupling, based on service groups and serves as a recommendation concerning possible stages of implementation. To realize a coupling between VRS and TIS, at least service 1 (static data) of both interacting parties has to be supported. Dependencies between different services are depicted in fig. 3.1.

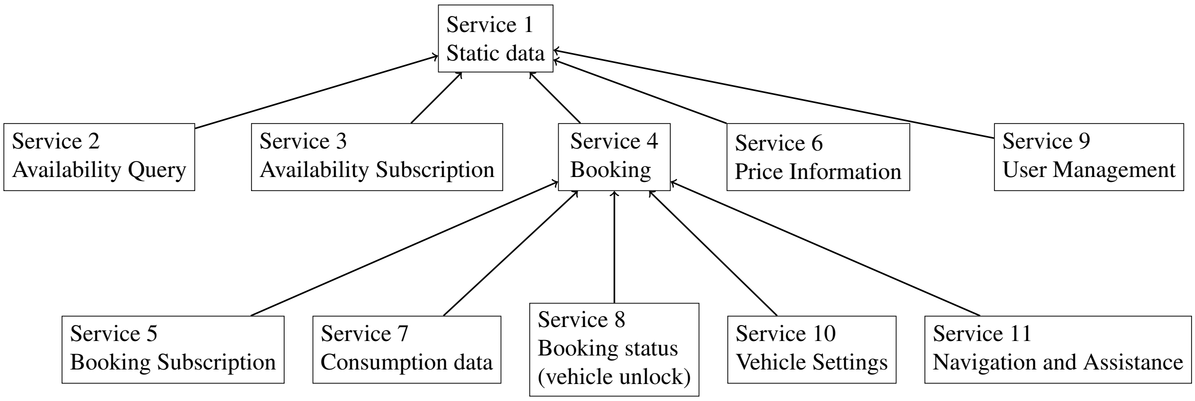


Figure 3.1: Service Dependencies

3.1 Base Service A – Session Handling

Service A enables the authentication of end-customers towards the VRS.

Functions

- Open / Close Session

Dependencies

none

3.2 Base Service B – Subscription Handling

Service B contains a function to check the status of a subscription connection (heartbeat)

Functions

- Heartbeat

Dependencies

none

3.3 Base Service C – Tokens

Service C contains a function to create authentication tokens for users, which can be saved / transferred instead of plaintext passwords.

Functions

- Creation of Tokens

Dependencies

none

3.4 Service 1 – Static Data

Service 1 serves the information exchange across vehicle rental companies and static data of booking targets. These include provider-, position- and vehicle-data.

For example, service 1 can be used, to display messages solely for existing locations of an VRS-provider in a TIS.

Functions

- Call of booking targets and provider information.

Dependencies

none

3.5 Service 2 – Availability Query

Service 2 serves for synchronous calls of availability information. The actual availability times of booking targets are called from the VRS during the travel inquiry through the TIS.

The actual availability times of booking targets are called during the travel inquiry by the TIS at the VRS.

Functions

- Calls of availabilities of booking targets
- Calls of location capacities (Service 2a)

Dependencies

- Service 1

3.6 Service 3 – Availability Subscription

Service 3 serves the asynchronous exchange of availability information.

To accelerate the travel inquiry, the TIS can subscribe availability timescales of booking targets to avoid a query during the travel inquiry. After subscribing for an amount of booking targets, the VRS informs continuously about changes in availability timescales.

Functions

- Availability subscription
- Availability information (push)
- Subscription of location capacities (Service 3a)

Dependencies

- Service 1
- Base Service B

3.7 Service 4 – Booking

Service 4 serves booking, booking changes and canceling of vehicles via the TIS on behalf of the customer of a VRS.

The booking of a vehicle requires that a customer authenticates himself towards the VRS. Therefore, authentication information is forwarded from the TIS to the VRS. The secret (Password, key, etc.) of the customer is not allowed to be saved due to security reasons. The TIS receives an authentication-token in case of a successful authentication of a user. It can either be saved on the TIS or the user's device. Using the authentication-token, the TIS can proceed queries concerning bookings and canceling. To allow booking changes, a booking can be replaced by another booking via a booking change request. In case of a booking errors, it has to be ensured that the initial booking retains it's validity.

Functions

- Authentication of users towards the VRS
- Booking Query
- Query concerning booking changes/ canceling
- Subscription of external Bookings (Service 4a)

Dependencies

- Service 1

3.8 Service 5 – Booking Subscription

Service 5 serves the subscription of booking changes.

The TIS is able to subscribe conducted bookings at the VRS to inform users in case of changes, e.g., damaged vehicles.

Functions

- Booking subscriptions
- Booking alert (Push)

Dependencies

- Service 4
- Base Service B

3.9 Service 6 – Price Information

Service 6 serves the price information of rental services.

Through the transmission of starting- and target-location as well as departure- and arrival-time of the travel, the TIS is able to query price information at the VRS to inform the user. The VRS responds with an overall price and eventually individual price parts.

Functions

- Query of Prices

Dependencies

- Service 1

3.10 Service 7 – Subscription Consumption Data/ Billing

Service 7 serves to exchange consumption data, e.g, usage duration and distance. Through the transmission of consumption data, the TIS is able to create bills for end-customers. Therefore, the TIS subscribes consumption data of an executed booking at the VRS and receives the corresponding consumption data automatically.

Functions

- Subscription consumption data
- Exchange consumption data (Push)

Dependencies

- Service 4
- Base service B

3.11 Service 8 – Booking State Change (Unlocking / Pausing / Finishing)

Service 8 serves to unlock booking of respective vehicles (probably via key case) by the TIS. This allows to unlock as well as to return the vehicle during the usage phase of the TIS e.g., via a mobile application. Furthermore, a booking can be paused, which locks the vehicle physically without finishing the booking. The state always refers to a preceding booking and an authentication is required.

Functions

- Unlock booking
- Pause booking
- Finish booking

Dependencies

- Service 4
- Base service A

3.12 Service 9 – User Management

Service 9 serves to create and lock of user accounts of the VRS through the TIS. Therefore, a user just has to register himself at the TIS operator. This registration is valid even for usage of the VRS.

Functions

- Create user account
- Lock user account

Dependencies

- Service 1
- Base service A

3.13 Service 10 – Vehicle or Booking Settings

Service 10 serves the management of vehicle settings. The TIS is able to transfer vehicle settings (e.g., air condition temperature, preferred radio stations) to the VRS and subscribe to settings of existing bookings.

Functions

- Set vehicle settings
- Subscribe to vehicle settings

Dependencies

- Service 4
- Base service A

3.14 Service 11 – Remote Configuration of Navigation System and Journey Progress Monitoring

Service 11 serves the integration of the vehicle navigation system into an intermodal journey. The TIS is able to transfer the target location to the vehicle after the booking, so that the traveler does not need to enter the location by himself. Additionally, the TIS is able to monitor the progress of a journey, in order to react in case of delays.

Functions

- Set navigation target of vehicle
- Subscribe to route progress

Dependencies

- Service 4
- Base service A

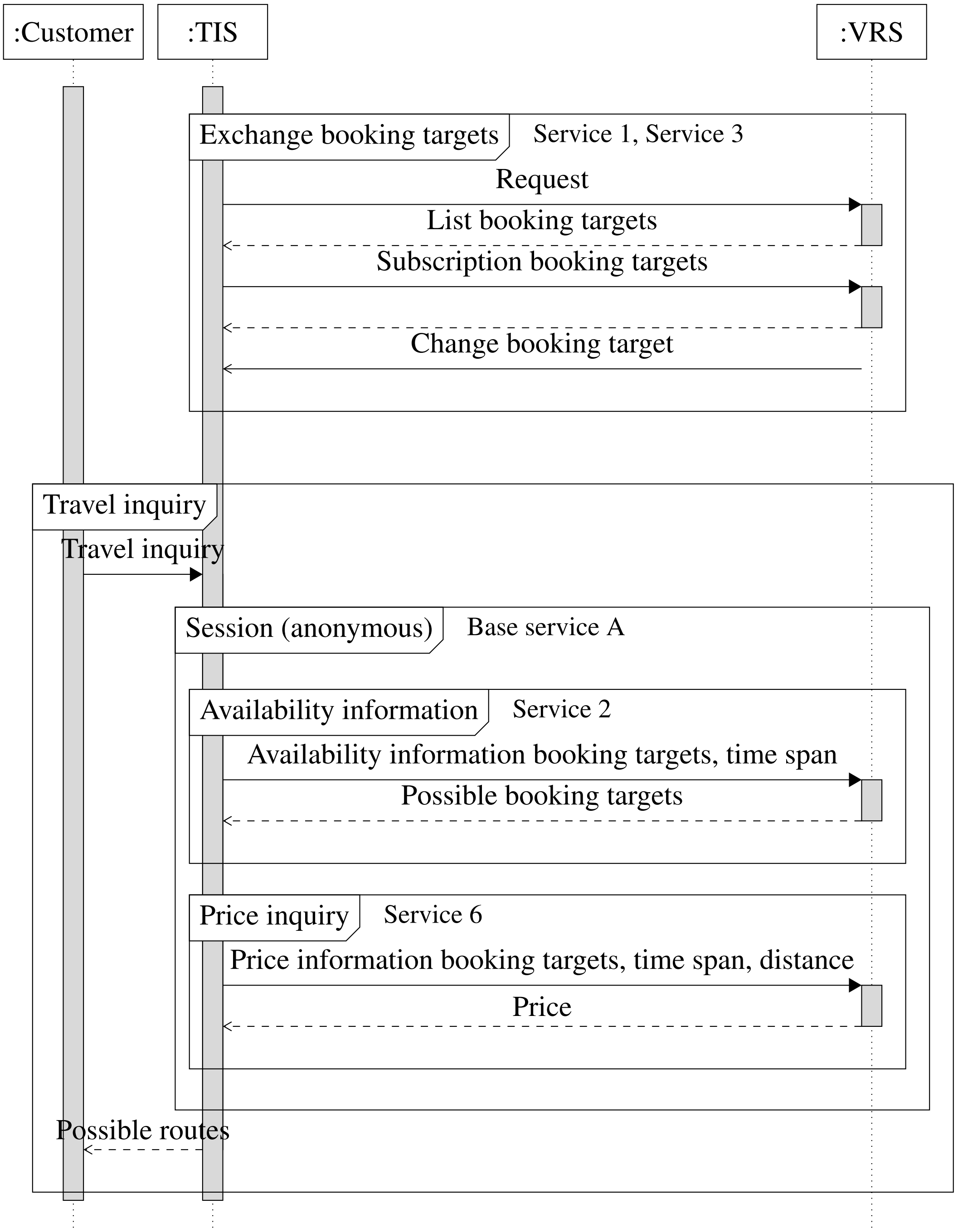
4 Interaction Sequences

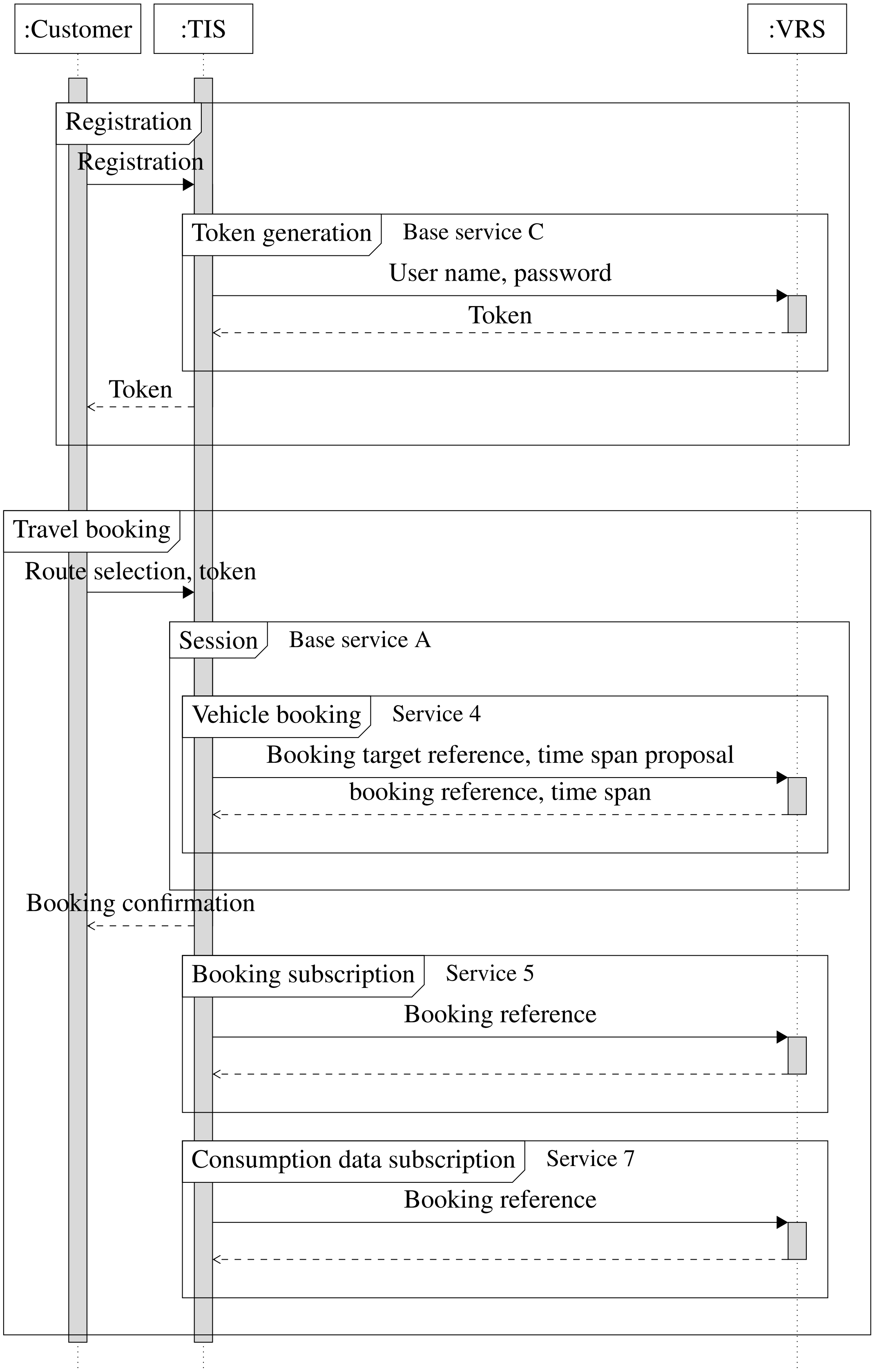
This section provides an overview about the interaction schemes, used by IXSI. For simplification, these interaction schemes are described by interaction sequence diagrams without using the technical terms of function calls. In principle, two types of interactions are used: The simple and well-known request/response- or the query- interaction scheme, whereby every call of the client (in this case TIS) is followed by exactly one response of the server (VRS). Furthermore, the subscription-scheme, whereby one object is subscribed by the client once and updated continuously by the server. Hereby, the communication channel is opened all the time.

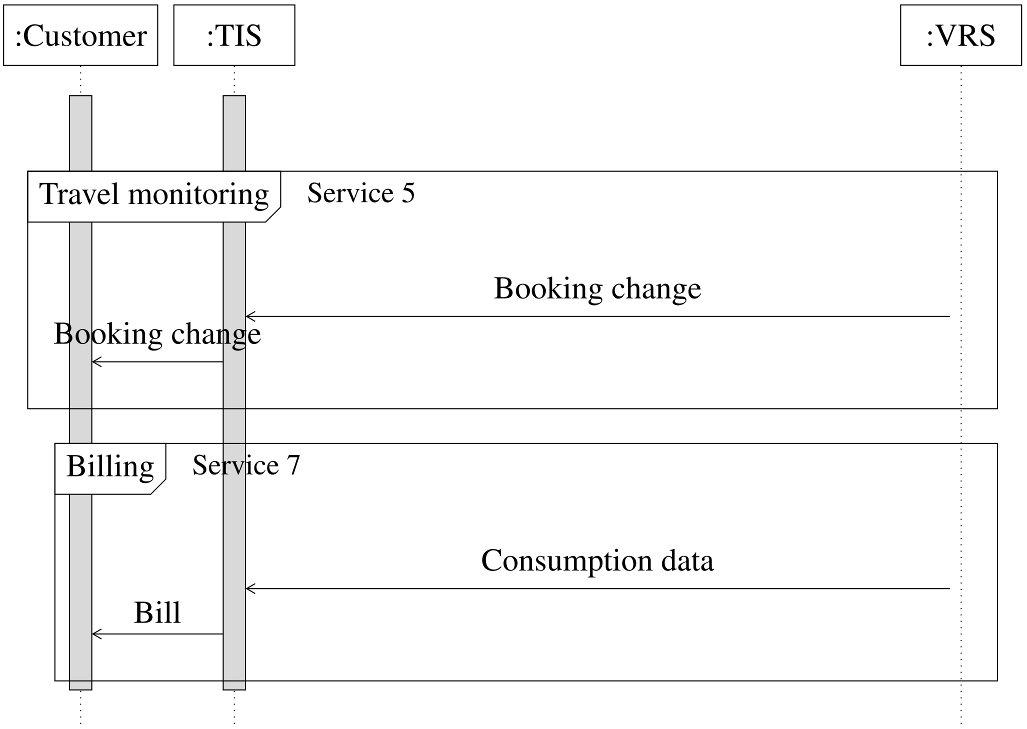
4.1 Overview

The following sequence diagram gives an exemplary overview of information coupling, possibly enabled by IXSI. The used services are going to be described in detail in the following chapters. In this use case, a customer sets up a travel inquiry, books a respective travel and gets informed concerning relevant booking changes.

In the first block *Exchange booking targets*, the booking targets, provided by the VRS are exchanged with the TIS and relevant booking targets are subscribed (see. sections 4.2 and 4.4). This happens proactive, without involving the customer. In block *Travel Inquiry*, a customer sets up a travel inquiry with e.g., his mobile device, on the TIS. Thereby, various rental vehicles are suitable for booking, whose availabilities are requested synchronously towards the VRS. The TIS additionally requests price information for available vehicles. As a result, the TIS provides a selection of possible routes / connections to the customer. Since this is a communication triggered by the customer, a session which proceeds the queries is created implicitly. An anonymous session is used, because the customer did not register on his device beforehand. In block *Travel booking*, a customer has chosen a travel route and intends to book it. For this purpose, he registers on his mobile device, whereby a token is generated (Block *Registration*). With this token, a (not anonymous) session is created, which proceeds the booking operation. For that, the TIS forwards booking reference and a time proposal to the VRS. This sends a booking confirmation, which is forwarded from the TIS to the customer. In addition, the TIS subscribes the respective booking on the VRS. In the last block *Travel monitoring*, the customer gets informed concerning booking changes, which are received by the TIS, coming from the VRS.

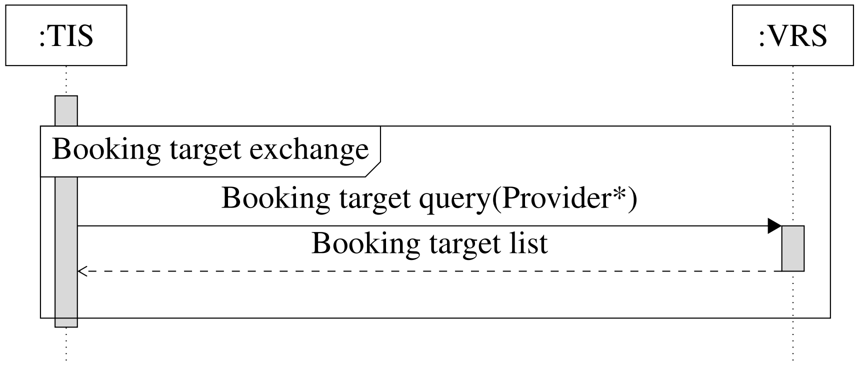

:Customer
:TIS
:VRS
Exchange booking targets
Service 1, Service 3
Request
List booking targets
Subscription booking targets
Change booking target
Travel inquiry
Travel inquiry
Session (anonymous)
Base service A
Availability information
Service 2
Availability information booking targets, time span
Possible booking targets
Price inquiry
Service 6
Price information booking targets, time span, distance
Price
Possible routes


:Customer
:TIS
:VRS
Registration
Registration
Token generation
Base service C
User name, password
Token
Token
Travel booking
Route selection, token
Session
Base service A
Vehicle booking
Service 4
Booking target reference, time span proposal
booking reference, time span
Booking confirmation
Booking subscription
Service 5
Booking reference
Consumption data subscription
Service 7
Booking reference



4.2 Service 1 – Static Data

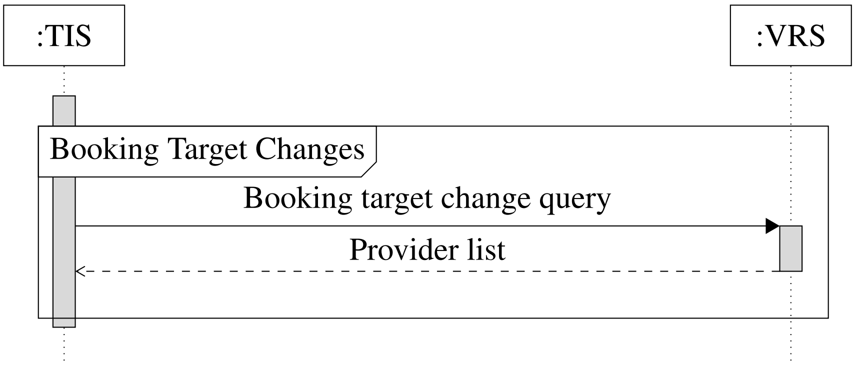
Booking Target Query



** optional*

Basis for information coupling is the exchange of booking targets. Booking targets are a logical representation of one or several vehicles with common characteristics, e.g., provided by the same provider, same vehicle type or same rental station. These characteristics are static. To receive booking target information concerning a certain provider solely, it is possible to filter concerning the respective provider. The transmission is initiated by the TIS.

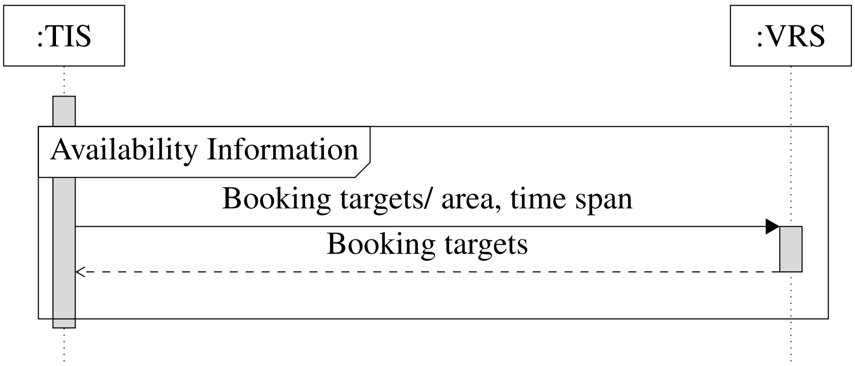
Booking Target Change Query



To avoid the transmission of all booking information per interval, with the help of the query `ChangedProviders` it is possible to request the provider, related to changes since a specific point in time, set by the parameter `timestamp`. A provider reference is returned, which in turn can be transferred as a parameter by the function call `BookingTargetsInfo`.

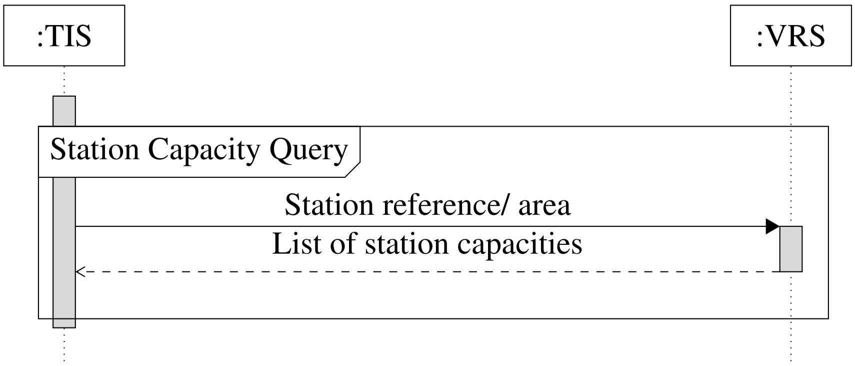
4.3 Service 2 – Availability Information

Availability Query



To request specific availabilities, the TIS sends a query, which either contains a list of booking targets or a geographic area in form of a proximity search or as a rectangle and a required time period. Without specification, the availabilities of all booking targets are returned. As a response, the VRS returns a list of booking targets and their availabilities.

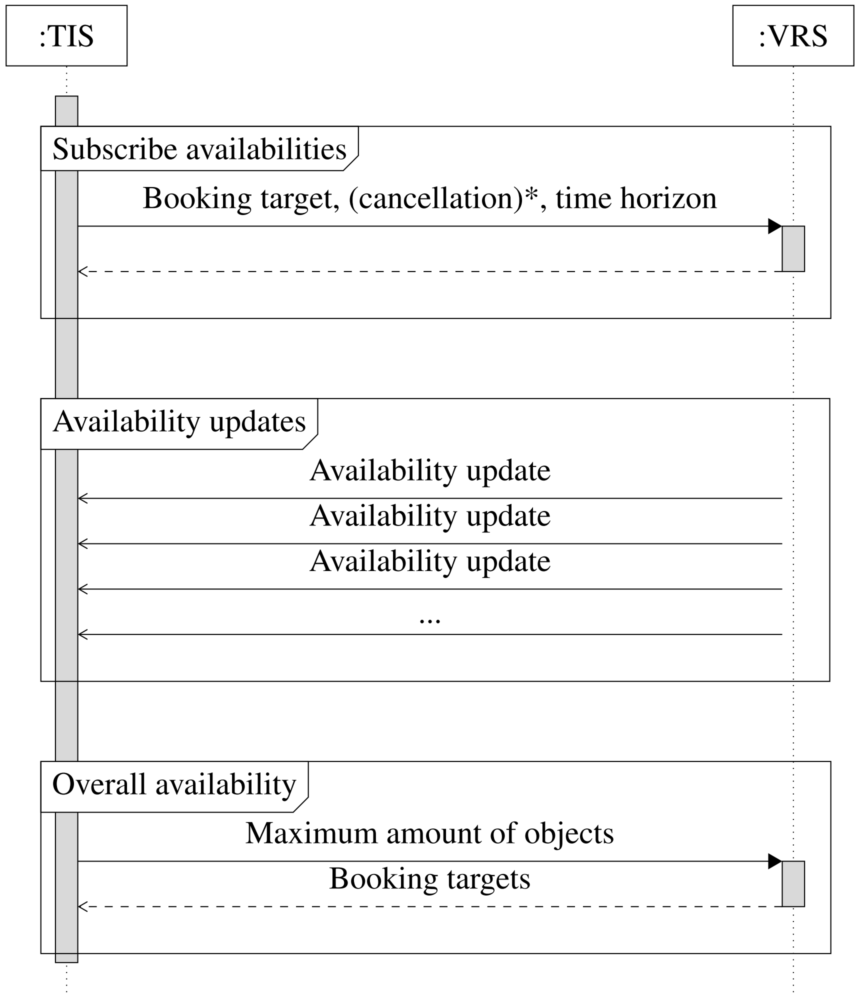
Current Station Capacity Query (Service 2a)



The TIS is able to request current capacities from rental stations, e.g., for map illustration. For this purpose, a list containing station IDs or an area has to be transferred. As a response, a list of locations and their current amount of available vehicles is returned.

4.4 Service 3 – Availability Subscription

Availability Subscription

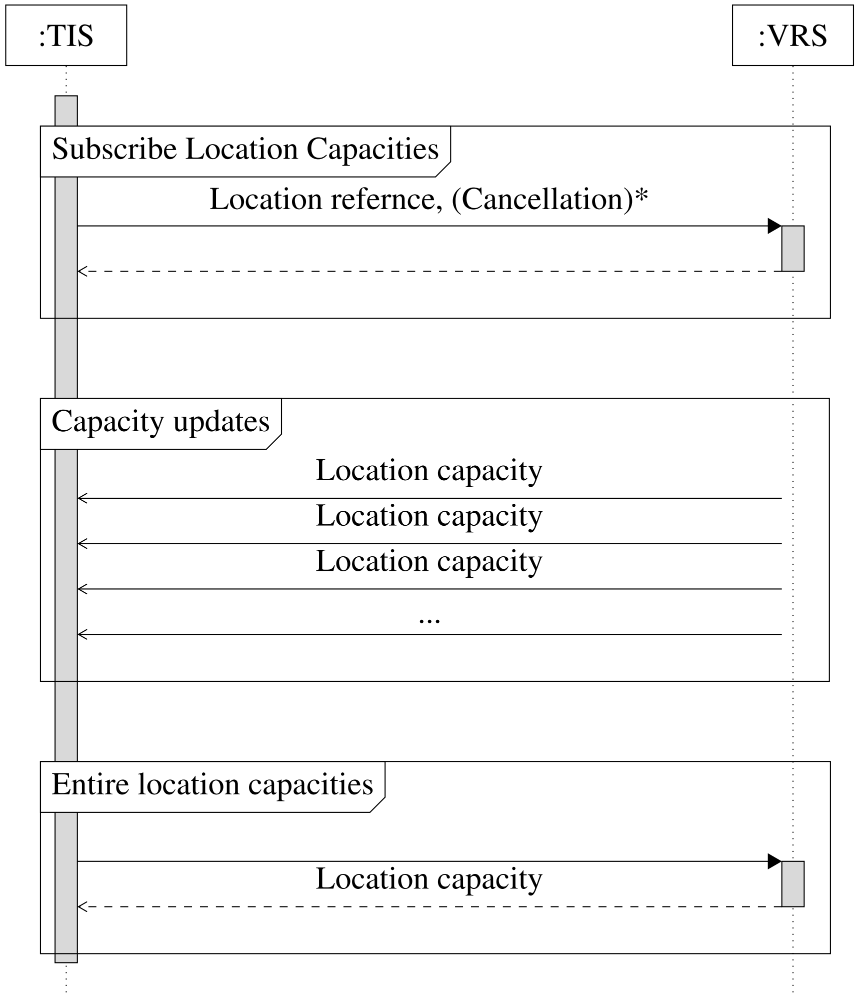


The TIS can subscribe to information concerning booking targets to get informed about availability changes immediately. In principle, this serves to enable responses to travel inquiries without additional (synchronous) requests towards the VRS.

Through the initial query `AvailabilitySubscriptionRequest` a subscription is started. For this purpose, the TIS forwards the related booking reference. A subscription can be canceled, by setting the flag "Cancellation". In case of changes concerning availabilities, the VRS forwards asynchronously `AvailabilityPushMessage`. These are delivered via the same communication channel as upon the subscription was created beforehand. By cancellation of the communication channel, all subscriptions become obsolete.

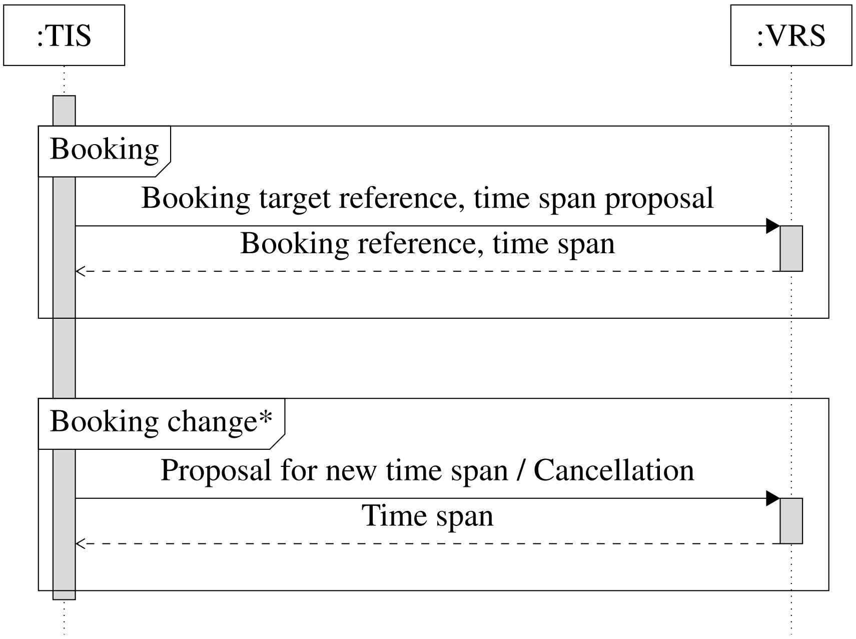
For the initial synchronization of the availabilites the TIS can call the function `CompleteAvailabilityRequest`.

Location Capacity Subscription (Service 3a)

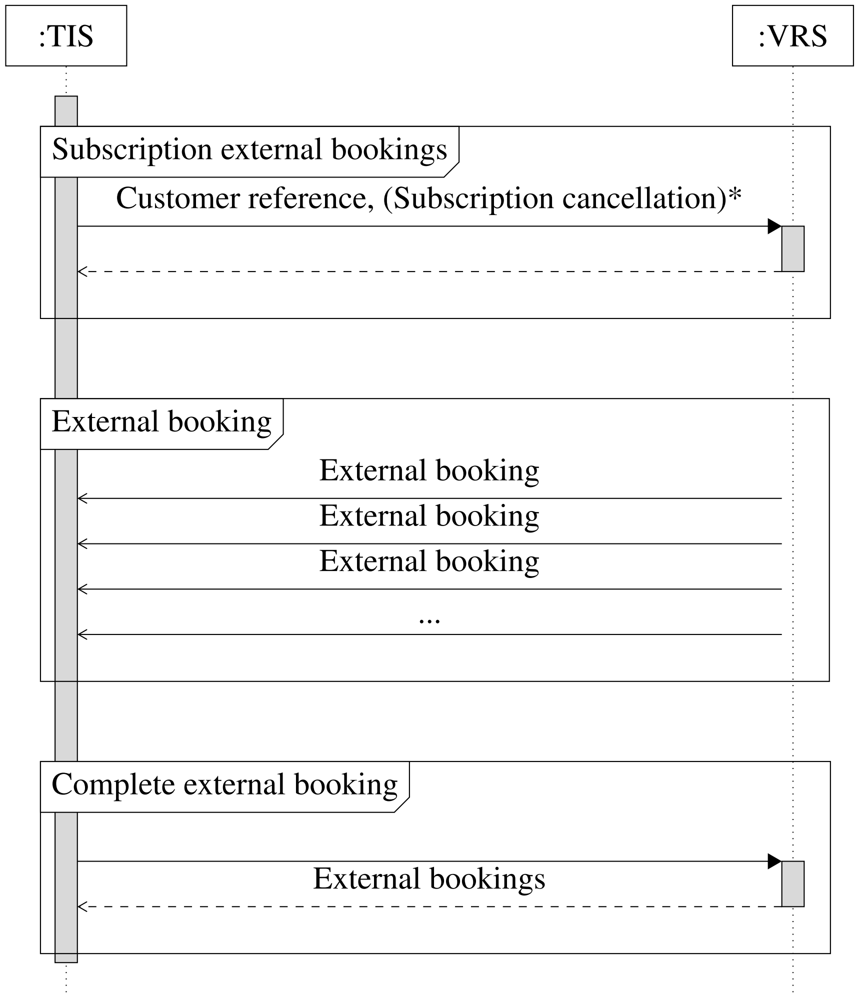


The TIS can subscribe capacity information of locations. The interaction procedure is analogous to section 4.4.

4.5 Service 4 – Booking/ Booking Change


:TIS
:VRS
Booking
Booking target reference, time span proposal
Booking reference, time span
Booking change*
Proposal for new time span / Cancellation
Time span

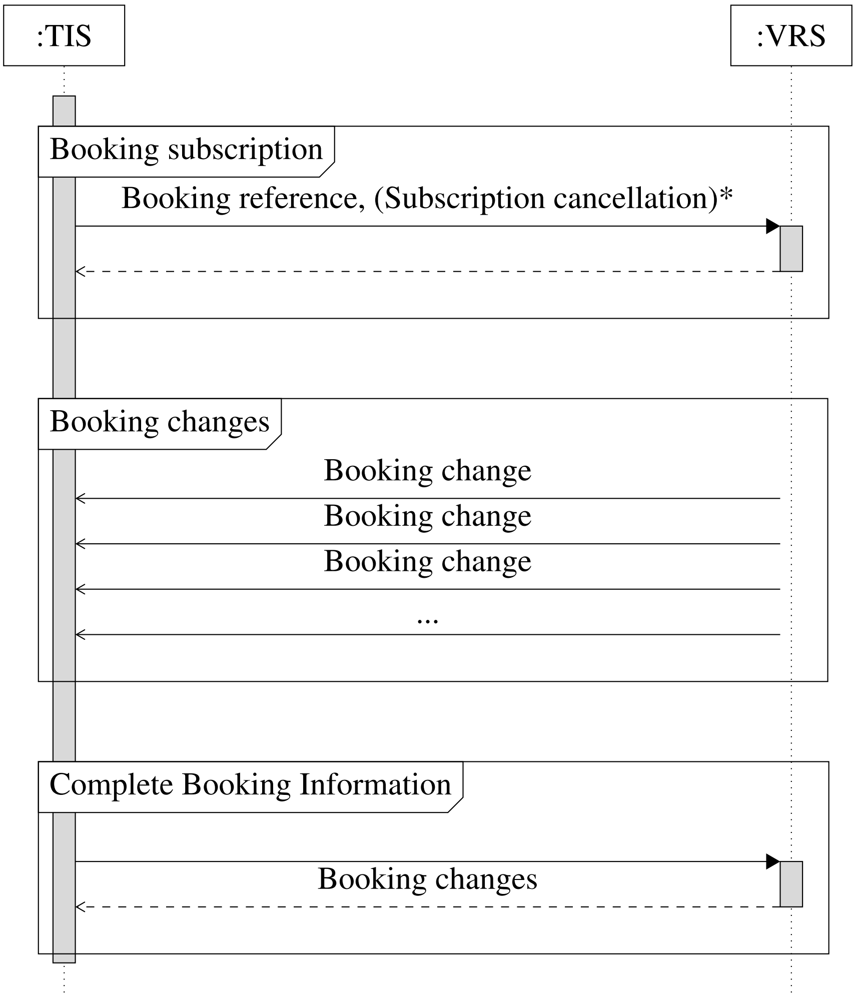
Subscription External Bookings (Service 4a)


:TIS
:VRS
Subscription external bookings
Customer reference, (Subscription cancellation)*
External booking
External booking
External booking
External booking
...
Complete external booking
External bookings

To book a vehicle in customer order, it is necessary that the TIS authenticates the customers towards the VRS. For this purpose, there are three possibilities, which are further specified in section 5.13. In this example, a session is opened explicitly and closed after the transaction. After that, a booking can be utilized by calling `Booking` with provision of the respective booking target ID and a time span proposal. “Vorschlag” because the VRS is able to change the time span to the used booking grid. As response, the used booking reference and the actual time span is returned. The booking reference can be used for monitoring of the booking (see section 4.6). For changes of the booking time span or for cancellation, `ChangeBooking` can be called. In case of changes of the booking target, a cancellation or a new booking becomes necessary.

The TIS can subscribe to external bookings (e.g., directly at the station without assist of the TIS) of the VRS, in order to subscribe to booking changes or request corresponding consumption data. Therefore one or more customer references are handed over.

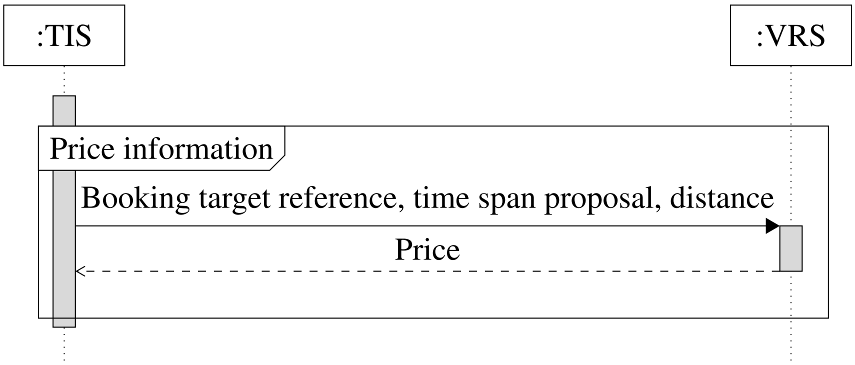
4.6 Service 5 – Booking Subscription



The TIS can subscribe to changes concerning bookings to provide this information to customers and probably propose alternatives. For example in case of a technical failure of a vehicle, the VRS can inform the TIS that a booking is not longer possible. In addition, it is possible to set a booking to “possible again”. Ultimately the booking can only be canceled customer himself.

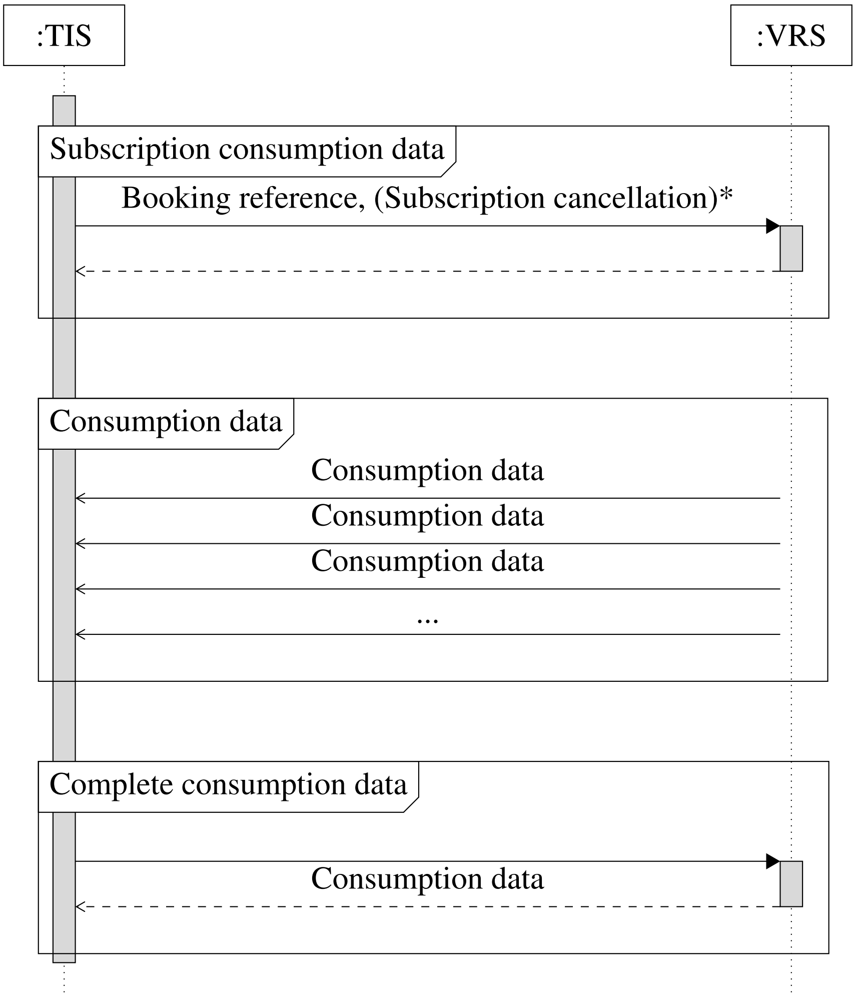
The interaction procedure is analogous to section 4.4.

4.7 Service 6 – Price Information



With the query `PriceInformationRequest`, the TIS can request price information towards the VRS on the basis of booking target ID, time span and traveled distance. In case of a customer authentication e.g., through `OpenSession` took place beforehand, the price request has to be replied-to accordingly to the customer contract.

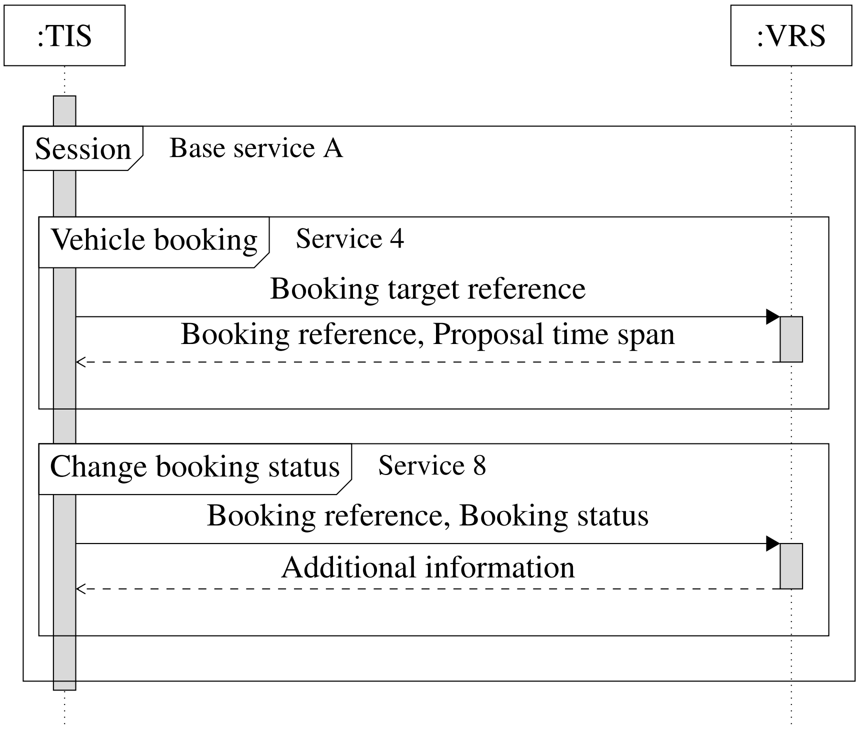
4.8 Service 7 – Subscription Consumption Data


:TIS
:VRS
Subscription consumption data
Booking reference, (Subscription cancellation)*
Consumption data
Consumption data
Consumption data
Consumption data
...
Complete consumption data
Consumption data

The TIS is able to subscribe for consumption data (e.g., usage duration, distance, others) for a booking. As soon as new consumption data occurs (e.g., by returning a vehicle), the VRS informs the TIS concerning the respective consumption. Concerning one booking, multiple consumption data information can be forwarded. New consumption datasets of one booking invalidate all previous datasets and accordingly need to be complete.

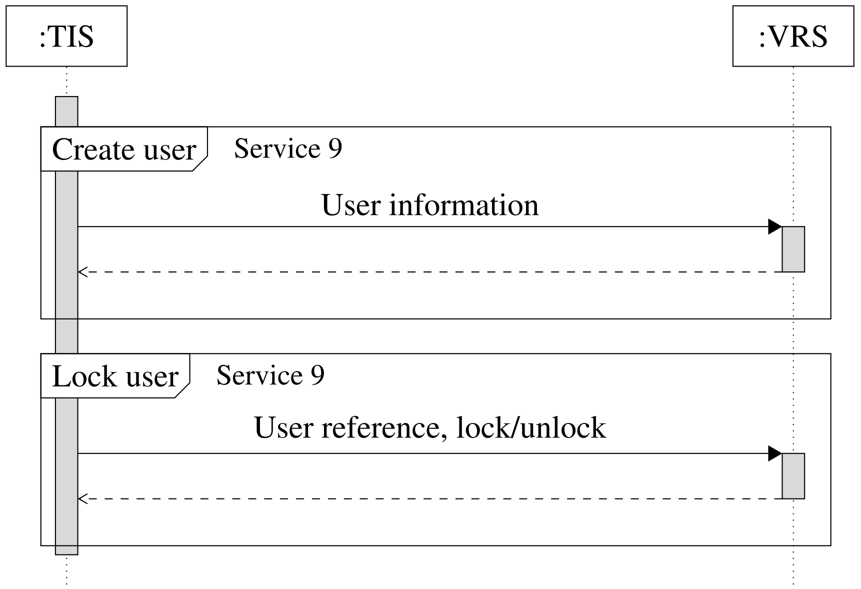
The interaction sequence is analogous to section 4.4.

4.9 Service 8 – Change Booking Status



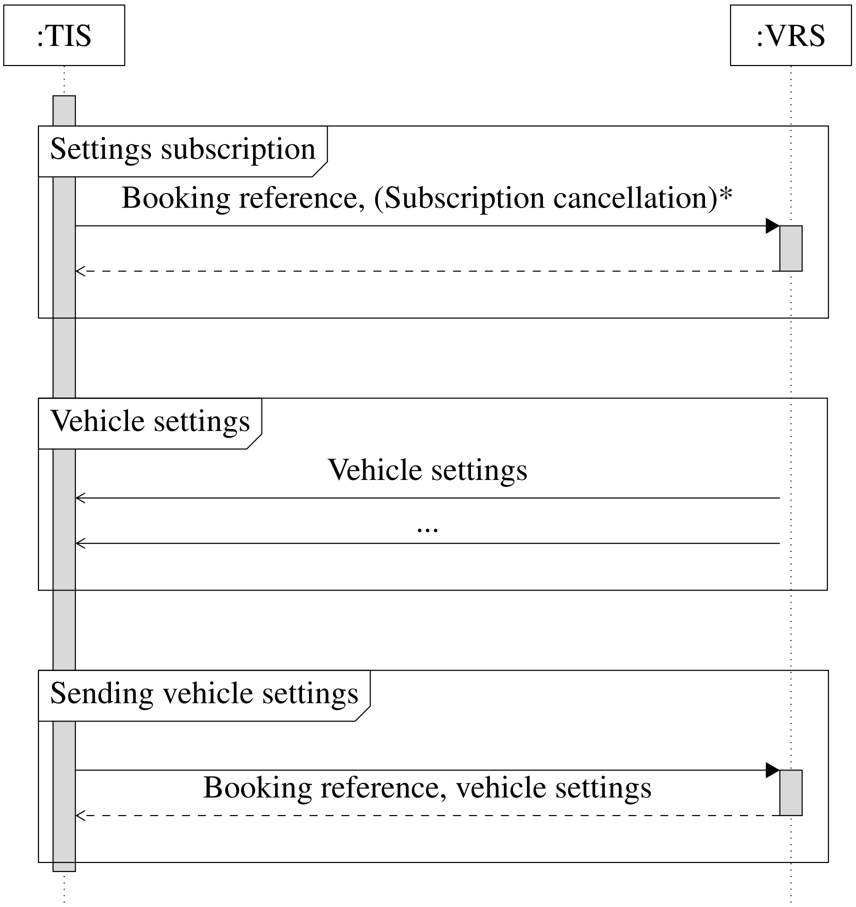
To unlock a booking target (vehicle or key box) via the TIS, a booking for this booking target has to be present (probably this has to be proceeded in a transparent way towards the user). Using a reference for this booking, the booking target can be unlocked. Therefore, a authentication is mandatory. Pausing and finishing of a booking happens analogous.

4.10 Service 9 – User Management



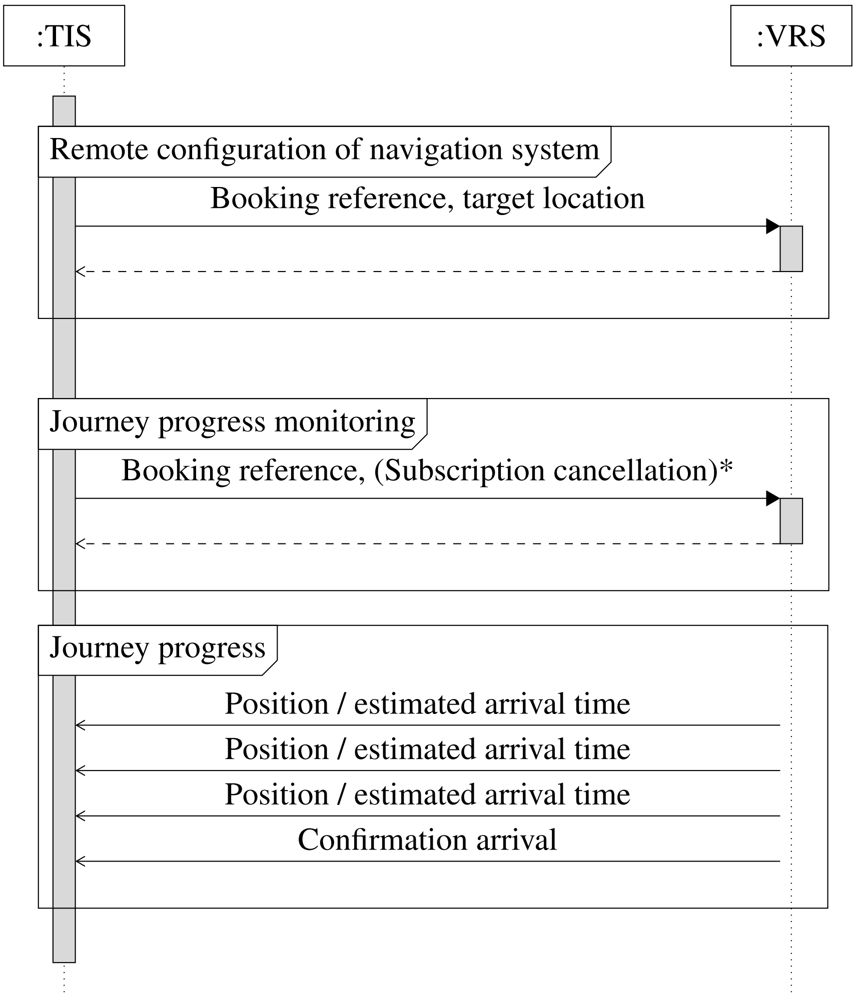
The TIS creates user accounts in the VRS to avoid multiple registrations in both services. Optionally, users can be locked/unlocked, too.

4.11 Service 10 – Vehicle Setting Management



The TIS is able to subscribe to booking or vehicle settings by the VRS. After a completed booking (or in intervals), the VRS transfers the vehicle setting, e.g., seat position, target temperature of the air condition, preferred radio stations, ... which were previously set by the user. Shortly before the beginning of a follow-up booking, the TIS is able to transfer vehicle settings, to provide the same vehicle configuration as before to the user.

4.12 Service 11 – Remote Configuration of Navigation System and Journey Progress Monitoring


:TIS
:VRS
Remote configuration of navigation system
Booking reference, target location
Journey progress monitoring
Booking reference, (Subscription cancellation)*
Journey progress
Position / estimated arrival time
Position / estimated arrival time
Position / estimated arrival time
Confirmation arrival

After the booking, the TIS can transfer the target location to the vehicle (remote configuration navigation system), so that the traveler does not need to enter the target location by himself. Furthermore, the TIS can subscribe to the progress of a journey and then gets continuously informed by the VRS concerning changes of the position and the estimated arrival time.

5 Data Model

This chapter describes the underlying data model.

Legend of Symbols:
○ Choice (`xs:choice`)
□ Optional (`minOccurs=0`)
⧈ Multivalent (`maxOccurs=0`)

5.1 Base Data Types

Simple Base Types

Simple base types are aliases for existing data types to allow a semantic differentiation.

Name	Basetype	Comment
AreaIDType	`xs:token`	Type for area IDs.
AttributeClassType	`xs:token`	Enumeration for attribute classes.
AttributeIDType	`xs:token`	Enumeration for attribute codes.
BookeeIDType	`xs:token`	Type for bookee IDs.
BookingIDType	`xs:token`	Type for booking IDs.
BookingSettingsClassType	`xs:token`	Enumeration for booking settings classes.
BookingStateType	`xs:token`	Enumeration for booking state types.
ClassType	`xs:token`	Enumeration for car classes.
ConsumptionClassType	`xs:token`	Enumeration for consumption classes.
EngineType	`xs:token`	Enumeration for engines.
ErrorCodeType	`xs:token`	Enumeration for error codes.
EuroPriceType	`xs:nonNegativeInteger`	Type for storing prices in Euro cents.
GlobalIDType	`xs:token`	Type for globally unique IDs for identifiying clone entities.
PercentType	`xs:nonNegativeInteger`	Type for percent between 0 and 100.
PlaceGroupIDType	`xs:token`	Type for place group IDs.
PlaceIDType	`xs:token`	Type for place IDs.
ProviderIDType	`xs:token`	Type for provider IDs.
SessionIDType	`xs:token`	Type for session IDs.
SystemIDType	`xs:token`	Type for system IDs.
UserFeatureClassType	`xs:token`	Enumeration for user attribute classes.
UserIDType	`xs:token`	Type for user IDs.
UserStateType	`xs:token`	Enumeration for user states.

```
<xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AreaIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="PlaceIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="PlaceGroupIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="BookeeIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="ProviderIDType">
```

```xml
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="SessionIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="BookingIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="UserIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="SystemIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="GlobalIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="EuroPriceType">
  <xs:restriction base="xs:nonNegativeInteger"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="PercentType">
  <xs:restriction base="xs:nonNegativeInteger">
    <xs:maxInclusive value="100"/>
  </xs:restriction>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="ErrorCodeType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="ClassType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="EngineType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="AttributeClassType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="ConsumptionClassType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="BookingSettingsClassType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="AttributeIDType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="UserStateType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="UserFeatureClassType">
  <xs:restriction base="xs:token"/>
</xs:simpleType><xs:simpleType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org
    ↪ /2001/XMLSchema" name="BookingStateType">
  <xs:restriction base="xs:token">
    <xs:enumeration value="open"/>
    <xs:enumeration value="suspended"/>
    <xs:enumeration value="closed"/>
  </xs:restriction>
</xs:simpleType>
```

XML source code 5.1: Baisdatentypen

Allowed values for enumerations *ClassType*, *EngineType*, *AttributeClassType* and *ErrorCodeType* are described in chapter 8.

Text

TextType: Type for a text.

	Element	Type	Comment
	Text	`xs:string`	
□	*Language*	`xs:language`	Language identifier according to BCP 47, which is based in ISO 639.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TextType">
  <xs:sequence>
    <xs:element name="Text" type="xs:string"/>
    <xs:element name="Language" type="xs:language" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.2: TextType

Geographical Coordinates

CoordType: Type for geodecimal coordinates. The reference ellipsoid is always WGS84 (GPS).

Element	Type	Comment
Longitude	`simpleType`	
Latitude	`simpleType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CoordType">
  <xs:sequence>
    <xs:element name="Longitude">
      <xs:simpleType>
        <xs:restriction base="xs:decimal">
          <xs:minInclusive value="-180.0"/>
          <xs:maxInclusive value="180.0"/>
        </xs:restriction>
```

```xml
      </xs:simpleType>
    </xs:element>
    <xs:element name="Latitude">
      <xs:simpleType>
        <xs:restriction base="xs:decimal">
          <xs:minInclusive value="-90.0"/>
          <xs:maxInclusive value="90.0"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.3: CoordType

Address

AddressType: Type for storing an address.

	Element	Type	Comment
□	*Country*	`xs:string`	Name of the country.
□	*PostalCode*	`xs:string`	Complete postal code of the address.
	City	`xs:string`	Name of the city.
□	*StreetHouseNr*	`xs:string`	Street and house number in one string.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AddressType">
  <xs:sequence>
    <xs:element name="Country" type="xs:string" minOccurs="0"/>
    <xs:element name="PostalCode" type="xs:string" minOccurs="0"/>
    <xs:element name="City" type="xs:string"/>
    <xs:element name="StreetHouseNr" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.4: AddressType

Location

GeoPositionType: Type for a geoprachical position.

	Element	Type	Comment
	Coord	`CoordType`	Geographical coordinates.
□	*Address*	`AddressType`	Optional address of the geographical position.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="GeoPositionType">
  <xs:sequence>
    <xs:element name="Coord" type="CoordType"/>
    <xs:element name="Address" type="AddressType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.5: GeoPositionType

Cycle

GeoCircleType: Type for a geographic circle.

Element	Type	Comment
Center	`CoordType`	Coordinates of the circle center.
Radius	`xs:nonNegativeInteger`	Radius in meters of the circle.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="GeoCircleType">
  <xs:sequence>
    <xs:element name="Center" type="CoordType"/>
    <xs:element name="Radius" type="xs:nonNegativeInteger"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.6: GeoCircleType

Rectangle

GeoRectangleType: Type for a geographic rectangle.

Element	Type	Comment
UpperLeft	`CoordType`	Upper left corner of the geographic rectangle.
LowerRight	`CoordType`	Lower right corner of the geographic rectangle.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="GeoRectangleType">
  <xs:sequence>
    <xs:element name="UpperLeft" type="CoordType"/>
    <xs:element name="LowerRight" type="CoordType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.7: GeoRectangleType

Area

GeoAreaType: Type for geographic areas.

	Element	Type	Comment
⊡	*PolyPoint*	`CoordType`	Geo positions of the closed border polygon of the area.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="GeoAreaType">
  <xs:sequence>
    <xs:element name="PolyPoint" type="CoordType" minOccurs="3" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.8: GeoAreaType

Area (Inclusion / Exclusion)

IncExcGeoAreaType: Type for geographic areas with exclusion possibility.
Basetype: *GeoAreaType*

	Element	Type	Comment
□	*Exclude*	`xs:boolean`	Only if set to true, this area definition defines an exclusion.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="IncExcGeoAreaType">
  <xs:complexContent>
    <xs:extension base="GeoAreaType">
      <xs:sequence>
        <xs:element name="Exclude" type="xs:boolean" default="false" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 5.9: IncExcGeoAreaType

Time Period

TimePeriodType: Type for a time period.

Element	Type	Comment
Begin	`xs:dateTime`	
End	`xs:dateTime`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TimePeriodType">
  <xs:sequence>
    <xs:element name="Begin" type="xs:dateTime"/>
    <xs:element name="End" type="xs:dateTime"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.10: TimePeriodType

Time Period (Proposal)

TimePeriodProposalType: Type for a time period proposal.
Basetype: *TimePeriodType*

	Element	Type	Comment
□	*MaxWait*	`xs:duration`	Maximal waiting time until the booking actually may begin in order to fulfil the booking time grid.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TimePeriodProposalType">
  <xs:complexContent>
    <xs:extension base="TimePeriodType">
      <xs:sequence>
        <xs:element name="MaxWait" type="xs:duration" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 5.11: TimePeriodProposalType

Origin/Target

OriginDestType: Type for an origin or a destination location.

	Element	Type	Comment
○	*PlaceID*	`PlaceIDType`	Place ID. If given, the location refers to the place with this ID. Used for place based booking targets.
○	*GeoPosition*	`CoordType`	If given, the location refers to this geo position. Used for free floating booking targets.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="OriginDestType">
  <xs:choice>
    <xs:element name="PlaceID" type="PlaceIDType"/>
    <xs:element name="GeoPosition" type="CoordType"/>
  </xs:choice>
</xs:complexType>
```

XML source code 5.12: OriginDestType

Structured Information

InfoType: Type for structured information classification with associated text.

	Element	Type	Comment
□ ⊡	*Text*	`xs:string`	Text of the informaton to be shown to the user.
	WithText	`xs:boolean`	Defines, whether this information has a text meant for the user or not.
□	*Class*	`AttributeClassType`	Class of the information, taken from the list of allowed values.
□	*Mandatory*	`xs:boolean`	Defines whether the attribute has to be shown to the user in any case.
□	*Importance*	`PercentType`	Importance of the attribute in percent.
□	*URL*	`xs:anyURI`	URL to additional information on this general attribute. If available, the whole attribute text has to be used as the marked link.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="InfoType">
  <xs:sequence>
    <xs:element name="Text" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="WithText" type="xs:boolean"/>
    <xs:element name="Class" type="AttributeClassType" minOccurs="0"/>
    <xs:element name="Mandatory" type="xs:boolean" default="false" minOccurs="0"/>
    <xs:element name="Importance" type="PercentType" minOccurs="0"/>
    <xs:element name="URL" type="xs:anyURI" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.13: InfoType

Connection to Public Transportation

StopLinkType: Type for link from a car-sharing place to a stop.
group membership: *DurationGroup*

	Element	Type	Comment
	StopID	`xs:token`	
□	*Distance*	`xs:nonNegativeInteger`	Distance for walking this link in meters. In case of areas, this is the probable distance to walk to get a vehicle.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="StopLinkType">
  <xs:sequence>
    <xs:element name="StopID" type="xs:token"/>
    <xs:group ref="DurationGroup"/>
    <xs:element name="Distance" type="xs:nonNegativeInteger" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.14: StopLinkType

5.2 Base Groups

Position or Area

PlaceOrAreaGroup: Group for information which is used for places and free floating areas.

	Element	Type	Comment
⊡	*Name*	`TextType`	Printable name(s) of the place or area.
	ProviderID	`ProviderIDType`	Provider ID, to which the place or area belongs to.
□ ⊡	*Description*	`TextType`	
□ ⊡	*StopLink*	`StopLinkType`	Walking links to stops.
□ ⊡	*AttributeID*	`AttributeIDType`	Additional information on this place or area.

```
<xs:group xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="
    ↪ PlaceOrAreaGroup">
  <xs:sequence>
    <xs:element name="Name" type="TextType" maxOccurs="unbounded"/>
    <xs:element name="ProviderID" type="ProviderIDType"/>
    <xs:element name="Description" type="TextType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="StopLink" type="StopLinkType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="AttributeID" type="AttributeIDType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:group>
```

XML source code 5.15: PlaceOrAreaGroup

Duration

DurationGroup: Group for the probable duration and its variance for getting a booking target.

	Element	Type	Comment
	Duration	`xs:duration`	Probable duration for finding a booking target.
□	*Variance*	`xs:duration`	Variance of the probable duration for finding a booking target. The probable duration can vary by this value up or down.

```
<xs:group xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="
    ↪ DurationGroup">
  <xs:sequence>
    <xs:element name="Duration" type="xs:duration"/>
    <xs:element name="Variance" type="xs:duration" minOccurs="0"/>
  </xs:sequence>
</xs:group>
```

XML source code 5.16: DurationGroup

5.3 Service 1 - Static Data

Booking Target ID

BookingTargetIDType: Type for booking target IDs.

Element	Type	Comment
BookeeID	`BookeeIDType`	
ProviderID	`ProviderIDType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingTargetIDType">
  <xs:sequence>
    <xs:element name="BookeeID" type="BookeeIDType"/>
    <xs:element name="ProviderID" type="ProviderIDType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.17: BookingTargetIDType

Rental Station ID

ProviderPlaceIDType: Type for provider/place combinations.

Element	Type	Comment
ProviderID	`ProviderIDType`	
PlaceID	`PlaceIDType`	

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="ProviderPlaceIDType">
  <xs:sequence>
    <xs:element name="ProviderID" type="ProviderIDType"/>
    <xs:element name="PlaceID" type="PlaceIDType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.18: ProviderPlaceIDType

Vehicle Attribute

AttributeType: Type for structured attribute classification with associated text.

	Element	Type	Comment
□ ⊡	*Text*	`TextType`	Text of the attribute to be shown to the user.
	WithText	`xs:boolean`	Defines, whether this attribute has a text meant for the user or not.
	ID	`AttributeIDType`	Internal code of the attribute for referencing.
□	*Class*	`AttributeClassType`	Class of the attribute, taken from the list of allowed values.
□	*Separate*	`xs:boolean`	If set to true, this attribute describes accessory equipment, which has to be booked separately.
□	*Mandatory*	`xs:boolean`	Defines whether the attribute has to be shown to the user in any case.
□	*Importance*	`PercentType`	Importance of the attribute in percent.
□	*URL*	`xs:anyURI`	URL to additional information on this general attribute. If available, the whole attribute text has to be used as the marked link.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AttributeType">
  <xs:sequence>
    <xs:element name="Text" type="TextType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="WithText" type="xs:boolean"/>
    <xs:element name="ID" type="AttributeIDType"/>
    <xs:element name="Class" type="AttributeClassType" minOccurs="0"/>
    <xs:element name="Separate" type="xs:boolean" default="false" minOccurs="0"/>
    <xs:element name="Mandatory" type="xs:boolean" default="false" minOccurs="0"/>
    <xs:element name="Importance" type="PercentType" minOccurs="0"/>
    <xs:element name="URL" type="xs:anyURI" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.19: AttributeType

Rental Station

PlaceType: Type for place information.
group membership: *PlaceOrAreaGroup*

	Element	Type	Comment
	ID	`PlaceIDType`	Place ID. For one physical place this ID has to remain constant for subsequent responses.
□	*GlobalID*	`GlobalIDType`	Globallly unique ID of this object. Used for identifying clones of this object, if they are delivered from different systems. Such ID should be formed using unique elements like DNS domain names.
	GeoPosition	`GeoPositionType`	Geo position of the place.
□	*Capacity*	`xs:nonNegativeInteger`	Capacity of the place for returning booking targets.
□	*OnPremisesTime*	`xs:duration`	Additional time needed for finding the place on the premises. Can be used for large parking areas or parking houses. Defaults to 0.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceType">
  <xs:sequence>
    <xs:element name="ID" type="PlaceIDType"/>
    <xs:element name="GlobalID" type="GlobalIDType" minOccurs="0"/>
    <xs:element name="GeoPosition" type="GeoPositionType"/>
    <xs:element name="Capacity" type="xs:nonNegativeInteger" minOccurs="0"/>
    <xs:group ref="PlaceOrAreaGroup"/>
    <xs:element name="OnPremisesTime" type="xs:duration" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.20: PlaceType

Area with Dense Indication

DensityAreaType: Type for geographic areas with probable time to get a vehicle.
group membership: *DurationGroup*

Element	Type	Comment
Area	`GeoAreaType`	Geographic area definition.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="DensityAreaType">
  <xs:sequence>
    <xs:element name="Area" type="GeoAreaType"/>
    <xs:group ref="DurationGroup"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.21: DensityAreaType

Freefloating Area

FloatingAreaType: Type for free floating areas.
group membership: *PlaceOrAreaGroup DurationGroup*

	Element	Type	Comment
	ID	`AreaIDType`	Area ID. For one physical free floating area this ID has to remain constant for subsequent responses.
⊡	*Area*	`IncExcGeoAreaType`	Geographic areas covered by the free floaters. A point belongs to the free floating area, if it is within at least one of the given inclusion areas, and not in one of the exclusion areas.
□ ⊡	*SubArea*	`DensityAreaType`	Sub areas of the free floating area. For each sub area a probable time (and its variance) for getting a free floating booking target is given. For stops covered by more than one sub area, the minimum duration is used.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="FloatingAreaType">
  <xs:sequence>
    <xs:element name="ID" type="AreaIDType"/>
    <xs:element name="Area" type="IncExcGeoAreaType" maxOccurs="unbounded"/>
    <xs:group ref="PlaceOrAreaGroup"/>
    <xs:group ref="DurationGroup"/>
    <xs:element name="SubArea" type="DensityAreaType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.22: FloatingAreaType

Group of Rental Stations

PlaceGroupType: Type for place groups.

	Element	Type	Comment
	ID	`PlaceGroupIDType`	Place group ID. For one physical place group this ID has to remain constant for subsequent responses.
⊡	*PlaceID*	`ProbabilityPlaceIDType`	
	Probability	`PercentType`	Average probability for getting a booking target at a specific place in this place group. It only applies for those places, for which no specific value is provided.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceGroupType">
  <xs:sequence>
    <xs:element name="ID" type="PlaceGroupIDType"/>
    <xs:element name="PlaceID" type="ProbabilityPlaceIDType" maxOccurs="unbounded"/>
    <xs:element name="Probability" type="PercentType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.23: PlaceGroupType

Provider

ProviderType: Type for provider information.

	Element	Type	Comment
	ID	`ProviderIDType`	Provider ID. For one provider this ID has to remain constant for subsequent responses.
	Name	`xs:normalizedString`	Printable full name of the provider.
	CustomerChoice	`xs:boolean`	If set, this provider is presented to the user as a "home provider". This way this provider becomes a "customer provider".
□	*ShortName*	`xs:normalizedString`	Printable short name of the provider.
□ ⊡	*AttributeID*	`AttributeIDType`	Additional information on this provider (e.g. URLs, logo, inter-app-URLs).

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ProviderType">
  <xs:sequence>
    <xs:element name="ID" type="ProviderIDType"/>
    <xs:element name="Name" type="xs:normalizedString"/>
    <xs:element name="CustomerChoice" type="xs:boolean"/>
    <xs:element name="ShortName" type="xs:normalizedString" minOccurs="0"/>
    <xs:element name="AttributeID" type="AttributeIDType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.24: ProviderType

Booking Target

BookingTargetType: Type for booking target information.

	Element	Type	Comment
	ID	`BookingTargetIDType`	BookingTargetID. For one physical booking target this ID has to remain constant for subsequent responses.
□	*GlobalID*	`GlobalIDType`	Globallly unique ID of this object. Used for identifying clones of this object, if they are delivered from different systems. Such ID should be formed using unique elements like DNS domain names.
⊡	*Name*	`TextType`	Printable name of the bookee.
○	*PlaceID*	`PlaceIDType`	Place ID, where the bookee is placed. This has to be filled for station based booking targets.
○	*PlaceGroupID*	`PlaceGroupIDType`	PLace group ID of this bookee, if it does not belong to a fixed place.
○	*AreaID*	`AreaIDType`	Area ID of this bookee. This has to be filled for free floating booking targets.
	Class	`ClassType`	Class of the bookee.
□	*BookingHorizon*	`xs:duration`	Maximum duration in future, when this target can be booked. This is relative to the current time.
□	*BookingGrid*	`xs:nonNegativeInteger`	Grid for start and end time of bookings in minutes. Should be a divisor of 60. Examples: 1, 5, 10, 15, 30, 60.
□	*OpeningTime*	`xs:duration`	Time needed for actually opening the booking target. If not given, the client system will assume some reasonable time.
□	*Engine*	`EngineType`	
□	*CO2Factor*	`xs:nonNegativeInteger`	CO2 emmision in gram per kilometer.
□	*MaxDistance*	`xs:nonNegativeInteger`	Maximal distance in meters, this bookee can go without longer stops. If not given, no limit is assumed.
□ ⊡	*AttributeID*	`AttributeIDType`	Additional information on this bookee, e.g. air-conditioning, navigational system, etc.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="BookingTargetType">
  <xs:sequence>
    <xs:element name="ID" type="BookingTargetIDType"/>
    <xs:element name="GlobalID" type="GlobalIDType" minOccurs="0"/>
    <xs:element name="Name" type="TextType" maxOccurs="unbounded"/>
    <xs:choice>
      <xs:element name="PlaceID" type="PlaceIDType"/>
      <xs:element name="PlaceGroupID" type="PlaceGroupIDType"/>
      <xs:element name="AreaID" type="AreaIDType"/>
    </xs:choice>
    <xs:element name="Class" type="ClassType"/>
    <xs:element name="BookingHorizon" type="xs:duration" minOccurs="0"/>
    <xs:element name="BookingGrid" type="xs:nonNegativeInteger" minOccurs="0"/>
    <xs:element name="OpeningTime" type="xs:duration" minOccurs="0"/>
    <xs:element name="Engine" type="EngineType" minOccurs="0"/>
    <xs:element name="CO2Factor" type="xs:nonNegativeInteger" minOccurs="0"/>
    <xs:element name="MaxDistance" type="xs:nonNegativeInteger" minOccurs="0"/>
    <xs:element name="AttributeID" type="AttributeIDType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.25: BookingTargetType

Booking

BookingType: Type for a booking.

	Element	Type	Comment
	ID	`BookingIDType`	
□	*TimePeriod*	`TimePeriodType`	Actual time period of the booking, eventually adapted to the booking time grid.
□ ⊡	*Info*	`InfoType`	Additional information on this booking.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="BookingType">
  <xs:sequence>
    <xs:element name="ID" type="BookingIDType"/>
    <xs:element name="TimePeriod" type="TimePeriodType" minOccurs="0"/>
    <xs:element name="Info" type="InfoType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.26: BookingType

5.4 Service 2 - Availability Information

Booking Characteristics

BookingTargetPropertiesType: Type for needed properties of a booking target for synchronous availability.

	Element	Type	Comment
	ID	`BookingTargetIDType`	Booking target ID.
□	*Distance*	`xs:nonNegativeInteger`	Distance to drive in meters. The car-sharing system should return only those booking targets as available, which are able to drive that distance (according to state of charge).
□	*SeverityFactor*	`xs:nonNegativeInteger`	Severity factor of the proposed route in percentage. 100 represents a nearly flat route. The higher the value, the more energy is cosumed for driving the route. E.g. a value of 200 indecates, that twice as much energy will be consumed as would be for a nearly flat route of the same length.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="BookingTargetPropertiesType">
  <xs:sequence>
    <xs:element name="ID" type="BookingTargetIDType"/>
    <xs:element name="Distance" type="xs:nonNegativeInteger" minOccurs="0"/>
    <xs:element name="SeverityFactor" type="xs:nonNegativeInteger" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.27: BookingTargetPropertiesType

Booking Target Availability

BookingTargetAvailabilityType: Type for availability of booking targets.

	Element	Type	Comment
	ID	`BookingTargetIDType`	
□	*PlaceID*	`PlaceIDType`	For currently available booking targets, which are not bound to a specific place, the current place should be returned.
□	*GeoPosition*	`GeoPositionType`	For currently available booking targets, which are not place bounded, the current position should be returned.
□ ⊡	*Inavailability*	`TimePeriodType`	Periods of inavailability of the booking target.
□	*CurrentStateOfCharge*	`PercentType`	Current state of charge iof booking target in percent.
□	*CurrentDrivingRange*	`xs:nonNegativeInteger`	Current driving range of booking target in meters, given a nearly flat route. If not given, the vehice is considered to have an infinite driving range.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="BookingTargetAvailabilityType">
  <xs:sequence>
    <xs:element name="ID" type="BookingTargetIDType"/>
    <xs:element name="PlaceID" type="PlaceIDType" minOccurs="0"/>
    <xs:element name="GeoPosition" type="GeoPositionType" minOccurs="0"/>
    <xs:element name="Inavailability" type="TimePeriodType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="CurrentStateOfCharge" type="PercentType" minOccurs="0"/>
    <xs:element name="CurrentDrivingRange" type="xs:nonNegativeInteger" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.28: BookingTargetAvailabilityType

5.5 Service 4 - Booking

Vehicle Availability

BookingTargetChangeAvailabilityType: Type for changes in availability of booking targets.

	Element	Type	Comment
	ID	`BookingTargetIDType`	Booking target ID.
□	*PlaceID*	`PlaceIDType`	For currently available booking targets, which are not bound to a specific place, the current place should be returned.
□	*GeoPosition*	`GeoPositionType`	For newly available vehicles, which are not place bounded, the position should be returned.
○	*Inavailability*	`TimePeriodType`	New inavailability period
○	*Availability*	`TimePeriodType`	New availability period.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingTargetChangeAvailabilityType">
  <xs:sequence>
    <xs:element name="ID" type="BookingTargetIDType"/>
    <xs:element name="PlaceID" type="PlaceIDType" minOccurs="0"/>
    <xs:element name="GeoPosition" type="GeoPositionType" minOccurs="0"/>
    <xs:choice>
      <xs:element name="Inavailability" type="TimePeriodType"/>
      <xs:element name="Availability" type="TimePeriodType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.29: BookingTargetChangeAvailabilityType

Location Capacity

PlaceAvailabilityType: Type for the available capacity of a place.

Element	Type	Comment
ID	`ProviderPlaceIDType`	
Availability	`xs:nonNegativeInteger`	Currently available capacity for returning booking targets. This is the number of booking targets which could be returned now, without exceeding the overall capacity of the place.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceAvailabilityType">
  <xs:sequence>
    <xs:element name="ID" type="ProviderPlaceIDType"/>
    <xs:element name="Availability" type="xs:nonNegativeInteger"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.30: PlaceAvailabilityType

Service 4a - External Bookings

ExternalBookingType: Type for information for external bookings.

Element	Type	Comment
BookingID	`BookingIDType`	BookingID of the externally added booking.
BookingTargetID	`BookingTargetIDType`	BookingTargetID of the booked vehicle.
UserInfo	`UserInfoType`	UserInfo of the user this booking belongs to.
TimePeriod	`TimePeriodType`	Time period of respective booking.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ExternalBookingType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:element name="BookingTargetID" type="BookingTargetIDType"/>
    <xs:element name="UserInfo" type="UserInfoType"/>
    <xs:element name="TimePeriod" type="TimePeriodType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.31: ExternalBookingType

5.6 Service 5 - Booking Subscription

Booking Availability

BookingChangeType: Type for changes of a booking.

	Element	Type	Comment
	BookingID	`BookingIDType`	
○	*Cancelled*	`xs:boolean`	Booking has been cancelled by user, eventually through different channels (e.g. native app, phone, etc). Afterwards the booking ID is not valid anymore.
○	*NewPeriod*	`TimePeriodType`	Booking has been given a new period by user. eventually through different channels (e.g. antive app, phone, etc.).
○	*Notification*	`xs:boolean`	The booking is still valid, but there is some infromation for the user concerning the booking.
○	*Impossible*	`xs:boolean`	The booking has been rendered impossible by the car sharing system (e.g. due to a car failure). However, the booking ID is still valid, as the booking might be re-possiblized later.
○	*Repossiblized*	`xs:boolean`	The booking has been rendered possible again (e.g. by a replacement car).
□ ⊡	*Reason*	`TextType`	

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingChangeType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:choice>
      <xs:element name="Cancelled" type="xs:boolean"/>
      <xs:element name="NewPeriod" type="TimePeriodType"/>
      <xs:element name="Notification" type="xs:boolean"/>
      <xs:element name="Impossible" type="xs:boolean"/>
      <xs:element name="Repossiblized" type="xs:boolean"/>
    </xs:choice>
    <xs:element name="Reason" type="TextType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.32: BookingChangeType

5.7 Service 6 - Price Information

Price Information (Details)

TariffDetailType: Type for a component of a traiff including the price.

	Element	Type	Comment
	Name	`xs:string`	Printable name of the tariff detail.
	Price	`EuroPriceType`	Price in Euro-Cent.
□	*Description*	`xs:string`	Printable description of the tariff detail.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TariffDetailType">
  <xs:sequence>
    <xs:element name="Name" type="xs:string"/>
    <xs:element name="Price" type="EuroPriceType"/>
    <xs:element name="Description" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.33: TariffDetailType

Price Information

TariffType: Type for a tariff including the price.

	Element	Type	Comment
	Name	`xs:string`	Printable name of the tariff.
	Price	`EuroPriceType`	Price in Euro-Cent.
□	*Description*	`xs:string`	Printable description of the tariff.
□ ⊡	*Detail*	`TariffDetailType`	Details for the overall price. If provided, it should contain a set of component prices, that sums up to the overall price.
□ ⊡	*Info*	`InfoType`	Additional information for this tariff.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TariffType">
  <xs:sequence>
    <xs:element name="Name" type="xs:string"/>
    <xs:element name="Price" type="EuroPriceType"/>
    <xs:element name="Description" type="xs:string" minOccurs="0"/>
    <xs:element name="Detail" type="TariffDetailType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Info" type="InfoType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.34: TariffType

5.8 Service 7 - Consumption Data Subscription

Consumption Data

ConsumptionType: Type for single Consumption item.

	Element	Type	Comment
	BookingID	`BookingIDType`	BookingID reference for this Consumption item.
	Type	`ConsumptionClassType`	Type of consumption.
⊡	*Description*	`TextType`	Printable name of consumption item.
○	*Value*	`xs:decimal`	Decimal consumption value (e.g., distance).
○	*TimePeriod*	`TimePeriodType`	Consumption time period.
□	*Unit*	`xs:string`	Printable name of Unit.
□	*Price*	`EuroPriceType`	Price in Euro-Cent.
□ ⊡	*Annotation*	`TextType`	Printable annotation for the consumption items, e.g. further information.
□	*Final*	`xs:boolean`	Consider consumption data for this booking final. No further consumption data is expected.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ConsumptionType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:element name="Type" type="ConsumptionClassType"/>
    <xs:element name="Description" type="TextType" maxOccurs="unbounded"/>
    <xs:choice>
      <xs:element name="Value" type="xs:decimal"/>
      <xs:element name="TimePeriod" type="TimePeriodType"/>
    </xs:choice>
    <xs:element name="Unit" type="xs:string" minOccurs="0"/>
    <xs:element name="Price" type="EuroPriceType" minOccurs="0"/>
    <xs:element name="Annotation" type="TextType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Final" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.35: ConsumptionType

5.9 Service 8 - Booking / Vehicle Unlock

For Service 8 no additional data types are required.

5.10 Service 9 - User Management

User Features

UserFeatureType: Type for structured user attribute classification with associated text.

Element	Type	Comment
Value	`xs:string`	value of the attribute to be shown to the user.
Class	`UserFeatureClassType`	Class of the user attribute, taken from the list of allowed values.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="UserFeatureType">
  <xs:sequence>
    <xs:element name="Value" type="xs:string"/>
    <xs:element name="Class" type="UserFeatureClassType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.36: UserFeatureType

User Type

UserType: Type for a user (used in user management).

	Element	Type	Comment
	ID	`UserInfoType`	UserID and ProviderID for identification, can also contain credentials.
	State	`UserStateType`	State of the (e.g., default, banned etc).
□ ⊡	*Features*	`UserFeatureType`	Additional information on user.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="UserType">
  <xs:sequence>
    <xs:element name="ID" type="UserInfoType"/>
    <xs:element name="State" type="UserStateType"/>
    <xs:element name="Features" type="UserFeatureType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.37: UserType

5.11 Service 10 - Booking / Vehicle Settings Management

Booking / Vehicle Settings

BookingSettingsType: Type for single booking settings item.

	Element	Type	Comment
	BookingID	`BookingIDType`	BookingID reference for this setting item.
□	*BookingTargetSpecific*	`xs:boolean`	Falls gesetzt, ist die Einstellung nur für das gleiche Buchungsziel anwendbar (fahrzeugspezifisch).
	Type	`BookingSettingsClassType`	Type of booking setting.
○	*Value*	`xs:decimal`	Decimal settings value (e.g., temperature in C).
○	*Value*	`xs:string`	Arbitrary string settings.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingSettingsType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:element name="BookingTargetSpecific" type="xs:boolean" default="false" minOccurs="0"/>
    <xs:element name="Type" type="BookingSettingsClassType"/>
    <xs:choice>
      <xs:element name="Value" type="xs:decimal"/>
      <xs:element name="Value" type="xs:string"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.38: BookingSettingsType

5.12 Service 11 - Remote Configuration Navigation System and Journey Progress Monitoring

Route Progress

BookingProgressType: Type for single booking progress item.

	Element	Type	Comment
	BookingID	`BookingIDType`	BookingID reference for this progress item.
	Timestamp	`xs:dateTime`	Timestamp of the geo localization / ETA calculation.
	ETA	`xs:dateTime`	Estimated time of arrival of the booking target at the destination.
□	*Position*	`CoordType`	Current position of the booking target. Optional for possible privacy concerns.
□	*Progress*	`PercentType`	Progress estimator for visual representation.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingProgressType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:element name="Timestamp" type="xs:dateTime"/>
    <xs:element name="ETA" type="xs:dateTime"/>
    <xs:element name="Position" type="CoordType" minOccurs="0"/>
    <xs:element name="Progress" type="PercentType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.39: BookingProgressType

5.13 Authentication

User Identification

UserInfoType: Type for a user identification. Password and Token are optional in case UserInfoType is used as identification.

	Element	Type	Comment
	ProviderID	`ProviderIDType`	ID of the provider of the user. Without the user ID is not unique.
	UserID	`UserIDType`	User ID.
○	*Password*	`xs:normalizedString`	Password of the user.
○	*Token*	`xs:token`	Authentication token for the user, which replaces the password for data security reasons.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="UserInfoType">
  <xs:sequence>
    <xs:element name="ProviderID" type="ProviderIDType"/>
    <xs:element name="UserID" type="UserIDType"/>
    <xs:choice minOccurs="0">
      <xs:element name="Password" type="xs:normalizedString"/>
      <xs:element name="Token" type="xs:token"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.40: UserInfoType

Authentication

AuthType: Type for an authentication within a request.

	Element	Type	Comment
○	*SessionID*	`SessionIDType`	Valid session ID. If a session is reused, fill this one.
⊡	*UserInfo*	`UserInfoType`	If no session is available, the user has to authenticate.
○	*Anonymous*	`xs:boolean`	If neither an existing session, nor a use information is available, an anonyous session can be opened.
□	*SessionTimeout*	`xs:duration`	If set, the client wishes to get a session with the given inactivity timeout.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AuthType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="SessionID" type="SessionIDType"/>
      <xs:sequence>
        <xs:element name="UserInfo" type="UserInfoType" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:element name="Anonymous" type="xs:boolean"/>
    </xs:choice>
    <xs:element name="SessionTimeout" type="xs:duration" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.41: AuthType

5.14 Error Handling

Errors

ErrorType: Type for errors.

	Element	Type	Comment
	Code	`ErrorCodeType`	Code of the error taken from the list of allowed error codes.
□	*NonFatal*	`xs:boolean`	If set, the corresponding operation was in general successful and operations can continue.
□	*SystemMessage*	`xs:string`	Text for system logging, explaining the error more detailed. Should be in English.
□	*UserMessage*	`xs:string`	Text for user information in requested language.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ErrorType">
  <xs:sequence>
    <xs:element name="Code" type="ErrorCodeType"/>
    <xs:element name="NonFatal" type="xs:boolean" default="false" minOccurs="0"/>
    <xs:element name="SystemMessage" type="xs:string" minOccurs="0"/>
    <xs:element name="UserMessage" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 5.42: ErrorType

6 Technical Realization

6.1 Message Encoding

The messages are transferred as XML-files between both systems. A precise type-definition is set by the XML-scheme, corresponding to the interface.

In case the overhead, resulting of the introduction of XML turns out to be problematic, there is the possibility to use the Efficient XML Interchange (EXI) protocol. The use of EXI would reduce the size of messages significantly, without losing the advantages of XML.

6.2 Communication channel

As the interface allows asynchronous subscription besides the usual request/response scheme, usage of the WebSocket protocol instead of plain HTTP is recommended. WebSockets allow persistent connections between both systems and a bidirectional message exchange. Since the interface also provides an asynchronous subscription model, besides the request and response scheme and a low response rate is desirable, the WebSocket-protocol is recommended for the communication channel. WebSockets allow to establish a connection between two systems and exchange messages bidirectional. The VRS represents the server and the TIS the client (in HTTP context). In principle, any amount of communication channels can be opened. Updates of subscribed objects are delivered via the same connection as the one they are subscribed at. In case of an interrupt of the connection, the subscription ends.

6.3 Authentication

IXSI is a B2B interface and therefore does not include an internal authentication mechanism, instead the communication partners are advised to use existing authentication mechanisms such as SSL certificates, a virtual private network (VPN) or HTTP authentication.

End-customer

Since IXSI also offers user-controlled requests that are transferred from system to system, it is necessary that a user can authenticate himself towards the VRS. This usually happens via the triplet provider reference/user reference/password. To avoid typing the password in plaintext, alternatively to the password, a token can be used. This token can be generated by the VRS with the help of the function TokenGeneration (vgl. section 7.5). Thus, the user initially has to type his password once, afterwards he is able to authenticate himself via the generated token. For example, this can be saved on the user's device.

In case of various consecutive requests which are invoked by the same user, an authentication should not be executed for each one. Therefore, with a user's first request, a session will be opened (explicitly via OpenSession or implicitly). For following requests of the same user, the ID of the opened session instead of an authentication can be transferred. These sessions have a time limited validity. In case this validity expires, the user needs to be identified again and a new session needs to be opened. In addition, a session can be closed explicitly via CloseSession.

6.4 Connection Security

To ensure confidentiality of the transferred data, encryption is required. Therefore, the SSL/TLS-protocol is suitable. This should be used in case that the connection is not already secured through other measures (e.g., by using VPN).

7 Messages

The messages that are transferred between the interaction partners are based upon five message types in form of request / response and push messages. All messages going from TIS to VRS represent requests. The type response is used for direct responses of the VRS of requests, coming from the TIS. To allow an allocation of response-messages to the corresponding request-messages, every request-message is marked by a unique transaction-ID, which has to be delivered within the corresponding response-message. Instead of its' usual content, a response-message can also contain errors. The last message type push is used in cases, where the TIS demanded for ongoing updates (subscriptions) towards the VRS.

For some messages, examples for illustration purposes are available, see listing 7.24.

7.1 Base Messages

Base Class

IxsiMessageType: Type for all IXSI messages.

	Element	Type	Comment
○ ▣	*Request*	`QueryRequestType`	Independent requests in query scheme.
○ ▣	*Response*	`QueryResponseType`	Independent responses in query scheme.
○	*SubscriptionRequest*	`SubscriptionRequestType`	Request in subscription scheme. Each request leads to at least one response.
○	*SubscriptionResponse*	`SubscriptionResponseType`	Response in subscription scheme. Each response corresponds to exactly one request.
○	*SubscriptionMessage*	`SubscriptionMessageType`	Push message in subscription scheme.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="IxsiMessageType">
  <xs:choice>
    <xs:element name="Request" type="QueryRequestType" maxOccurs="unbounded"/>
    <xs:element name="Response" type="QueryResponseType" maxOccurs="unbounded"/>
    <xs:element name="SubscriptionRequest" type="SubscriptionRequestType"/>
    <xs:element name="SubscriptionResponse" type="SubscriptionResponseType"/>
    <xs:element name="SubscriptionMessage" type="SubscriptionMessageType"/>
  </xs:choice>
</xs:complexType>
```

XML source code 7.1: IxsiMessageType

Base Response

AbstractBaseResponseType: Abstract type as a base for responses.

	Element	Type	Comment
□ ⊡	*Error*	`ErrorType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AbstractBaseResponseType" abstract="true">
  <xs:sequence>
    <xs:element name="Error" type="ErrorType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.2: AbstractBaseResponseType

A abstract base class for a request does not exist, because it is not necessary.

Transaction Data

TransactionType: Type for an transaction ID within requests and responses.

Element	Type	Comment
TimeStamp	`xs:dateTime`	
MessageID	`xs:nonNegativeInteger`	Unique ID.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TransactionType">
  <xs:sequence>
    <xs:element name="TimeStamp" type="xs:dateTime"/>
    <xs:element name="MessageID" type="xs:nonNegativeInteger"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.3: TransactionType

Query Request

QueryRequestType: Request messages in query scheme.
group membership: *StaticDataRequestGroup UserTriggeredRequestGroup*

Element	Type	Comment
Transaction	`TransactionType`	Transaction identifier, set by the requestor.
SystemID	`SystemIDType`	ID of the requesting system.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="QueryRequestType">
  <xs:sequence>
    <xs:element name="Transaction" type="TransactionType"/>
    <xs:element name="SystemID" type="SystemIDType"/>
    <xs:choice>
      <xs:group ref="StaticDataRequestGroup"/>
      <xs:group ref="UserTriggeredRequestGroup"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.4: QueryRequestType

Query Response

QueryResponseType: Response messages in query scheme.
group membership: *StaticDataResponseGroup UserTriggeredResponseGroup*

Element	Type	Comment
Transaction	`TransactionType`	Transaction identifier, copied from the corresponding request.
CalcTime	`xs:duration`	Real time needed for calculating the contained results.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="QueryResponseType">
  <xs:sequence>
    <xs:element name="Transaction" type="TransactionType"/>
    <xs:element name="CalcTime" type="xs:duration"/>
    <xs:choice>
      <xs:group ref="StaticDataResponseGroup"/>
      <xs:group ref="UserTriggeredResponseGroup"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.5: QueryResponseType

Subscription Request

SubscriptionRequestType: Request messages in subscription scheme.
group membership: *SubscriptionAdministrationRequestGroup SubscriptionRequestGroup RequestMessageGroup*

Element	Type	Comment
Transaction	`TransactionType`	Transaction identifier, set by the requestor.
SystemID	`SystemIDType`	ID of the requesting system.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="SubscriptionRequestType">
  <xs:sequence>
    <xs:element name="Transaction" type="TransactionType"/>
    <xs:element name="SystemID" type="SystemIDType"/>
    <xs:choice>
      <xs:group ref="SubscriptionAdministrationRequestGroup"/>
      <xs:group ref="SubscriptionRequestGroup"/>
      <xs:group ref="RequestMessageGroup"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.6: SubscriptionRequestType

Subscription Response

SubscriptionResponseType: Response messages in subscription scheme.
group membership: *SubscriptionAdministrationResponseGroup SubscriptionResponseGroup ResponseMessageGroup*

Element	Type	Comment
Transaction	`TransactionType`	Transaction identifier, copied from the corresponding request.
CalcTime	`xs:duration`	Real time needed for calculating the contained results.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="SubscriptionResponseType">
  <xs:sequence>
    <xs:element name="Transaction" type="TransactionType"/>
    <xs:element name="CalcTime" type="xs:duration"/>
    <xs:choice>
      <xs:group ref="SubscriptionAdministrationResponseGroup"/>
      <xs:group ref="SubscriptionResponseGroup"/>
      <xs:group ref="ResponseMessageGroup"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.7: SubscriptionResponseType

7.2 Base Groups

User-Initiated

UserTriggeredRequestGroup: Requests which are directly triggered through a user interaction.

	Element	Type	Comment
□	*Language*	`xs:language`	Requested language for the results. All texts returned in the respective responses should be in this language.
	Auth	`AuthType`	Authentication information.
○	*OpenSession*	`OpenSessionRequestType`	Open a session for a user.
○	*CloseSession*	`CloseSessionRequestType`	Explicitely close a session.
○	*TokenGeneration*	`TokenGenerationRequestType`	Generate a token for a user which replaces his password.
○	*Availability*	`AvailabilityRequestType`	Get the availability for the given booking targets.
○	*PlaceAvailability*	`PlaceAvailabilityRequestType`	Get the availabile capacity for the given places.
○	*PriceInformation*	`PriceInformationRequestType`	Get a price information.
○	*Booking*	`BookingRequestType`	Book a booking target.
○	*ChangeBooking*	`ChangeBookingRequestType`	Change a booking.
○	*ChangeBookingState*	`ChangeBookingStateRequestType`	Change the state of the booked vehicle.
○	*CreateUser*	`CreateUserRequestType`	Create a user.
○	*ChangeUser*	`ChangeUserRequestType`	Change a

```
<xs:group xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="
    ↪ UserTriggeredRequestGroup">
  <xs:sequence>
    <xs:element name="Language" type="xs:language" minOccurs="0"/>
    <xs:element name="Auth" type="AuthType"/>
    <xs:choice>
      <xs:element name="OpenSession" type="OpenSessionRequestType"/>
      <xs:element name="CloseSession" type="CloseSessionRequestType"/>
      <xs:element name="TokenGeneration" type="TokenGenerationRequestType"/>
      <xs:element name="Availability" type="AvailabilityRequestType"/>
      <xs:element name="PlaceAvailability" type="PlaceAvailabilityRequestType"/>
      <xs:element name="PriceInformation" type="PriceInformationRequestType"/>
      <xs:element name="Booking" type="BookingRequestType"/>
      <xs:element name="ChangeBooking" type="ChangeBookingRequestType"/>
      <xs:element name="ChangeBookingState" type="ChangeBookingStateRequestType"/>
      <xs:element name="CreateUser" type="CreateUserRequestType"/>
      <xs:element name="ChangeUser" type="ChangeUserRequestType"/>
      <xs:element name="SetBookingSettings" type="SetBookingSettingsRequestType"/>
      <xs:element name="SetNavigationDestination" type="SetNavigationDestinationRequestType"/>
    </xs:choice>
  </xs:sequence>
</xs:group>
```

XML source code 7.8: UserTriggeredRequestGroup

UserTriggeredResponseGroup: Responses, which are directly triggered through a user interaction.

	Element	Type	Comment
□	*SessionID*	`SessionIDType`	If returned, a session for the user was opened. This session ID can be reused for authentication in the next request.
□	*SessionTimeout*	`xs:duration`	Inactivity timeout of the session. If a period of inactivity on this session exceeds this value, the session timeouts.
○	*OpenSession*	`OpenSessionResponseType`	Open a session for the user.
○	*CloseSession*	`CloseSessionResponseType`	Explicitely close a session.
○	*TokenGeneration*	`TokenGenerationResponseType`	Generate a token for a user which replaces his password.
○	*Availability*	`AvailabilityResponseType`	Get the availability for the given booking targets.
○	*PlaceAvailability*	`PlaceAvailabilityResponseType`	Get the availabile capacity for the given

```
<xs:group xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="
    ↪ UserTriggeredResponseGroup">
  <xs:sequence>
    <xs:element name="SessionID" type="SessionIDType" minOccurs="0"/>
    <xs:element name="SessionTimeout" type="xs:duration" minOccurs="0"/>
    <xs:choice>
      <xs:element name="OpenSession" type="OpenSessionResponseType"/>
      <xs:element name="CloseSession" type="CloseSessionResponseType"/>
      <xs:element name="TokenGeneration" type="TokenGenerationResponseType"/>
      <xs:element name="Availability" type="AvailabilityResponseType"/>
      <xs:element name="PlaceAvailability" type="PlaceAvailabilityResponseType"/>
      <xs:element name="PriceInformation" type="PriceInformationResponseType"/>
      <xs:element name="Booking" type="BookingResponseType"/>
      <xs:element name="ChangeBooking" type="ChangeBookingResponseType"/>
      <xs:element name="ChangeBookingState" type="ChangeBookingStateResponseType"/>
      <xs:element name="CreateUser" type="CreateUserResponseType"/>
      <xs:element name="ChangeUser" type="ChangeUserResponseType"/>
      <xs:element name="SetBookingSettings" type="SetBookingSettingsResponseType"/>
      <xs:element name="SetNavigationDestination" type="SetNavigationDestinationResponseType"/>
    </xs:choice>
  </xs:sequence>
</xs:group>
```

XML source code 7.9: UserTriggeredResponseGroup

Origin/Target Specification

OriginDestGroup: Group for origin and destination location.

	Element	Type	Comment
□	*Origin*	`OriginDestType`	Location where the booking target is to be picked up.
□	*Dest*	`OriginDestType`	Location where the booking target is to be returned.

```
<xs:group xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="
    ↪ OriginDestGroup">
  <xs:sequence>
    <xs:element name="Origin" type="OriginDestType" minOccurs="0"/>
    <xs:element name="Dest" type="OriginDestType" minOccurs="0"/>
  </xs:sequence>
</xs:group>
```

XML source code 7.10: OriginDestGroup

Groups solely for Organisation Purposes

Remark: In the XML scheme are further groups defined, which are solely used for organisational purposes, have no impact on the message format and therefore are not listed here.

Name	Comment
StaticDataRequestGroup	
StaticDataResponseGroup	
SubscriptionAdministrationRequestGroup	
SubscriptionAdministrationResponseGroup	
SubscriptionRequestGroup	
SubscriptionResponseGroup	
RequestMessageGroup	
ResponseMessageGroup	
PushMessageGroup	

7.3 Service A - Sessions

Open Session

OpenSessionRequestType: Request for opening a session for a user. Can be used for Function 2, 4, and 6
group membership: *UserTriggeredRequestGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="OpenSessionRequestType"/>
```

XML source code 7.11: OpenSessionRequestType

OpenSessionResponseType: Response for for opening a session for a user.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="OpenSessionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.12: OpenSessionResponseType

Close Session

CloseSessionRequestType: Request for closing an existing session.
group membership: *UserTriggeredRequestGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CloseSessionRequestType"/>
```

XML source code 7.13: CloseSessionRequestType

CloseSessionResponseType: Response for clsoing an existing session.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CloseSessionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.14: CloseSessionResponseType

7.4 Service B - Subscriptions

Heartbeat

HeartBeatRequestType: Request for checking a connection.
group membership: *SubscriptionAdministrationRequestGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="HeartBeatRequestType"/>
```

XML source code 7.15: HeartBeatRequestType

HeartBeatResponseType: Response for checking a connection.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionAdministrationResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="HeartBeatResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.16: HeartBeatResponseType

7.5 Service C - Tokens

Token Generation

TokenGenerationRequestType: Request for generating a token for a user.
group membership: *UserTriggeredRequestGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TokenGenerationRequestType"/>
```

XML source code 7.17: TokenGenerationRequestType

TokenGenerationResponseType: Response for generating a token for a user.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□	*Token*	`xs:token`	Token for the user. Can be used in future instead of the password for authentication.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="TokenGenerationResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Token" type="xs:token" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.18: TokenGenerationResponseType

7.6 Service 1 - Statistical Data

Request Booking Targets

BookingTargetsInfoRequestType: Request for getting all infos for all booking targets.
group membership: *StaticDataRequestGroup*

	Element	Type	Comment
□ ⊡	*ProviderFilter*	`ProviderIDType`	If set, the filter contains those providers, data is requested for. If not set, data for all providers is requested.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingTargetsInfoRequestType">
  <xs:sequence>
    <xs:element name="ProviderFilter" type="ProviderIDType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.19: BookingTargetsInfoRequestType

BookingTargetsInfoResponseType: Response for getting all infos for all booking targets.
Basetype: *AbstractBaseResponseType*
group membership: *StaticDataResponseGroup*

	Element	Type	Comment
□	*Timestamp*	`xs:dateTime`	Timestamp of this data delivery. The delivering system should be able to identify changes compared to this delivery by this timestamp.
□ ▣	*Bookee*	`BookingTargetType`	
□ ▣	*Place*	`PlaceType`	
□ ▣	*PlaceGroup*	`PlaceGroupType`	
□ ▣	*FreeFloatingArea*	`FloatingAreaType`	
□ ▣	*Provider*	`ProviderType`	
□ ▣	*Attributes*	`AttributeType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingTargetsInfoResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="Bookee" type="BookingTargetType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="Place" type="PlaceType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="PlaceGroup" type="PlaceGroupType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="FreeFloatingArea" type="FloatingAreaType" minOccurs="0" maxOccurs="unbounded
            ↪ "/>
        <xs:element name="Provider" type="ProviderType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="Attributes" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.20: BookingTargetsInfoResponseType

Request Change Booking Targets

ChangedProvidersRequestType: Request for getting those providers with changed static data.
group membership: *StaticDataRequestGroup*

Element	Type	Comment
Timestamp	`xs:dateTime`	Timestamp of the static data delivery, the changes are requested for.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangedProvidersRequestType">
  <xs:sequence>
    <xs:element name="Timestamp" type="xs:dateTime"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.21: ChangedProvidersRequestType

ChangedProvidersResponseType: Repsonse for getting those providers with changed static data.
Basetype: *AbstractBaseResponseType*
group membership: *StaticDataResponseGroup*

	Element	Type	Comment
□ ⊡	*Provider*	`ProviderIDType`	List of providers, which have changes in their static data.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangedProvidersResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Provider" type="ProviderIDType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.22: ChangedProvidersResponseType

7.7 Service 2 - Availability Information

Request Availabilities

AvailabilityRequestType: Synchronous request for availabilities of booking targets.
group membership: *UserTriggeredRequestGroup*

	Element	Type	Comment
○ ⊡	*BookingTarget*	`BookingTargetPropertiesType`	
○	*Circle*	`GeoCircleType`	
○	*GeoRectangle*	`GeoRectangleType`	
□	*TimePeriod*	`TimePeriodType`	Interesting time period, for which availabilities of the given booking targets shall be returned. If no given, currently available booking targets should be returned.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AvailabilityRequestType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="BookingTarget" type="BookingTargetPropertiesType" maxOccurs="unbounded"/>
      <xs:element name="Circle" type="GeoCircleType"/>
      <xs:element name="GeoRectangle" type="GeoRectangleType"/>
    </xs:choice>
    <xs:element name="TimePeriod" type="TimePeriodType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.23: AvailabilityRequestType

```
<Ixsi>
    <Request>
        <Transaction>
            <TimeStamp>2014-11-03T11:19:01.976+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <Auth>
            <UserInfo>
                <ProviderID>2</ProviderID>
                <UserID>40</UserID>
                <Password>x</Password>
            </UserInfo>
        </Auth>
        <Availability>
            <BookingTarget>
                <ID>
                    <BookeeID>10</BookeeID>
                    <ProviderID>2</ProviderID>
                </ID>
            </BookingTarget>
            <TimePeriod>
                <Begin>2014-11-04T15:01:00.000+01:00</Begin>
                <End>2014-11-09T17:14:00.000+01:00</End>
            </TimePeriod>
        </Availability>
    </Request>
</Ixsi>
```

XML source code 7.24: AvailabilityRequest Beispiel

AvailabilityResponseType: Synchronous response for availabilites of booking targets.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□ ⊡	*BookingTarget*	`BookingTargetAvailabilityType`	List of booking targets and their availabilities. Not listed booking targets are considered to be either unknown to the car sharing system or not available at all in the requested time period.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AvailabilityResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="BookingTarget" type="BookingTargetAvailabilityType" minOccurs="0" maxOccurs=
            ↪ "unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.25: AvailabilityResponseType

```
<Ixsi>
    <Response>
        <Transaction>
            <TimeStamp>2014-11-03T11:19:01.976+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <CalcTime>PT0.090S</CalcTime>
        <Availability>
            <BookingTarget>
```

```
                <ID>
                    <BookeeID>10</BookeeID>
                    <ProviderID>2</ProviderID>
                </ID>
                <Inavailability>
                    <Begin>2014-11-09T17:00:00.000+01:00</Begin>
                    <End>2014-11-09T20:00:00.000+01:00</End>
                </Inavailability>
                <CurrentStateOfCharge>57</CurrentStateOfCharge>
            </BookingTarget>
        </Availability>
    </Response>
</Ixsi>
```

XML source code 7.26: AvailabilityResponse Beispiel

Request Location Capacities (Service 2a)

PlaceAvailabilityRequestType: Synchronous request for the available capacity of places.
group membership: *UserTriggeredRequestGroup*

	Element	Type	Comment
○ ⊡	*PlaceID*	`ProviderPlaceIDType`	
○	*Circle*	`GeoCircleType`	
○	*GeoRectangle*	`GeoRectangleType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceAvailabilityRequestType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="PlaceID" type="ProviderPlaceIDType" maxOccurs="unbounded"/>
      <xs:element name="Circle" type="GeoCircleType"/>
      <xs:element name="GeoRectangle" type="GeoRectangleType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.27: PlaceAvailabilityRequestType

PlaceAvailabilityResponseType: Synchronous response for the available capacity of places..
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□ ⊡	*Place*	`PlaceAvailabilityType`	List of places and their available capacity. Not listed places are considered to be either unknown to the car sharing system or not available at all.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceAvailabilityResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Place" type="PlaceAvailabilityType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.28: PlaceAvailabilityResponseType

7.8 Service 3 - Availability Subscription

Availability Subscription

AvailabilitySubscriptionRequestType: Request for subscribing to availabilities of given booking targets.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
⊡	*BookingTargetID*	`BookingTargetIDType`	Booking tragets to subscribe/unsubscribe.
□	*Unsubscription*	`xs:boolean`	If set, unsubcribe from the given booking targets. Otherwise subscribe to them.
□	*EventHorizon*	`xs:duration`	If set, the event horizon defines the time in future, for wich the subscription is valid. This value is used for all subscribed booking targets. This duration is relative to the current time.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AvailabilitySubscriptionRequestType">
  <xs:sequence>
    <xs:element name="BookingTargetID" type="BookingTargetIDType" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
    <xs:element name="EventHorizon" type="xs:duration" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.29: AvailabilitySubscriptionRequestType

```
<Ixsi>
    <SubscriptionRequest>
        <Transaction>
            <TimeStamp>2014-11-03T11:23:47.309+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <AvailabilitySubscription>
            <BookingTargetID>
                <BookeeID>7</BookeeID>
                <ProviderID>2</ProviderID>
            </BookingTargetID>
        </AvailabilitySubscription>
    </SubscriptionRequest>
</Ixsi>
```

XML source code 7.30: AvailabilitySubscriptionRequest Beispiel

AvailabilitySubscriptionResponseType: Response for subscribing to availabilities of given booking targets.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AvailabilitySubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.31: AvailabilitySubscriptionResponseType

AvailabilitySubscriptionStatusRequestType: Request for getting all subsribed booking targets.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

AvailabilitySubscriptionStatusResponseType: Response for getting all subscribed booking targets.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*BookingTargetID*	`BookingTargetIDType`	Subscribed booking targets.

AvailabilityPushMessageType: Push message containing changes in availability of subscribed booking targets.
group membership: *PushMessageGroup*

	Element	Type	Comment
⊡	*AvailabilityChange*	`BookingTargetChangeAvailabilityType`	

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="AvailabilityPushMessageType">
  <xs:sequence>
    <xs:element name="AvailabilityChange" type="BookingTargetChangeAvailabilityType" maxOccurs="
        ↪ unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.32: AvailabilityPushMessageType

```xml
<Ixsi>
    <SubscriptionMessage>
        <AvailabilityPush>
            <AvailabilityChange>
                <ID>
                    <BookeeID>7</BookeeID>
                    <ProviderID>2</ProviderID>
                </ID>
                <Inavailability>
                    <Begin>2014-11-04T15:00:00.000+01:00</Begin>
                    <End>2014-11-04T17:30:00.000+01:00</End>
                </Inavailability>
            </AvailabilityChange>
        </AvailabilityPush>
    </SubscriptionMessage>
</Ixsi>
```

XML source code 7.33: AvailabilityPush Beispiel

Complete Availability Subscription Information

CompleteAvailabilityRequestType: Request for getting the complete current status of subscribed availabilities.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxTargets*	`xs:integer`	Defines the maximum number of targets to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteAvailabilityRequestType">
  <xs:sequence>
    <xs:element name="MaxTargets" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.34: CompleteAvailabilityRequestType

```
<Ixsi>
    <SubscriptionRequest>
        <Transaction>
            <TimeStamp>2014-11-03T11:23:30.059+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <CompleteAvailability>
            <MaxTargets>5</MaxTargets>
        </CompleteAvailability>
    </SubscriptionRequest>
</Ixsi>
```

XML source code 7.35: CompleteAvailabilityRequest Beispiel

CompleteAvailabilityResponseType: Response for getting the complete current status of subscribed abailabilities. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow.
□ ⊡	*BookingTarget*	`BookingTargetAvailabilityType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteAvailabilityResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
        <xs:element name="BookingTarget" type="BookingTargetAvailabilityType" minOccurs="0" maxOccurs=
            ↪ "unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.36: CompleteAvailabilityResponseType

```
<Ixsi>
    <SubscriptionResponse>
        <Transaction>
            <TimeStamp>2014-11-03T11:23:30.059+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <CalcTime>PT0.000S</CalcTime>
        <CompleteAvailability>
            <MessageBlockID>100</MessageBlockID>
            <Last>true</Last>
            <BookingTarget>
                <ID>
                    <BookeeID>14</BookeeID>
                    <ProviderID>2</ProviderID>
                </ID>
                <Inavailability>
                    <Begin>2014-11-04T15:00:00.000+01:00</Begin>
                    <End>2014-11-04T18:00:00.000+01:00</End>
                </Inavailability>
            </BookingTarget>
            <BookingTarget>
                <ID>
                    <BookeeID>15</BookeeID>
                    <ProviderID>2</ProviderID>
                </ID>
            </BookingTarget>
            <BookingTarget>
                <ID>
                    <BookeeID>16</BookeeID>
                    <ProviderID>2</ProviderID>
                </ID>
            </BookingTarget>
        </CompleteAvailability>
    </SubscriptionResponse>
</Ixsi>
```

XML source code 7.37: CompleteAvailabilityResponse Beispiel

Location Capacity Subscription (Service 3a)

PlaceAvailabilitySubscriptionRequestType: Request for subscribing to available capacity of places.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
⊡	*PlaceID*	`ProviderPlaceIDType`	Places to subscribe/unsubscribe.
□	*Unsubscription*	`xs:boolean`	If set, unsubcribe from the given booking targets. Otherwise subscribe to them.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceAvailabilitySubscriptionRequestType">
  <xs:sequence>
    <xs:element name="PlaceID" type="ProviderPlaceIDType" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.38: PlaceAvailabilitySubscriptionRequestType

PlaceAvailabilitySubscriptionResponseType: Response for subscribing to available capacity of places.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceAvailabilitySubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.39: PlaceAvailabilitySubscriptionResponseType

PlaceAvailabilitySubscriptionStatusRequestType: Request for getting all subsribed places.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

PlaceAvailabilitySubscriptionStatusResponseType: Response for getting all subscribed places.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*PlaceID*	`ProviderPlaceIDType`	Subscribed places.

PlaceAvailabilityPushMessageType: Push message containing changes in availability of subscribed places.
group membership: *PushMessageGroup*

	Element	Type	Comment
⊡	*PlaceAvailability*	`PlaceAvailabilityType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PlaceAvailabilityPushMessageType">
  <xs:sequence>
    <xs:element name="PlaceAvailability" type="PlaceAvailabilityType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.40: PlaceAvailabilityPushMessageType

Complete Location Capacity Information (Service 3a)

CompletePlaceAvailabilityRequestType: Request for getting the complete current status of subscribed places.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxPlaces*	`xs:integer`	Defines the maximum number of places to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompletePlaceAvailabilityRequestType">
  <xs:sequence>
    <xs:element name="MaxPlaces" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.41: CompletePlaceAvailabilityRequestType

CompletePlaceAvailabilityResponseType: Response for getting the complete current status of subscribed places. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow.
□ ⊡	*PlaceAvailability*	`PlaceAvailabilityType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompletePlaceAvailabilityResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
      <xs:element name="PlaceAvailability" type="PlaceAvailabilityType" minOccurs="0" maxOccurs="
          ↪ unbounded"/>
    </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

XML source code 7.42: CompletePlaceAvailabilityResponseType

7.9 Service 4 - Booking / Booking Change

Booking

BookingRequestType: Request for booking a booking target. A booking does not imply a inavailability.
group membership: *OriginDestGroup UserTriggeredRequestGroup*

Element	Type	Comment
BookingTargetID	`BookingTargetIDType`	
TimePeriodProposal	`TimePeriodProposalType`	Proposed time period for the usage of the booking target.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingRequestType">
  <xs:sequence>
    <xs:element name="BookingTargetID" type="BookingTargetIDType"/>
    <xs:element name="TimePeriodProposal" type="TimePeriodProposalType"/>
    <xs:group ref="OriginDestGroup"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.43: BookingRequestType

```
<Ixsi>
    <Request>
        <Transaction>
            <TimeStamp>2014-11-03T11:19:02.258+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <Auth>
            <UserInfo>
                <ProviderID>2</ProviderID>
                <UserID>40</UserID>
                <Password>x</Password>
            </UserInfo>
        </Auth>
        <Booking>
            <BookingTargetID>
                <BookeeID>14</BookeeID>
                <ProviderID>2</ProviderID>
            </BookingTargetID>
            <TimePeriodProposal>
                <Begin>2014-11-04T15:21:00.000+01:00</Begin>
                <End>2014-11-04T17:18:00.000+01:00</End>
            </TimePeriodProposal>
        </Booking>
    </Request>
</Ixsi>
```

XML source code 7.44: BookingRequest Beispiel

BookingResponseType: Response for booking a booking target.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□	*Booking*	BookingType	Information on a successful booking.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Booking" type="BookingType" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.45: BookingResponseType

```
<Ixsi>
    <Response>
        <Transaction>
            <TimeStamp>2014-11-03T11:19:02.258+01:00</TimeStamp>
            <MessageID>100</MessageID>
        </Transaction>
        <CalcTime>PT0.325S</CalcTime>
        <Booking>
            <BookingID>2-9</BookingID>
            <TimePeriod>
                <Begin>2014-11-04T15:00:00.000+01:00</Begin>
                <End>2014-11-04T17:30:00.000+01:00</End>
            </TimePeriod>
        </Booking>
    </Response>
</Ixsi>
```

XML source code 7.46: BookingResponse Beispiel

Booking Change

ChangeBookingRequestType: Request for changing a booking.
group membership: *UserTriggeredRequestGroup*

	Element	Type	Comment
	BookingID	`BookingIDType`	Booking to change.
○	*NewTimePeriodProposal*	`TimePeriodProposalType`	New time period proposal for changing the booking.
○	*Cancel*	`xs:boolean`	The booking shall be cancelled.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangeBookingRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:choice>
      <xs:element name="NewTimePeriodProposal" type="TimePeriodProposalType"/>
      <xs:element name="Cancel" type="xs:boolean"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.47: ChangeBookingRequestType

ChangeBookingResponseType: Response for changing a booking.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□	*Booking*	`BookingType`	Information on a successful re-booking.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangeBookingResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Booking" type="BookingType" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.48: ChangeBookingResponseType

Request External Bookings (Service 4a)

ExternalBookingSubscriptionRequestType: Request to fetch BookingIDs referencing external bookings. Can be use to gather consumption data.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
□ ⊡	*UserInfo*	`UserInfoType`	Only get bookings of respective user (get all available of omitted).
□	*Unsubscription*	`xs:boolean`	If set, unsubcribe from the given external bookings. Otherwise subscribe to them.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ExternalBookingSubscriptionRequestType">
  <xs:sequence>
    <xs:element name="UserInfo" type="UserInfoType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.49: ExternalBookingSubscriptionRequestType

ExternalBookingSubscriptionResponseType: Response for subscribing to alerts for given external bookings.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ExternalBookingSubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.50: ExternalBookingSubscriptionResponseType

ExternalBookingSubscriptionStatusRequestType: Request for getting all for external bookings subscribed users.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

ExternalBookingSubscriptionStatusResponseType: Response for getting all for external bookings subscribed users.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*UserInfo*	`UserInfoType`	Subscribed Users.

ExternalBookingPushMessageType: Response to request for external bookings.
Basetype: *AbstractBaseResponseType*
group membership: *PushMessageGroup*

Element	Type	Comment
ExternalBooking	`ExternalBookingType`	Information about bookings added externally.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ExternalBookingPushMessageType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="ExternalBooking" type="ExternalBookingType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.51: ExternalBookingPushMessageType

CompleteExternalBookingRequestType: Request for getting the complete current status of subscribed external bookings.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxResults*	`xs:integer`	Defines the maximum number of results to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteExternalBookingRequestType">
  <xs:sequence>
    <xs:element name="MaxResults" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.52: CompleteExternalBookingRequestType

CompleteExternalBookingResponseType: Message for getting the complete current status of subscribed external bookings. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow.
□ ⊡	*ExternalBooking*	`ExternalBookingType`	Information about bookings added externally.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteExternalBookingResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
        <xs:element name="ExternalBooking" type="ExternalBookingType" minOccurs="0" maxOccurs="
            ↪ unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.53: CompleteExternalBookingResponseType

7.10 Service 5 - Booking Subscription

Booking Subscription

BookingAlertSubscriptionRequestType: Request for subscribing to alerts for given bookings.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
⊡	*BookingID*	`BookingIDType`	Bookings to subscribe/unsubscribe.
□	*Unsubscription*	`xs:boolean`	If set, unsubcribe from the given bookings. Otherwise subscribe to them.

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingAlertSubscriptionRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.54: BookingAlertSubscriptionRequestType

BookingAlertSubscriptionResponseType: Response for subscribing to alerts for given bookings.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingAlertSubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.55: BookingAlertSubscriptionResponseType

BookingAlertSubscriptionStatusRequestType: Request for getting all subscribed bookings.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

BookingAlertSubscriptionStatusResponseType: Response for getting all subscribed bookings.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*BookingID*	`BookingIDType`	Subscribed bookings.

BookingAlertPushMessageType: Push message containing alerts for subscribed bookings.
group membership: *PushMessageGroup*

	Element	Type	Comment
⊡	*BookingChange*	`BookingChangeType`	

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingAlertPushMessageType">
  <xs:sequence>
    <xs:element name="BookingChange" type="BookingChangeType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.56: BookingAlertPushMessageType

Complete Booking Subscription Information

CompleteBookingAlertRequestType: Request for getting the complete current status of subscribed booking alerts.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxResults*	`xs:integer`	Defines the maximum number of results to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteBookingAlertRequestType">
  <xs:sequence>
    <xs:element name="MaxResults" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.57: CompleteBookingAlertRequestType

CompleteBookingAlertResponseType: Message for getting the complete current status of subscribed bookings. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow.
□ ⊡	*BookingChange*	`BookingChangeType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteBookingAlertResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
        <xs:element name="BookingChange" type="BookingChangeType" minOccurs="0" maxOccurs="unbounded"/
          ↪ >
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.58: CompleteBookingAlertResponseType

7.11 Service 6 - Price Inquiry

Price Information

PriceInformationRequestType: Request for getting a price.
group membership: *OriginDestGroup UserTriggeredRequestGroup*

Element	Type	Comment
BookingTargetID	`BookingTargetIDType`	
TimePeriodProposal	`TimePeriodProposalType`	Proposed time period for the usage of the booking target.
Distance	`xs:nonNegativeInteger`	Distance to drive in meters.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PriceInformationRequestType">
  <xs:sequence>
    <xs:element name="BookingTargetID" type="BookingTargetIDType"/>
    <xs:element name="TimePeriodProposal" type="TimePeriodProposalType"/>
    <xs:element name="Distance" type="xs:nonNegativeInteger"/>
    <xs:group ref="OriginDestGroup"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.59: PriceInformationRequestType

PriceInformationResponseType: Response for getting a price.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□	*Tariff*	`TariffType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="PriceInformationResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Tariff" type="TariffType" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.60: PriceInformationResponseType

7.12 Service 7 - Consumption Data Subscription

Consumption Data Subscription

ConsumptionSubscriptionRequestType: Request to create or cancel a consumption data subscription.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
⊡	*BookingID*	`BookingIDType`	Booking ID to subscribe for.
□	*Unsubscription*	`xs:boolean`	If true, cancel subscription.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ConsumptionSubscriptionRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.61: ConsumptionSubscriptionRequestType

ConsumptionSubscriptionResponseType: Response to consumption data subscription.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ConsumptionSubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.62: ConsumptionSubscriptionResponseType

ConsumptionSubscriptionStatusRequestType: Request for getting all subscribed consumption data items.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

ConsumptionSubscriptionStatusResponseType: Response for getting all subscribed consumption data items.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*BookingID*	`BookingIDType`	Subscribed bookings.

ConsumptionPushMessageType: Push message containing alerts for subscribed bookings.
group membership: *PushMessageGroup*

	Element	Type	Comment
⊡	*Consumption*	`ConsumptionType`	

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ConsumptionPushMessageType">
  <xs:sequence>
    <xs:element name="Consumption" type="ConsumptionType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.63: ConsumptionPushMessageType

Complete Consumption Data

CompleteConsumptionRequestType: Request for getting the complete current status of subscribed consumption data items.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxResults*	`xs:integer`	Defines the maximum number of results to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteConsumptionRequestType">
  <xs:sequence>
    <xs:element name="MaxResults" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.64: CompleteConsumptionRequestType

CompleteConsumptionResponseType: Message for getting the complete current status of subscribed consumption data. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow
□ ⊡	*Consumption*	`ConsumptionType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteConsumptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
        <xs:element name="Consumption" type="ConsumptionType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.65: CompleteConsumptionResponseType

7.13 Service 8 - Change Booking Status

ChangeBookingStateRequestType: Request to change to booking state.
group membership: *UserTriggeredRequestGroup*

Element	Type	Comment
BookingID	`BookingIDType`	The respective booking.
BookingState	`BookingStateType`	The new state of the booking (open = unlock vehicle, suspended = lock vehicle but dont close booking, close = lock and return vehicle).

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangeBookingStateRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:element name="BookingState" type="BookingStateType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.66: ChangeBookingStateRequestType

ChangeBookingStateResponseType: Response to change booking state with attributes to be conveyed to the customer.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□ ⊡	*Attributes*	`AttributeType`	

```xml
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangeBookingStateResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="Attributes" type="AttributeType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.67: ChangeBookingStateResponseType

7.14 Service 9 - User Management

Create User

CreateUserRequestType: Request for creating a user (deleting is intentionally not supported, only locking).

group membership: *UserTriggeredRequestGroup*

	Element	Type	Comment
⊡	*User*	`UserType`	Proposed User info.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CreateUserRequestType">
  <xs:sequence>
    <xs:element name="User" type="UserType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.68: CreateUserRequestType

CreateUserResponseType: Response for creating users.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□ ⊡	*User*	`UserType`	Information about a successful created user (might differ, unsuccessful creation attempts are not returned).

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CreateUserResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="User" type="UserType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.69: CreateUserResponseType

Change User

ChangeUserRequestType: Request for changing users.
group membership: *UserTriggeredRequestGroup*

	Element	Type	Comment
⊡	*User*	`UserType`	changed user info.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangeUserRequestType">
  <xs:sequence>
    <xs:element name="User" type="UserType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.70: ChangeUserRequestType

ChangeUserResponseType: Response for changing a user.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

	Element	Type	Comment
□ ⊡	*User*	`UserType`	Information on changed user (unsuccessful changes are not returned).

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="ChangeUserResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="User" type="UserType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.71: ChangeUserResponseType

7.15 Service 10 - Vehicle Settings Management

Set Vehicle Settings

SetBookingSettingsRequestType: Request for setting vehicle settings for a specific booking.
group membership: *UserTriggeredRequestGroup*

	Element	Type	Comment
⊡	*BookingSettings*	`BookingSettingsType`	Set of settings to be applied (e.g. seat position, air conditioning, radio station etc).

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="SetBookingSettingsRequestType">
  <xs:sequence>
    <xs:element name="BookingSettings" type="BookingSettingsType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.72: SetBookingSettingsRequestType

SetBookingSettingsResponseType: Response for setting vehicle settings.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="SetBookingSettingsResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.73: SetBookingSettingsResponseType

Subscribe to Vehicle Settings

BookingSettingsSubscriptionRequestType: Request to create or cancel a settings data subscription.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
⊡	*BookingID*	`BookingIDType`	Booking ID to subscribe for.
□	*Unsubscription*	`xs:boolean`	If true, cancel subscription.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingSettingsSubscriptionRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.74: BookingSettingsSubscriptionRequestType

BookingSettingsSubscriptionResponseType: Response to settings data subscription.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingSettingsSubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.75: BookingSettingsSubscriptionResponseType

BookingSettingsSubscriptionStatusRequestType: Request for getting all subscribed settings data items.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

BookingSettingsSubscriptionStatusResponseType: Response for getting all subscribed settings data items.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*BookingID*	`BookingIDType`	Subscribed bookings.

CompleteBookingSettingsRequestType: Request for getting the complete current status of subscribed booking settings data items.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxResults*	`xs:integer`	Defines the maximum number of results to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteBookingSettingsRequestType">
  <xs:sequence>
    <xs:element name="MaxResults" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.76: CompleteBookingSettingsRequestType

CompleteBookingSettingsResponseType: Message for getting the complete current status of subscribed booking settings data. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow
□ ⊡	*BookingSettings*	`BookingSettingsType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteBookingSettingsResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
        <xs:element name="BookingSettings" type="BookingSettingsType" minOccurs="0" maxOccurs="
            ↪ unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.77: CompleteBookingSettingsResponseType

BookingSettingsPushMessageType: Push message containing alerts for subscribed bookings.
group membership: *PushMessageGroup*

	Element	Type	Comment
⊡	*BookingSettings*	`BookingSettingsType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingSettingsPushMessageType">
  <xs:sequence>
    <xs:element name="BookingSettings" type="BookingSettingsType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.78: BookingSettingsPushMessageType

7.16 Service 11 - Remote Configuration Navigation System and Journey Progress Monitoring

Set Target Location of Navigation System

SetNavigationDestinationRequestType: Request to set the navigation destination.
group membership: *UserTriggeredRequestGroup*

Element	Type	Comment
BookingID	`BookingIDType`	The respective booking.
Destination	`GeoPositionType`	The destination of the booking target.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="SetNavigationDestinationRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType"/>
    <xs:element name="Destination" type="GeoPositionType"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.79: SetNavigationDestinationRequestType

SetNavigationDestinationResponseType: Response to set the navigation destination.
Basetype: *AbstractBaseResponseType*
group membership: *UserTriggeredResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="SetNavigationDestinationResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.80: SetNavigationDestinationResponseType

Journey Progress (Booking Progress) Monitoring

BookingProgressSubscriptionRequestType: Request to create or cancel a a booking progression subscription.
group membership: *SubscriptionRequestGroup*

	Element	Type	Comment
⊡	*BookingID*	`BookingIDType`	Booking ID to subscribe for.
□	*Unsubscription*	`xs:boolean`	If true, cancel subscription.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingProgressSubscriptionRequestType">
  <xs:sequence>
    <xs:element name="BookingID" type="BookingIDType" maxOccurs="unbounded"/>
    <xs:element name="Unsubscription" type="xs:boolean" default="false" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.81: BookingProgressSubscriptionRequestType

BookingProgressSubscriptionResponseType: Response to create or cancel a a booking progression subscription.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

Element	Type	Comment
(empty)		

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema" name="BookingProgressSubscriptionResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType"/>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.82: BookingProgressSubscriptionResponseType

BookingProgressSubscriptionStatusRequestType: Request for getting all subscribed progress items.
group membership: *SubscriptionRequestGroup*

Element	Type	Comment
(empty)		

BookingProgressSubscriptionStatusResponseType: Response for getting all subscribed progress items.
Basetype: *AbstractBaseResponseType*
group membership: *SubscriptionResponseGroup*

	Element	Type	Comment
□ ⊡	*BookingID*	`BookingIDType`	Subscribed bookings.

CompleteBookingProgressRequestType: Request for getting the complete current status of subscribed booking progress items.
group membership: *RequestMessageGroup*

	Element	Type	Comment
□	*MaxResults*	`xs:integer`	Defines the maximum number of results to be returned in one message.

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteBookingProgressRequestType">
  <xs:sequence>
    <xs:element name="MaxResults" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.83: CompleteBookingProgressRequestType

CompleteBookingProgressResponseType: Message for getting the complete current status of subscribed booking progress data. Several such responses may belong to one single request.
Basetype: *AbstractBaseResponseType*
group membership: *ResponseMessageGroup*

	Element	Type	Comment
	MessageBlockID	`xs:token`	ID of the block, this message belongs to. All messages of one block have the same ID.
□	*Last*	`xs:boolean`	If set, this message is the last one in the corresponding block. Otherwise more messages of the same block will follow
□ ⊡	*BookingProgress*	`BookingProgressType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="CompleteBookingProgressResponseType">
  <xs:complexContent>
    <xs:extension base="AbstractBaseResponseType">
      <xs:sequence>
        <xs:element name="MessageBlockID" type="xs:token"/>
        <xs:element name="Last" type="xs:boolean" default="false" minOccurs="0"/>
        <xs:element name="BookingProgress" type="BookingProgressType" minOccurs="0" maxOccurs="
            ↪ unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XML source code 7.84: CompleteBookingProgressResponseType

BookingProgressPushMessageType: Push message containing alerts for subscribed bookings.
group membership: *PushMessageGroup*

	Element	Type	Comment
⊡	*BookingProgress*	`BookingProgressType`	

```
<xs:complexType xmlns="http://www.ixsi-schnittstelle.de/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
    ↪ name="BookingProgressPushMessageType">
  <xs:sequence>
    <xs:element name="BookingProgress" type="BookingProgressType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

XML source code 7.85: BookingProgressPushMessageType

8 Code Tables

The syntactical definition for IXSI does not contain concrete values (enumerations) for e.g., vehicle types or error codes. Instead, these values are determined within the following code tables. Only these determined values are allowed to be used by IXSI.

8.1 Vehicle Types

For the selection *ClassType*, the following values can be used:

Value	Meaning
`bike`	Bike
`motorcycle`	Motorcycle
`micro`	Micro car (z. B. Smart4two)
`mini`	Mini car (z. B. Opel Corsa)
`small`	Small car (z. B. VW Golf)
`medium`	Medium class car (z. B. Audi A4)
`large`	Upper class car (z. B. BMW 7er)
`van`	Van (z. B. VW T5 Multivan)
`transporter`	Transporter (z. B. Ford Transit)

8.2 Engine Types

For the selection *EngineType*, the following values can be used:

Value	Meaning
`none`	No engine (Muscle Power)
`diesel`	Diesel engine
`gasoline`	Otto engine
`electric`	Electro engine
`liquidgas`	Liquid gas (LPG)
`naturalgas`	Natural gas (CNG)
`hydrogen`	Hydrogen power
`hybrid`	Hybrid power with electro- and internal combustion engine

Note: A pedelec can be described as a combination of vehicle type `bike` and engine type `electric`.

8.3 Vehicle Attributes

Attributes and characteristics of a booking target can be classified to interpret them automatically. Therefore, attribute classes are used. Attribute classes are saved in IXSI in the data type *AttributeClassType*. The following values are allowed:

Value	Meaning
`trailer_hitch`	Trailer hitch
`automatic,`	Automatic gear
`convertible`	Cabriolet
`air_condition`	Air condition
`navigation`	Navigation system
`cruise_control`	Cruise control
`winter_tyres`	Winter and all season tires
`child_seat_0`	Child seat
`child_seat_1`	Child seat (9-18kg)
`child_seat_4`	Child seat (15-36kg)
`utility`	Kombi
`doors_4`	4/5 Doors
`seats_9`	At least 9 seats
`seats_7`	At least 7 seats
`seats_5`	At least 5 seats
`seats_4`	At least 4 seats

8.4 Consumption Data

Classes of consumption data for accounting of services, e.g., distance, duration or additional services like seats for children. Is saved in *ConsumptionClassType*. The following values are allowed:

Value	Meaning
`distance`	Distance
`duration`	Duration

8.5 User States

States, a user can have in the system. Is saved in *UserStateType*. The following values are allowed:

Values	Meaning
`operative`	User is activated and allowed to book vehicles.
`nonoperative`	User is locked and not allowed to book vehicles (possible to be unlocked).
`deleted`	User is (finally) deleted.

8.6 Error Codes

Instead of usual response-, update- or handshake-messages, the VRS is able to transfer error-messages to the TIS in case an error occurs. Error codes are saved in the data type *ErrorCodeType*.

Values	Meaning
`auth_provider_unknown`	Authentication: Unknown provider-ID
`auth_invalid_password`	Authentication: User-password-combination invalid
`auth_invalid_token`	Authentication: User-token-combination invalid
`auth_session_invalid`	Authentication: Session is invalid/ expired
`auth_anon_not_allowed`	Authentication: Anonymous user not allowed
`auth_not_authorized`	Autorization: User is not justified for this request
`sys_backend_failed`	System: Background system does not respond
`sys_unknown_failure`	System: Unknown error
`sys_not_implemented`	System: Request not implemented
`sys_request_not_plausible`	System: Request is not plausible. This value should always be used in case of content errors, included in the request.
`booking_target_unknown`	Booking target unknown
`price_info_not_available`	Price information not available
`booking_too_short`	Booking duration too short
`booking_too_long`	Booking duration too long
`booking_target_not_available`	Booking target not bookable in the given time span
`booking_change_not_possible`	Booking change can not be executed
`booking_id_unknown`	Unknown booking-ID. This value should also be used, if the booking-ID is assigned to a different user.
`booking_state_change_failed`	Changes of booking status failed.
`booking_settings_not_understood`	Booking settings unknown.
`booking_navigation_set_failed`	Setting of journey target failed.
`language_not_supported`	Requested language not unconditionally supported, other language provided.

Source Code Index

Index