TIMOCOM API Freight Exchange

Web Service Interface Documentation

TIMOCOM GmbH <support.api@timocom.com>

Version 2.6.0, 2024-01

Table of Contents

Chapter 1. General

The *TIMOCOM API Freight Exchange* (hereinafter referred to as the "*API*") is the web service interface for importing data into TIMOCOM's *Freight Application* and *Vehicle Space Application* (hereinafter referred to as "*Apps*"). It is capable of processing the complete lifecycle for the following entity types:

- Freight (Cargo) offers
- Loading space (Truck) offers
- Contact persons
- Price proposals

Furthermore, there is a read-only interface for getting information about your customers and the reference data (like countries, languages, vehicle body types, etc) used in other entities.

Specifically, the API can be used to carry out the following actions:

- Create, delete and modify contact persons
- Create, delete and modify freight offers
- Create, delete and modify loading space offers
- Create, delete and modify price proposals
- Retrieval of entity IDs from your own data
- Retrieval of detail information from your own data (a.k.a. "Inside Search")
- Retrieval of offer detail information from all data (a.k.a. "Outside Search")

This document is a technical description of the *API* web service interface.

After a brief introduction about the webservice, authentication and the request/response handling the chapter Clarifying the Term sets up a common understanding (a.k.a. ubiquitous language) for the entities handled by this web service.

The chapter Response Handling is about how to handle the responses (in particular in case of errors). In chapter API Methods (Endpoints) we dive into the API looking at all endpoints. Each endpoint has its example XML in order to make the understanding easier.

The chapters Types of the XML Schema Definition and Reference Data Index repeat the annotated XSD annotations in a more readable manner.

And finally, if you want to start coding you should read chapter Getting Started in the first place; it contains tips, hints and code examples. Furthermore, it outlines the important aspect of authentication and security.

Chapter 2. The Webservice

2.1. Staging

The *API* web services runs on two stages:

- Certification
- Production

2.1.1. Certification Stage

This stage serves as a testing and training stage for you and has its own URL. The intention is that you implement the client against this stage in the first place. Your TIMOCOM contact person may then move your access data to the Production stage as soon you are successful and you have a valid *API Freight Exchange* contract.

2.2. SOAP Version

The *API* v2 supports SOAP 1.2, only.

The *API* web service interface works in a document-centred (document-style) mode. The SOAP request (Request) to the interface contains an XML document specifying the data sets that are to be created, deleted or modified. The structure of the XML document is defined using several XML schemas.

The *API* replies to the SOAP request with a SOAP response (Response) that documents the processing result.

2.3. URLs

The *API* Web Service interface is called via the following URLs:

Table 1. API Freight Exchange API v2 URLs

Stage	URL
CERTIFICATON	https://ws-test.timocom.com/tcconnect/ws_v2/soap1_2
PRODUCTION	https://webservice.timocom.com/tcconnect/ws_v2/soap1_2

Chapter 3. Authentication

Each request must contain an authentication element in the first place. If you agreed performing Web Service Security this element just contains your group name, otherwise it must contain group name and password, separated by a colon ':' without space characters. See Authentication and Security how to set up your authentication credentials.

Chapter 4. Request, Response and Payload

The *API* acts resource based, i.e. each data manipulation request handles only one entity. However query requests may return multiple entities.

The different request types follow a common naming pattern:

Table 2. Request Naming Patterns for an Entity E, where E serves as a placeholder for the entity

Pattern	Super Type	Meaning	Available for
`Store<E>Request`	`StoreEntityRequest`	Create or update an entity	CargoOffer, TruckOffer, ContactPerson
`Delete<E>Request`	`DeleteCustomer-ReferencingEntity-RequestType`	Delete an entity	CargoOffer, TruckOffer, ContactPerson
`Get<E>Request`	`GetEntityRequestType`	Get entity detail by its ID	CargoOffer, TruckOffer, ContactPerson, Customer
`Find<E>sRequest`	`FindEntitiesRequest-Type`	Find your entities detail by a filter	CargoOffer, TruckOffer, ContactPerson, Customer
`Find<E>KeysRequest`	`FindPrimaryKeys-RequestType`	Find your entities (IDs only) by a filter	CargoOffer, TruckOffer, ContactPerson, Customer
`GetReferenceData-Request`	-	Get all reference data	ReferenceData

Store Means Create or Update

- If you want to create or update an entity you have to use the Store request, e.g. `StoreCargoOfferRequest`.
- There is no partial update, i.e. for an update you have to send the complete object.

Each method (also known as *endpoint*) on *API* side corresponds to a certain request type (inherits from `TCConnectRequestType` of the XML schema) and in an analogous manner, to a certain response type (`TCConnectResponseType`).

If you send a `GetTruckOfferRequest` you will get a `GetTruckOfferResponse`.

Each request contains a `Payload` which is in case of a data modification request the entity, a key object or in case of a query, a filter object. Successful responses of query (Get and Find) requests contain a `Payload`, too.

Table 3. Request – Response – Payload Examples

Request	Request-Payload	Response	Response-Payload
StoreCargoOfferRequest	CargoOfferType	StoreCargoOffer-Response	null
GetCargoOfferRequest	CustomerReferencing-EntityFilterType	GetCargoOfferResponse	CargoOfferType
DeleteCargoOffer-Request	DeleteCustomerReferencingEntityRequestType	DeleteCargoOffer-Response	null
FindCargoOffersRequest	ExtendedCustomer-Referencing-EntityFilterType	FindCargoOffersResponse	CargoOfferListType
FindCargoOfferKeysRequest	ExtendedCustomer-Referencing-EntityFilterType	FindCargoOfferKeys-Response	CustomerReferencing-EntityKeyListType
GetReferencedData-Request	FilterType	GetReferencedDataResponse	ReferencedDataType

Chapter 5. Clarifying the Term

5.1. CargoOffer

Describes the details of a freight offer.

5.2. TruckOffer

Describes the details of a loading space (or vehicle) offer.

5.3. Vehicle Properties

Vehicle properties describe the present resp. required properties in a loading space resp freight offer. For instance the vehicle bodies *box* or *curtain*, the vehicle type *trailer* or equipment like *access ramp* or vehicle load securing like *lashing straps*.

Currently, there are five categories of vehicle properties:

- Vehicle body (e.g. box, curtain)
- Vehicle body property (e.g. widenable, sliding roof)
- Vehicle swap body (e.g. roll-on roll-off container, 40 ft container)
- Vehicle equipment (e.g. partition wall, tarpaulin cover)
- Vehicle load securing (e.g. lashing straps, perforated batten)
- Vehicle type (e.g. waggon and drag (rigid truck), vehicle up to 12 tons)

These categories' values are reference data. See under Reference Data Index for current possible values and Reference Data how reference data are being handled.

The category values are also treated as reference data and may be extended in the near future.

Possible movement of values ("repotting")

With the introduction of new categories, it is possible that values may be "repotted" i.e. moved to another category. However, this will result in a new minor version of the *API* and we move the values according to the version of the request transparently for the client during request processing. In this case you will receive a warning with the message key `VEHICLE_PROPERTY_VALUE_MOVED`. (According to the WSDL/XSD each request must contain a `version`-attribute.)

If you need a list of all available vehicle properties, look at Vehicle Property Category et seq. or fire a `GetReferencedDataRequest`: its response will contain German and English translation of each enumeration values.

5.4. Price Proposal for Cargo Offers

We offer services for the two different scenarios:

1. Proposition: Submitting of price proposal for a freight offer
2. Receival: Getting all price proposals for your freight offer

5.4.1. Price Proposal Proposition

If the freight offer field `acceptPriceProposals` is set to true, price proposals can be submitted for this offer. You can retrieve your own submitted price proposal with a `GetPriceProposalRequest`.

5.4.2. Price Proposal Receival

Freight offerers can retrieve all received price proposals for their freight offer and can then choose to accept one of received price proposals. Accepting is done with a `DeleteCargoOfferRequest` with all remaining price proposals being rejected at the same time.

5.5. Customer and Customer Group

5.5.1. Customer

Describes the details of a TIMOCOM customer. Each customer corresponds to his so called *TimoComID* though the TimoComID is not part of the definition of a Customer within the *API*.

5.5.2. Customer Group

A *customer group* is a set of TIMOCOM customers who share the authentication credentials and further administration information within the *API*.

It often represents a corporate group with its subsidiaries in which a subsidiary is a TIMOCOM customer.

The Customer Group is an intrinsic concept of the *API* and must not be confused with a Closed User Group.

Each Customer Group has got its *customer group name* which serves as the principal name within each request.

5.5.3. Principal

The *principal* from the *API* point of view is synonymic term for the *customer group*, ie the *principal name* is the *customer group name*.

5.6. User and Contact Person

5.6.1. User

A *user* (also: User of the system) has one or more authorised access points (Accounts) to the TIMOCOM platform and is enabled to enter and search for offers and to chat with other users. The user entity as a data type in the *API* is not available but has an effect on the handling of the contact person entity as described below.

5.6.2. Contact Person

Describes the details of a contact person. These details may be either a value object without its own life cycle and thus dependent part of a freight or loading space offer or an entity with a dedicated life cycle which is manageable via the contact person endpoints of the API.

Embedded Contact Person (Value Object)

An embedded contact person is sent within an offer and is disposed when the offer is deleted or expires. The data is nowhere else referenceable or usable in the TIMOCOM *Smart Logistics System*.

A contact embedded within a freight or loading space offer transmitted via the *API* can be used in the TIMOCOM *Smart Logistics System* as following:

1. As contact person for a freight or loading space offer

See also StoreCargoOfferRequest example.

Referenceable and Manageable Contact Person (Entity)

A referenceable contact person is an entity with a dedicated life cycle and is created either via this API or in TIMOCOM's *Smart Logistics System Settings*. Once created it may be used by its `contactRef` in a transmitted cargo or truck offer or may be used in TIMOCOM's *Warehouse Application*.

A contact transmitted via the *API* can be used in the TIMOCOM *Smart Logistics System* as following:

1. As contact person for a freight or loading space offer, which has been entered either via the *API* or the web-interface.
2. As basic data (like name, phone number, language) of a user
3. As well as a contact person in an offer or as basic data of a user

5.6.3. Links Between User and Contact in Settings

Name and surname as well as contact data are, as mentioned above, assigned to a user. That means that every user is also always assigned to one and only one (referenceable) contact person. This contact person cannot be edited or deleted via the *API*, even if it was created via the *API*; it can only be edited or deleted by the user via TIMOCOM's *Smart Logistics System Settings* (otherwise an error message will be returned).

5.6.4. Referencing a Contact Person When Submitting an Offer Via the *API*

The following cases exist:

1. The contact person is or will be used as a basis for a user. A possible offer related chat may be initiated with this user, but it may not be modified nor deleted by the *API*.
2. The contact person is not used as a basis for a user: In this case an offer related chat is not possible, but it may be modified or deleted via the *API*.

5.6.5. Referencing a Contact When Creating an Offer in the *Apps* Web Application

If a user submits an offer in the *Apps*, the application will automatically reference the contact person in the offer that the user has selected in "Settings → My information → Configuration" in the *Smart Logistics System*. This selection can be changed to use a different contact person. Either in *Settings* or while creating an offer it is possible to

- select yourself, i.e. your own user contact. A possible offer related chat is initiated with you
- select another user's Contact. A possible offer related chat is initiated with that user
- select a Contact that is not assigned to a user. In this case an offer related chat is not possible.

5.7. Search Filter for Offers

The so-called "Outside Search" means that you are able to search for any public offer (freight resp. loading space) in the TIMOCOM *Smart Logistics System*. This is implemented with special `FilterType`s (`CargoOfferFilerType` and `TruckOfferFilterType`).

5.7.1. Restrictive Usage

The usage of these filters is restricted to authorized customer groups, only. So if you do not have a valid contract your request will be rejected with a `UNAUTHORIZED_SEARCH_FILTER` message.

5.7.2. Filter Parameters

See Filter Types and Their Fields for parameters of these filters. However, we go into more detail on a few parameters.

Sorting of the result set

It is possible to specify a sorting by the following offer fields (ascending or descending)

- `startDate` – for cargo offers it is the date of the first loading place
- `creationDateTime`

The default sorting is by `startDate` ascending. It is also possible to specify both fields in order to control the sorting if the values for the first specified field are equal.

Date Search

With the complex parameter `date` you may define either an interval or individual (dedicated) dates. The values will be matched against the first and last loading date of the start loading place for cargo offer resp. the start date for truck offers.

This search is about matching the offers' dates, not the technical creation timestamps of the offers.

Date Search by Individual Dates

The individual dates list specify all discrete dates you search for all

- cargo offers whose start loading place's first and last loading date interval match one of the discrete given dates
- truck offers whose start date matches one of the discrete given dates

Date Search by Interval

The interval is specified by a `DateIntervalType` with mandatory `start` and `end` date.

With a date interval [A,B] (left closed, right closed interval) you search for all

- freight offers whose start loading place's interval of earliest and latest loading date match one of the discrete dates between A and B.
- truck offers whose start date matches one of the discrete dates between A and B

E.g. for freight offers: You search for the interval dates [5,8] (month and year omitted here). Then you will find for instance all freight offer whose start loading place has

- the earliest loading date 7 and the latest loading date 12.
- the earliest loading date 4 and the latest loading date 12.
- the earliest loading date 4 and the latest loading date 6.
- the earliest loading date 6 and the latest loading date 7.

5.7.3. Pagination and Query Date Time

The size of a result set of each Outside Search request is limited to 30 offers. For querying the 31st and further offers the pagination parameters must be used:

- `firstResult` – Offset position of the complete result set, starting with 0
- `maxResults` – Page size, between 1 and 30

E.g. for querying the last 10 out of 30 offers set `firstResult=20` and `maxResults=10`.

Query Date Time for Avoiding Gaps

Without the `queryDateTime` parameter pagination it is not guaranteed that all (still public) offers can be requested chunk wise.

Table 4. Example Pagination without query date time

Time	Event
0:00	Offer A inserted
0:10	Offer B inserted
0:20	Offer C inserted
0:30	Query for offers w/ `firstResult=0`, `maxResults=1` returns {A}
0:32	Query for offers w/ `firstResult=1`, `maxResults=1` returns {B}
0:33	Offer A is removed from the platform
0:34	Query for offers w/ `firstResult=2`, `maxResults=1` returns empty result set, not {C}

This scenario may occur with any of the `firstResult` and `maxResults` parameters.

Table 5. Example Pagination with query date time

Time	Event	Query Date Time	Message Key
0:00	Offer A inserted		
0:10	Offer B inserted		
0:20	Offer C inserted		
0:30	Query for offers w/ `firstResult=0`, `maxResults=1` returns {A}	0:30	
0:32	Query for offers w/ `firstResult=1`, `maxResults=1` returns {B}	0:30	
0:33	Offer A is removed from the platform		
0:34	Query for offers w/ `firstResult=2`, `maxResults=1` returns {C}	0:30	
0:35	Query for offers w/ `firstResult=0`, `maxResults=3` returns {B,C}. A is detached from the result set because it was deleted meanwhile	0:30	`DETACHED_DELETED_ENTITIES`

Pagination Best Practice

If you want to paginate until the last element of a result set, you have to paginate until the result is empty and there is no message key `DETACHED_DELETED_ENTITIES` returned in a message. Each pagination requests counts as a request unless the result set is empty.

Comparison of Pagination between the *Apps* and the *API*

The *Apps* also use a query date time in order to avoid the above mentioned gaps, but presents deleted offers strikethrough formatted whereas the *API* detaches them from the result.

Interval Search or New Offers Only

With the parameters `updatedAfterDateTime` and `queryDateTime` you may define a search interval (left open, right closed) in matters of the modification timestamp of the offers in search. If `updatedAfterDateTime` is set it serves as the left bound of a date time interval, ie. you exclude offers which have been modified before or at the `updatedAfterDateTime`. As `queryDateTime` is a mandatory parameter you always define an upper bound for your search.

This search is about the technical creation timestamps of the offers, not about matching the offers' loading or start dates.

Interval Search Use Case

If you have already queried an interval (*A*,*B*] (with *A*, *B* points in time) you may not want to get the results again if you query later at a point in time *C* for more up-to-date offers. So, you will probably query for the interval (*B*,*C*] and will get no offer twice.

The interval (`updatedAfterDateTime`,`queryDateTime`] mimics the button "New Offers Only" of the *Apps* which allows querying for offers created or modified after the last executed query.

Whereas the Web frontend stores the point in time *B* in its session, you will have to set it here as the lower bound (`updatedAfterDateTime`) in the next request because of the statelessness of the *API*.

5.8. Lookup Filter for Offers

The so called "Lookup" means that you are looking up a set of offers by submitting a set of publicIds. Use the endpoints Looking Up Cargo Offers Outside of Your Customer Group resp. Looking Up Loading Space Offers Outside of Your Customer Group and see Outside Lookup for the filter.

5.8.1. Restrictive Usage

The usage of these filters is restricted to authorized customer groups, only. So if you do not have a valid contract your request will be rejected with a `UNAUTHORIZED_SEARCH_FILTER` message.

5.9. IDs, Refs and PublicIds

In the web service data structures, you will find reference elements and ID elements. This section explains the difference between these elements.

Every request which creates, modifies or deletes an entity must contain exactly one `id` element which identifies this entity.

If you are referring to another entity the element is called `<entityName>Ref`. For instance in an offer or contact person storage request you will have to refer to a customer by the `customerRef` element. The value must be the `id` of the referred entity described above.

If filter objects (all read requests usually have a `FilterType` payload object) the element is called `id` for the searched entity and have the `ref` suffix if the referred entity (like `customerRef`) is a further

part of the search filter.

5.9.1. Scope of the IDs

Every entity which is sent to the *API* must have an ID. When you send new entities to the *API*, e.g. you create a new offer, you have to select an ID for that entity. You need this ID to refer to that entity later, e.g. if you want to update or delete it.

You are free to select any ID you like, as long as it does not exceed the length restriction of the web service element in which it is contained. IDs must be unique within in their domain (truck, cargo, contact and customer).

Customer entities are a special case, because it is not possible to create a new customer record by means of a web service request. Therefore, TIMOCOM will create it and you have to tell us the ID of your customer you want to use for the *API*. See section From Testing to Production.

These IDs are abstracted from the internal IDs TIMOCOM uses for the corresponding entities and may be an abstraction of your own local IDs as well. However you may choose to use your local IDs as entity ID for the *API*.

5.9.2. Public Id

A `publicId` is the entity's primary key generated by TIMOCOM.

- The offer's publicId is returned if you search for offers, it may then be used in a cargo offer (resp. loading space) offer request.
- The customer's publicId is the TimoComID.

5.10. Closed User Group

The *Closed User Group* (briefly *CUG*, or in German: *Geschlossene Benutzergruppe* or briefly *GBG*) is a concept of grouping TIMOCOM customers for offer publication constraints purposes. The common use case is that an offer with CUG settings will be published exclusively within the CUG in the first place. This means that it is not visible to any TIMOCOM customer which is not member of the set CUG. After a set duration of time (maybe never) it will be automatically visible to all TIMOCOM customers unless it has not been withdrawn before.

If your company has a valid CUG contract with TIMOCOM you are able to send via the *API* all Closed User Group settings with the submitted offer.

For further information see Closed User Group Objects and Their Fields.

5.11. Reference Data

Reference data (also see Wikipedia definition) used within the *API* v2 are now represented by strings i.e. soft enumeration types.

5.11.1. Reference Data Types

The following reference data are defined:

- Additional Cargo Information
- Closed User Group Publication Type
- Contact channel to be shown in offer detail
- Country Codes ISO 3166-2
- Currency Codes ISO 4217
- Form of Address (Person)
- Language Codes ISO 639-1
- Loading Type
- Logistics Document Types
- Message Key
- Message Level
- Response Status
- Status of a price proposal (read only)
- Vehicle Body (Vehicle Property Category)
- Vehicle Body Property (Vehicle Property Category)
- Vehicle Equipment (Vehicle Property Category)
- Vehicle Load Securing (Vehicle Property Category)
- Vehicle Property Category
- Vehicle Swap Body (Vehicle Property Category)
- Vehicle Type (Vehicle Property Category)

For more detailed information see Reference Data Index.

5.11.2. Soft Enumerations

A so called soft enum is an enumeration type that allows extensions without changing the *API*. API changes require a new version on *API Freight Exchange* side and thus an additional administration effort.

In order to reduce the frequency of API changes all enumeration types are defined as a String data type by the following manner:

```
<xsd:simpleType name="TCCountryISOEnumType"> ①
    <xsd:annotation>
        <xsd:documentation xml:lang="en">Country Codes</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="AT">
            <xsd:annotation>
                <xsd:documentation xml:lang="de">Österreich</xsd:documentation>
                <xsd:documentation xml:lang="en">Austria</xsd:documentation>
                <xsd:documentation xml:lang="fr">Autriche</xsd:documentation>
            </xsd:annotation>
        </xsd:enumeration>
        <xsd:enumeration value="ES">
            <xsd:annotation>
                <xsd:documentation xml:lang="de">Spanien</xsd:documentation>
                <xsd:documentation xml:lang="en">Spain</xsd:documentation>
                <xsd:documentation xml:lang="fr">Espagne</xsd:documentation>
            </xsd:annotation>
        </xsd:enumeration>
        <!-- further types omitted here-->
    </xsd:restriction>
<xsd:simpleType>

<xsd:simpleType name="TCCountryISO">  ②
    <xsd:union memberTypes="TCCountryISOEnumType xsd:string"/>
</xsd:simpleType>

<xsd:complexType name="AddressType">
    <xsd:sequence>
        <xsd:element name="country" type="TCCountryISO"/>  ③
        <!-- further types omitted here-->
    </xsd:sequence>
</xsd:complexType>
```

① The ‘real’ enumeration type

② The soft enum wrapping allowing the ‘real’ enumeration type and any string

③ Usage of the soft enum just as any other type in the XSD

This XML schema snippet results in the following Java code snippet

```
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "AddressType", propOrder = {
    "country",
    "postalCode",
    "city",
    "geocoded"
})
@XmlSeeAlso({
    PostalAddressType.class,
    AreaType.class
})
public class AddressType {

    @XmlElement(required = true)
    @XmlSchemaType(name = "anySimpleType")
    protected String country;
    @XmlJavaTypeAdapter(CollapsedStringAdapter.class)
    @XmlSchemaType(name = "token")
    protected String postalCode;
    @XmlJavaTypeAdapter(CollapsedStringAdapter.class)
    @XmlSchemaType(name = "token")
    protected String city;
    protected Boolean geocoded;

    /**
     * Getters and setters omitted.
     */
}
```

You may then use enumeration type from the XML schema by converting it to a string or set the value directly from a string:

```
final AddressType cargoOffer = new AddressType();
cargoOffer.setCountry(TCCountryISOEnumType.AT.value());

// alternativly
cargoOffer.setCountry("AT");
```

5.11.3. Incompleteness of Enumeration Values in the XML Schema

Be aware that you are able to handle unknown enumeration (i.e. String) values in soft enumeration types. If a new enumeration value (e.g. a new country code or vehicle body) is introduced on the *API* side, most probably there will not be a new release of the *API Freight Exchange* Web Service Interface API.

5.11.4. Reference Data Query

The *API* provides an endpoint for querying all currently valid soft enumeration values.

You may fire a `GetReferencedDataRequest` for loading all values grouped by type with translations in German and English language.

Further languages may be supported in the future. If you are interested ask your TIMOCOM contact person.

These reference data have a versioning of themselves and all supported versions are contained in the *release notes* section of the response.

For detailed information about the API method see Reference Data Endpoint.

Chapter 6. Response Handling

6.1. SOAP-Fault

A SOAP-fault is a SOAP standard response if something went wrong on server side and the server is not able or does not want to handle the request within its regular request processing. In general the *API* tries to respond with an appropriate response type defined in the XSD.

Common scenarios for a SOAP-fault response are a broken client XML, or the client sent a request with the wrong SOAP version.

Example 1. SOAP Fault example if XML is broken

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
   <env:Body>
      <env:Fault>
         <env:Code>
            <env:Value>env:Sender</env:Value>
         </env:Code>
         <env:Reason>
            <env:Text xml:lang="en">Invalid XML due to following cause: Invalid
XML due to following cause: Could not parse XML; nested exception is
org.xml.sax.SAXParseException; lineNumber: 10; columnNumber: 17; Content is not
allowed in trailing section.</env:Text>
         </env:Reason>
      </env:Fault>
   </env:Body>
</env:Envelope>
```

6.2. Status

Each response contains a `status` element indicating successful processing.

6.2.1. Successful processing

`OK`

for `GetEntityResponseType` if the object could be found and for `FindSomethingResponseType`

`NOT_FOUND`

for `GetEntityResponseType` if the object could not be found

`DELETED`

for `DeleteEntityResponseType` if the entity could be deleted

`STORED`

for `StoreEntityResponseType` if the entity could be stored (created or updated)

6.2.2. Rejected storing processing

`UNPROCESSABLE_ENTITY`

for store or delete requests if server side validation detected incomplete or illegal field values

`CONFLICT`

for two or more store or delete requests which were processed at the same time but interfering each other

6.2.3. General errors during processing

We recommend that you raise an exception for the following status values.

`UNAUTHORIZED`

for any request if the client is not authorized for the operation. E.g.

1. The referred customer is unknown or not authorized (anymore)
2. The referred contact person is not registered
3. The corresponding user of the referred contact person locked
4. The contact person cannot be modified because the corresponding user is a system user.

`BAD_REQUEST`

for any request if the request is broken (e.g. illegal object structure or XML)

1. The payload is of an illegal (not acceptable) type
2. Constraints are violated in a read request, e.g. a missing or invalid mandatory id

`INTERNAL_SERVER_ERROR`

for any request if a server side problem prevented a successful processing

6.3. Message Type

Each `TCConnectResponseType` contains a list of messages (of type `MessageType`). Such a message is an object which contains information about the request processing. Sometimes a message's level is just `INFO` for possible logging purpose on client side. Sometimes a message's level is `WARN` for at least recommended logging on client side. And sometimes a message's level is `ERROR` if it represents a field validation constraint violation.

Each message type contains a message key which describes the message and may be used as a look-up key for a translated message to be logged in the log file or even shown in any kind of user interface.

Please note that you may query all translations at runtime in English and German by the Reference Data Endpoint.

6.4. Validation

The *API* validates all constraints on fields or across fields before storing a submitted entity. If the request is rejected due to validation failure the service will response with the status `UNPROCESSABLE_ENTITY` and at least one message with the appropriate message key.

It is not guaranteed that the response contains all messages of level `ERROR` if your sent entity violates more than one constraint. This is due to performance reason and multiple phases of validation on server side.

Example 2. Validation Failure

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
   <env:Header/>
   <env:Body>
      <con:StoreContactPersonResponse
         xmlns:con="http://webservice.timocom.com/schema/connect/v2">
         <con:requestRef>ahi5hb4afi</con:requestRef>
         <con:status>UNPROCESSABLE_ENTITY</con:status>
         <con:messages>
            <con:message>
               <con:messageLevel>ERROR</con:messageLevel>
               <con:messageKey>MISSING_FIRST_NAME</con:messageKey>
               <con:propertyPath>firstName</con:propertyPath>
            </con:message>
         </con:messages>
      </con:StoreContactPersonResponse>
   </env:Body>
</env:Envelope>
```

Chapter 7. API Methods (Endpoints)

This chapter describes all WSDL operations.

7.1. Customer Endpoints

7.1.1. Getting a Customer By Its Primary Key

- WSDL operation: `GetCustomer`
- Request type: `GetCustomerRequest`
- Request payload (filter) type: `IdFilterType`
- Response type: `GetCustomerResponse`
- Response payload type: `CustomerType`
- Successful response status: `OK`

Example 3. GetCustomerRequest

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
               xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:GetCustomerRequest>
         <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
         <con:payload xsi:type="con:IdFilterType"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
           <con:id>CUSTOMER_REF</con:id>
         </con:payload>
      </con:GetCustomerRequest>
   </soap:Body>
</soap:Envelope>
```

Example 4. GetCustomerResponse

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
   <env:Header/>
   <env:Body>
      <con:GetCustomerResponse
            xmlns:con="http://webservice.timocom.com/schema/connect/v2">
         <con:requestRef>7nqkf0vzmr</con:requestRef>
         <con:status>OK</con:status>
         <con:messages/>
         <con:payload xsi:type="con:CustomerType"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <con:id>99</con:id>
            <con:creationDateTime>2017-05-31T08:47:30.875Z</con:creationDateTime>
            <con:name>My Company Name</con:name>
            <con:phone>11122333</con:phone>
            <con:fax>444555666</con:fax>
            <con:companyAddress>
               <con:country>DE</con:country>
               <con:postalCode>40599</con:postalCode>
               <con:city>Berlin</con:city>
               <con:streetOrPostbox>Unter den Linden 2</con:streetOrPostbox>
            </con:companyAddress>
            <con:postalAddress>
               <con:country>DE</con:country>
               <con:streetOrPostbox>Postfach 8 15</con:streetOrPostbox>
            </con:postalAddress>
            <con:taxId>777/8888/9999</con:taxId>
         </con:payload>
      </con:GetCustomerResponse>
   </env:Body>
</env:Envelope>
```

7.1.2. Finding Customer Keys of Your Group

- WSDL operation: `FindCustomerKeys`
- Request type: `FindCustomerKeysRequest`
- Request payload (filter) type: `FilterType` or omitted
- Response type: `FindCustomerKeysResponse`
- Response payload type: `EntityKeyListType`
- Successful response status: `OK`

Example 5. FindCustomerKeysRequest

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:FindCustomerKeysRequest version="2.6.0">
         <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
         <con:payload xsi:type="con:FilterType"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      </con:FindCustomerKeysRequest>
   </soap:Body>
</soap:Envelope>
```

Example 6. FindCustomerKeysResponse.payload

```
<con:payload xsi:type="con:EntityKeyListType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:key>
      <con:id>16</con:id>
   </con:key>
   <con:key>
      <con:id>99</con:id>
   </con:key>
   <!-- further keys omitted -->
</con:payload>
```

7.1.3. Finding Customers of Your Group

- WSDL operation: `FindCustomers`
- Request type: `FindCustomersRequest`
- Request payload (filter) type: `FilterType` or omitted
- Response type: `FindCustomersResponse`
- Response payload type: `CustomerListType`
- Successful response status: `OK`

Example 7. FindCustomersRequest

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
       xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:FindCustomersRequest version="2.6.0">
         <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
         <con:payload xsi:type="con:FilterType"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      </con:FindCustomersRequest>
   </soap:Body>
</soap:Envelope>
```

Example 8. FindCustomersResponse.payload

```
<con:payload xsi:type="con:CustomerListType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
      <con:id>99</con:id>
      <con:creationDateTime>2017-05-31T08:47:30.875Z</con:creationDateTime>
      <con:name>My Company Name</con:name>
      <con:phone>11122333</con:phone>
      <con:fax>444555666</con:fax>
      <con:companyAddress>
         <con:country>DE</con:country>
         <con:postalCode>40599</con:postalCode>
         <con:city>Berlin</con:city>
         <con:streetOrPostbox>Unter den Linden 2</con:streetOrPostbox>
      </con:companyAddress>
      <con:postalAddress>
         <con:country>DE</con:country>
         <con:streetOrPostbox>Postfach 8 15</con:streetOrPostbox>
      </con:postalAddress>
      <con:taxId>777/8888/9999</con:taxId>
   </con:entity>
   <!-- further entities omitted -->
</con:payload>
```

7.1.4. Storing or Deletion of a Customer

This functionality is not available for the *API*.

7.2. Contact Person Endpoints

7.2.1. Getting a Contact Person By Its Primary Key

- WSDL operation: `GetContactPerson`
- Request type: `GetContactPersonRequest`
- Request payload (filter) type: `CustomerReferencingEntityFilterType`
- Response type: `GetContactPersonResponse`
- Response payload type: `ContactPersonType`
- Successful response status: `OK`

Example 9. GetContactPersonRequest

```
<con:GetContactPersonRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:CustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:id>CONTACT_REF</con:id>
   </con:payload>
</con:GetContactPersonRequest>
```

Example 10. GetContactPersonResponse.payload

```
<con:payload
   xsi:type="con:ContactPersonType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:id>CONTACT_REF</con:id>
   <con:creationDateTime>2017-05-31T12:41:54.808Z</con:creationDateTime>
   <con:customer xsi:type="con:CustomerRefWrapperType">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
   </con:customer>
   <con:lastName>Doe</con:lastName>
   <con:firstName>John</con:firstName>
   <con:title>MR</con:title>
   <con:email>john.doe@acme.com</con:email>
   <con:languages>
      <con:language>de</con:language>
      <con:language>es</con:language>
   </con:languages>
   <con:position>Senior Dispatcher</con:position>
   <con:phone>111222333444</con:phone>
   <con:phonePrivate/>
   <con:phoneMobile/>
   <con:fax/>
   <con:faxPrivate/>
</con:payload>
```

7.2.2. Finding Contact Person Keys

- WSDL operation: `FindContactPersonKeys`
- Request type: `FindContactPersonKeysRequest`
- Request payload (filter) type: `ExtendedCustomerReferencingEntityFilterType`
- Response type: `FindContactPersonKeysResponse`
- Response payload type: `CustomerReferencingEntityKeyListType`
- Successful response status: `OK`

Example 11. FindContactPersonKeysRequest

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:FindContactPersonKeysRequest version="2.6.0">
        <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
        <con:payload
          xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <con:customerRef>99</con:customerRef>
            <con:createdAfter>2017-08-16T15:00:00Z</con:createdAfter>
            <con:createdBefore>2017-08-16T17:00:00Z</con:createdBefore>
        </con:payload>
      </con:FindContactPersonKeysRequest>
   </soap:Body>
</soap:Envelope>
```

Example 12. FindContactPersonKeysResponse.payload

```
<con:payload
   xsi:type="con:CustomerReferencingEntityKeyListType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:key>
      <con:id>AB-97123</con:id>
      <con:customerRef>99</con:customerRef>
   </con:key>
   <!-- further keys omitted -->
</con:payload>
```

7.2.3. Finding Contact Persons

- WSDL operation: `FindContactPersons`
- Request type: `FindContactPersonsRequest`
- Request payload (filter) type: `ExtendedCustomerReferencingEntityFilterType`
- Response type: `FindContactPersonsResponse`
- Response payload type: `ContactPersonListType`
- Successful response status: `OK`

Example 13. FindContactPersonsRequest

```
<con:FindContactPersonsRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:createdAfter>2017-07-26T15:00:00Z</con:createdAfter>
      <con:createdBefore>2017-08-16T17:00:00Z</con:createdBefore>
   </con:payload>
</con:FindContactPersonsRequest>
```

Example 14. FindContactPersonsResponse.payload

```
<con:payload xsi:type="con:ContactPersonListType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
      <con:id>CONTACT_REF</con:id>
      <con:creationDateTime>2017-05-31T12:41:54.808Z</con:creationDateTime>
      <con:customer xsi:type="con:CustomerRefWrapperType">
         <con:customerRef>CUSTOMER_REF</con:customerRef>
      </con:customer>
      <con:lastName>Doe</con:lastName>
      <con:firstName>John</con:firstName>
      <con:title>MR</con:title>
      <con:email>john.doe@acme.com</con:email>
      <con:languages>
         <con:language>de</con:language>
         <con:language>es</con:language>
      </con:languages>
      <con:position>Senior Dispatcher</con:position>
      <con:phone>111222333444</con:phone>
      <con:phonePrivate/>
      <con:phoneMobile/>
      <con:fax/>
      <con:faxPrivate/>
   </con:entity>
   <con:entity>
      <!-- fields omitted here-->
   </con:entity>
</con:payload>
```

7.2.4. Storing a Contact Person

- WSDL operation: `StoreContactPerson`

- Request type: `StoreContactPersonRequest`
- Request payload type: `ContactPersonType`
- Response type: `StoreContactPersonResponse`
- Response payload type: `null`
- Successful response status: `STORED`
- Possible Message keys in response messages list:
  - `CONTACT_NOT_MODIFIABLE`
  - `CONTACT_USER_LOCKED`
  - `CUSTOMER_NOT_REGISTERED`
  - `INVALID_EMAIL`
  - `INVALID_FAX_NUMBER`
  - `INVALID_ID_VALUE`
  - `INVALID_MOBILE_NUMBER`
  - `INVALID_PHONE_NUMBER`
  - `MAX_LENGTH_EMAIL_EXCEEDED`
  - `MAX_LENGTH_FIRST_NAME_EXCEEDED`
  - `MAX_LENGTH_LAST_NAME_EXCEEDED`
  - `MISSING_CUSTOMER_REF`
  - `MISSING_FIRST_NAME`
  - `MISSING_LAST_NAME`
  - `MISSING_PHONE_NUMBER`
  - `MISSING_TITLE`
  - `UNKNOWN_LANGUAGE_CODE`
  - `UNKNOWN_TITLE`
  - `UPDATE_IGNORED`

Constraints

1. `firstName` and `lastName` are mandatory fields
2. You should provide `phone`, `phoneMobile` or `email`. This is not validated yet but may be a cross-field constraint in the future.

Example 15. StoreContactPersonRequest

```
<con:StoreContactPersonRequest version="2.6.0" xmlns:con=
"http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:ContactPersonType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:id>CONTACT_REF</con:id>
      <con:customer xsi:type="con:CustomerRefWrapperType">
         <con:customerRef>CUSTOMER_REF</con:customerRef>
      </con:customer>
      <con:lastName>Doe</con:lastName>
      <con:firstName>John</con:firstName>
      <con:title>MR</con:title>
      <con:email></con:email>
      <con:languages>
         <con:language>de</con:language>
         <con:language>es</con:language>
      </con:languages>
      <con:position></con:position>
      <con:phone>111222333</con:phone>
      <con:phonePrivate></con:phonePrivate>
      <con:phoneMobile></con:phoneMobile>
      <con:fax></con:fax>
      <con:faxPrivate></con:faxPrivate>
   </con:payload>
</con:StoreContactPersonRequest>
```

Example 16. StoreContactPersonResponse

```
<con:StoreContactPersonResponse
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hdltywlvzl</con:requestRef>
   <con:status>STORED</con:status>
   <con:messages/>
</con:StoreContactPersonResponse>
```

7.2.5. Deleting a Contact Person

- WSDL operation: `DeleteContactPerson`
- Request type: `DeleteContactPersonRequest`
- Request payload type: `CustomerReferencingEntityKeyType`
- Response type: `DeleteContactPersonResponse`
- Response payload type: `null`

- Successful response status: `DELETED`
- Possible Message keys in response messages list:
  - `ALREADY_DELETED`
  - `CONTACT_NOT_DELETABLE`
  - `CONTACT_USER_LOCKED`
  - `CUSTOMER_NOT_REGISTERED`

Example 17. DeleteContactPersonRequest

```
<con:DeleteContactPersonRequest version="2.6.0" xmlns:con=
"http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:CustomerReferencingEntityKeyType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
     <con:id>CONTACT_REF</con:id>
     <con:customerRef>CUSTOMER_REF</con:customerRef>
   </con:payload>
</con:DeleteContactPersonRequest>
```

Example 18. DeleteContactPersonResponse

```
<con:DeleteContactPersonResponse
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hebvbnti4l</con:requestRef>
   <con:status>DELETED</con:status>
   <con:messages/>
</con:DeleteContactPersonResponse>
```

7.3. Cargo Offer Endpoints

7.3.1. Getting a Cargo Offer By Its Primary Key

- WSDL operation: `GetCargoOffer`
- Request type: `GetCargoOfferRequest`
- Request payload (filter) type: `CustomerReferencingEntityFilterType`
- Response type: `GetCargoOfferResponse`
- Response payload type: `CargoOfferType`
- Successful response status: `OK`

Example 19. GetCargoOfferRequest

```
<con:GetCargoOfferRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:CustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:id>OFFER_REF</con:id>
   </con:payload>
</con:GetCargoOfferRequest>
```

Example 20. GetCargoOfferResponse.payload

```
<con:payload
  xsi:type="con:CargoOfferType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:id>OFFER_REF</con:id>
   <con:creationDateTime>2017-06-01T11:52:56Z</con:creationDateTime>
   <con:customer xsi:type="con:CustomerRefWrapperType">
     <con:customerRef>CUSTOMER_REF</con:customerRef>
   </con:customer>
   <con:contactPerson xsi:type="con:ContactRefWrapperType">
     <con:contactRef>CONTACT_REF</con:contactRef>
   </con:contactPerson>
   <con:publicRemark></con:publicRemark>
   <con:contactChannels>
       <con:contactChannel>EMAIL_ADDRESS</con:contactChannel>
       <con:contactChannel>PHONE_NUMBER</con:contactChannel>
   </con:contactChannels>
   <con:internalRemark></con:internalRemark>
   <con:trackable>false</con:trackable>
   <con:closedUserGroupSetting xsi:type="con:ResponseClosedUserGroupSettingType">
      <con:id>1000</con:id>
      <con:publicationType>EXTERNAL_LATER</con:publicationType>
      <con:remark></con:remark>
      <con:retentionTimeMinutes>22</con:retentionTimeMinutes>
      <con:publicationDate>2017-06-01T12:14:56Z</con:publicationDate>
   </con:closedUserGroupSetting>
   <con:vehicleProperties>
      <con:property>
         <con:category>VEHICLE_BODY</con:category>
         <con:values>
            <con:value>CURTAIN_SIDER</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_EQUIPMENT</con:category>
         <con:values>
            <con:value>ADR_EQUIPMENT_SET</con:value>
            <con:value>FORTY_FIVE_FEET_CONTAINER</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_TYPE</con:category>
         <con:values>
            <con:value>TRAILER</con:value>
         </con:values>
      </con:property>
   </con:vehicleProperties>
   <con:logisticsDocumentTypes>
      <con:documentType>GMP_CERTIFICATE</con:documentType>
```

```
    </con:logisticsDocumentTypes>
    <con:additionalInformationList>
       <con:additionalInformation>PART_LOAD</con:additionalInformation>
    </con:additionalInformationList>
    <con:freightDescription>stuff</con:freightDescription>
    <con:price>
       <con:amount>600.00</con:amount>
       <con:currency>USD</con:currency>
    </con:price>
    <con:lengthInMetres>13.6</con:lengthInMetres>
    <con:weightInTons>24.33</con:weightInTons>
    <con:loadingPlaces>
       <con:loadingPlace>
          <con:loadingType>LOADING</con:loadingType>
          <con:address>
             <con:country>FR</con:country>
             <con:postalCode>13281</con:postalCode>
             <con:city>Marseille</con:city>
             <con:geocoded>true</con:geocoded>
          </con:address>
          <con:earliestLoadingDate>2020-05-29</con:earliestLoadingDate>
          <con:latestLoadingDate>2020-06-01</con:latestLoadingDate>
          <con:startTime>09:00:00</con:startTime>
          <con:endTime>18:00:00</con:endTime>
       </con:loadingPlace>
       <con:loadingPlace>
          <con:loadingType>LOADING</con:loadingType>
       </con:loadingPlace>
       <con:loadingPlace>
          <con:loadingType>UNLOADING</con:loadingType>
          <con:address>
             <con:country>IT</con:country>
             <con:postalCode>50139</con:postalCode>
             <con:city>Fiorenze</con:city>
             <con:geocoded>true</con:geocoded>
          </con:address>
          <con:earliestLoadingDate>2020-05-30</con:earliestLoadingDate>
          <con:latestLoadingDate>2020-06-01</con:latestLoadingDate>
          <con:startTime>23:00:00</con:startTime>
          <con:endTime>23:58:00</con:endTime>
       </con:loadingPlace>
    </con:loadingPlaces>
</con:payload>
```

7.3.2. Finding Cargo Offer Keys

- WSDL operation: `FindCargoOfferKeys`
- Request type: `FindCargoOfferKeysRequest`

- Request payload (filter) type: `ExtendedCustomerReferencingEntityFilterType`
- Response type: `FindCargoOfferKeysResponse`
- Response payload type: `CustomerReferencingEntityKeyListType`
- Successful response status: `OK`

Example 21. FindCargoOfferKeysRequest

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:FindCargoOfferKeysRequest version="2.6.0">
        <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
        <con:payload
          xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <con:customerRef>99</con:customerRef>
            <con:createdAfter>2017-08-16T15:00:00Z</con:createdAfter>
            <con:createdBefore>2017-08-16T17:00:00Z</con:createdBefore>
        </con:payload>
      </con:FindCargoOfferKeysRequest>
   </soap:Body>
</soap:Envelope>
```

Example 22. FindCargoOfferKeysResponse.payload

```
<con:payload
   xsi:type="con:CustomerReferencingEntityKeyListType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:key>
      <con:id>CGO-97123</con:id>
      <con:customerRef>99</con:customerRef>
   </con:key>
   <!-- further keys omitted -->
</con:payload>
```

7.3.3. Finding Cargo Offers Inside of Your Customer Group

- WSDL operation: `FindCargoOffers`
- Request type: `FindCargoOffersRequest`
- Request payload (filter) type: `ExtendedCustomerReferencingEntityFilterType`
- Response type: `FindCargoOffersResponse`
- Response payload type: `CargoOfferListType`

- Successful response status: `OK`

Example 23. FindCargoOffersRequest

```
<con:FindCargoOffersRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:createdAfter>2017-07-26T15:00:00Z</con:createdAfter>
      <con:createdBefore>2017-07-27T17:00:00Z</con:createdBefore>
   </con:payload>
</con:FindCargoOffersRequest>
```

Example 24. FindCargoOffersResponse.payload

```
<con:payload xsi:type="con:CargoOfferListType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
      <con:id>OFFER_REF</con:id>
      <con:creationDateTime>2017-05-31T12:41:54.808Z</con:creationDateTime>
      <con:customer xsi:type="con:CustomerRefWrapperType">
        <con:customerRef>CUSTOMER_REF</con:customerRef>
      </con:customer>
      <!-- further fields omitted here, see CargoOfferType above -->
   </con:entity>
   <con:entity>
      <!-- fields omitted here-->
      <con:vehicleProperties>
      <!-- fields omitted here-->
      </con:vehicleProperties>
      <con:logisticsDocumentTypes>
          <con:documentType>GMP_CERTIFICATE</con:documentType>
      </con:logisticsDocumentTypes>
      <!-- fields omitted here-->
   </con:entity>
</con:payload>
```

7.3.4. Finding Cargo Offers Outside of Your Customer Group

This search is restricted to authorized customer groups, only.

- WSDL operation: `FindCargoOffers`
- Request type: `FindCargoOffersRequest`

- Request payload (filter) type: `CargoOfferFilterType`
- Response type: `FindCargoOffersResponse`
- Response payload type: `CargoOfferListType`
- Successful response status: `OK`
- Possible Message keys in response messages list:
  - `DATE_OUT_OF_RANGE`
  - `DETACHED_DELETED_ENTITIES`
  - `INVALID_AREA_SIZE`
  - `INVALID_DATE_INTERVAL`
  - `INVALID_ENDPOINT`
  - `INVALID_LOWER_BOUND_DATE_TIME`
  - `INVALID_NEGATIVE_DECIMAL_VALUE`
  - `INVALID_POSTAL_CODE`
  - `INVALID_QUERY_DATE_TIME`
  - `INVALID_QUERY_INTERVAL`
  - `INVALID_RESULT_SIZE`
  - `INVALID_SORTING`
  - `MAX_NUMBER_COUNTRY_SEARCH_LINES_EXCEEDED`
  - `MAX_NUMBER_DATES_EXCEEDED`
  - `MAX_NUMBER_POSTAL_CODES_EXCEEDED`
  - `MAX_NUMBER_VEHICLE_BODIES_EXCEEDED`
  - `MAX_NUMBER_VEHICLE_TYPES_EXCEEDED`
  - `MISSING_AREA_SIZE`
  - `MISSING_COUNTRY_CODE`
  - `MISSING_DATE`
  - `MISSING_DESTINATION_LOCATION`
  - `MISSING_LOCATION_SEARCH_CHOICE`
  - `MISSING_QUERY_DATE_TIME`
  - `MISSING_START_LOCATION`
  - `MUTUALLY_EXCLUSIVE_DATE_CHOICES`
  - `RESET_QUERY_DATE_TIME`
  - `UNAUTHORIZED_SEARCH_FILTER`
  - `UNKNOWN_COUNTRY_CODE`
  - `UNKNOWN_VEHICLE_BODY`
  - `UNKNOWN_VEHICLE_TYPE`

- UNSUCCESSFUL_GEOCODING_FOR_AREA_SEARCH
- VEHICLE_PROPERTY_VALUE_BACKWARDS_MAPPED

Example 25. FindCargoOffersRequest Outside Search

```
<con:FindCargoOffersRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:CargoOfferFilterType"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:updatedAfterDateTime>2020-05-
01T01:00:00.123Z</con:updatedAfterDateTime>
      <con:queryDateTime>2020-05-01T12:00:00.123Z</con:queryDateTime>
      <con:firstResult>0</con:firstResult>
      <con:maxResults>10</con:maxResults>
      <con:sortings>
         <con:sorting>
            <con:field>creationDateTime</con:field>
            <con:ascending>false</con:ascending>
         </con:sorting>
      </con:sortings>
      <con:date>
        <con:dateInterval>  ①
           <con:start>2020-05-01</con:start>
           <con:end>2020-05-15</con:end>
        </con:dateInterval>
      </con:date>
      <con:startLocation>
         <con:countrySearch> ②
            <con:searchLine>
               <con:country>FR</con:country>
               <con:postalCodes>
                  <con:postalCode>13</con:postalCode>
                  <con:postalCode>14</con:postalCode>
                  <con:postalCode>2</con:postalCode>
               </con:postalCodes>
            </con:searchLine>
            <con:searchLine>
               <con:country>ES</con:country>
            </con:searchLine>
         </con:countrySearch>
      </con:startLocation>
      <con:destinationLocation>
         <con:areaSearch>   ③
            <con:country>SE</con:country>
            <con:postalCode>211 11</con:postalCode>
            <con:city>Malmö</con:city>
            <con:areaInKilometres>45</con:areaInKilometres>
         </con:areaSearch>
      </con:destinationLocation>
      <con:vehicleProperties>
         <con:property>
            <con:category>VEHICLE_EQUIPMENT</con:category>
            <con:values>
```

```
                <con:value>ADR_EQUIPMENT_SET</con:value>
                <con:value>ESCORT_VEHICLE_TYPE_3</con:value>
             </con:values>
          </con:property>
          <con:property>
             <con:category>VEHICLE_SWAP_BODY</con:category>
             <con:values>
                <con:value>FORTY_FEET_CONTAINER</con:value>
             </con:values>
          </con:property>
          <con:property>
             <con:category>VEHICLE_BODY</con:category>
             <con:values>
                <con:value>CURTAIN_SIDER</con:value>
                <con:value>VAN_CAR</con:value>
             </con:values>
          </con:property>
          <con:property>
             <con:category>VEHICLE_TYPE</con:category>
             <con:values>
                <con:value>TRAILER</con:value>
             </con:values>
          </con:property>
       </con:vehicleProperties>
    </con:payload>
</con:FindCargoOffersRequest>
```

① Instead of a date interval it is possible to provide up to 5 individual dates, see FindTruckOffersRequest example.

② The `countrySearch` in the `startLocation` means that we are looking for freight offers from France with postal codes starting with '13', '14' or '2' or from Spain at any location.

③ The `areaSearch` in the `destinationLocation` means that we are looking for freight offers to a destination within an area of 45 km around 211 11 Malmö, Sweden.

Example 26. FindCargoOffersResponse.payload Outside Search

```
<con:payload xsi:type="con:CargoOfferListType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
     <con:publicId>MASpTY6_RI2zqJ5Ox_7f4w</con:publicId>
     <con:creationDateTime>2018-03-15T13:42:32Z</con:creationDateTime>
     <con:customer xsi:type="con:CustomerDetailWrapperType">   ①
        <con:detail>
           <con:publicId>104</con:publicId>
           <con:name>Timotrans International GmbH &amp;amp; Co. KG</con:name>
           <con:phone>+49 211 740 00 -0</con:phone> ③
           <con:fax>+49 211 740 00 -47</con:fax>  ③
           <con:companyAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1/con:streetOrPostbox>
           </con:companyAddress>
           <con:postalAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1</con:streetOrPostbox>
           </con:postalAddress>
        <con:taxId>DE272039727</con:taxId>
        </con:detail>
      </con:customer>
      <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">   ②
        <con:detail>
           <con:lastName>Doe</con:lastName>
           <con:firstName>John</con:firstName>
           <con:title>MR</con:title>
           <con:email>john.doe@acme.com</con:email>   ③
           <con:languages>
              <con:language>de</con:language>
              <con:language>es</con:language>
           </con:languages>
           <con:position></con:position>
           <con:phone>111222333</con:phone>  ③
           <con:phonePrivate></con:phonePrivate> ③
           <con:phoneMobile></con:phoneMobile> ③
           <con:fax></con:fax>    ③
           <con:faxPrivate></con:faxPrivate>  ③
        </con:detail>
      </con:contactPerson>
      <!-- further fields omitted here, see CargoOfferType above -->
      <con:deepLink>
https://my.timocom.com/fakedeeplink/PBrO34gdScmJ2kVfNNTnKA</con:deepLink>
      <!-- fields omitted here, see CargoOfferType above -->
```

```
      <con:publicRemark>Public remark</con:publicRemark>
      <!-- fields omitted here, see CargoOfferType above -->
      <con:loadingPlaces>
         <!-- fields omitted here, see CargoOfferType above -->
      </con:loadingPlaces>
      <con:distanceInKilometres>145</con:distanceInKilometres>
      <!-- further fields omitted here, see CargoOfferType above -->
   </con:entity>
   <con:entity>
      <!-- fields omitted here-->
   </con:entity>
</con:payload>
```

① You need a dedicated authorization if you want to see the customer details of each offer.

② You need a dedicated authorization if you want to see the contact person details of each offer.

③ Field is filled according to the contact channel setting for this offer.

7.3.5. Looking Up Cargo Offers Outside of Your Customer Group

This lookup is restricted to authorized customer groups, only.

- WSDL operation: `LookupCargoOffers`
- Request type: `LookupCargoOffersRequest`
- Request payload (filter) type: `PublicIdsFilterType`
- Response type: `LookupCargoOffersResponse`
- Response payload type: `CargoOfferListType`
- Successful response status: `OK`
- Possible Message keys in response messages list:
  - `INVALID_ENDPOINT`
  - `INVALID_ID_VALUE`
  - `MAX_NUMBER_IDS_EXCEEDED`
  - `MISSING_ID_VALUE`

Example 27. LookupCargoOffersRequest Outside Search

```
<con:LookupCargoOffersRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:PublicIdsFilterType"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
       <con:publicIds>
          <con:publicId>timocom_public_offer_id_1</con:publicId>
          <con:publicId>timocom_public_offer_id_2</con:publicId>
       </con:publicIds>
   </con:payload>
</con:LookupCargoOffersRequest>
```

Example 28. LookupCargoOffersResponse.payload Outside Search

```
<con:payload xsi:type="con:CargoOfferListType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
     <con:publicId>timocom_public_offer_id_1</con:publicId>
     <con:creationDateTime>2018-03-15T13:42:32Z</con:creationDateTime>
     <con:customer xsi:type="con:CustomerDetailWrapperType">   ①
        <con:detail>
           <con:publicId>104</con:publicId>
           <con:name>Timotrans International GmbH &amp;amp; Co. KG</con:name>
           <con:phone>+49 211 740 00 -0</con:phone> ③
           <con:fax>+49 211 740 00 -47</con:fax>  ③
           <con:companyAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1/con:streetOrPostbox>
           </con:companyAddress>
           <con:postalAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1</con:streetOrPostbox>
           </con:postalAddress>
        <con:taxId>DE272039727</con:taxId>
        </con:detail>
      </con:customer>
      <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">   ②
        <con:detail>
           <con:lastName>Doe</con:lastName>
           <con:firstName>John</con:firstName>
           <con:title>MR</con:title>
           <con:email>john.doe@acme.com</con:email>   ③
           <con:languages>
              <con:language>de</con:language>
              <con:language>es</con:language>
           </con:languages>
           <con:position></con:position>
           <con:phone>111222333</con:phone>  ③
           <con:phonePrivate></con:phonePrivate> ③
           <con:phoneMobile></con:phoneMobile> ③
           <con:fax></con:fax>    ③
           <con:faxPrivate></con:faxPrivate>  ③
        </con:detail>
      </con:contactPerson>
      <con:publicRemark>Public remark</con:publicRemark>
      <!-- further fields omitted here, see CargoOfferType above -->
   </con:entity>
   <con:entity>
```

```
      <!-- 2nd offer w/ fields omitted here-->
   </con:entity>
</con:payload>
```

① You need a dedicated authorization if you want to see the customer details of each offer.

② You need a dedicated authorization if you want to see the contact person details of each offer.

③ Field is filled according to the contact channel setting for this offer.

7.3.6. Storing a Cargo Offer

- WSDL operation: `StoreCargoOffer`
- Request type: `StoreCargoOfferRequest`
- Request payload type: `CargoOfferType`
- Response type: `StoreCargoOfferResponse`
- Response payload type: `null`
- Successful response status: `STORED`
- Possible Message keys in response messages list:
  - `CONTACT_NOT_REGISTERED`
  - `CUG_CHANGED_TO_MANDATORY`
  - `CUG_IGNORED_AND_NO_MANDATORY_FOUND`
  - `CUG_PUBLICATION_TIME`
  - `CUG_RETENTION_TIME_CHANGED`
  - `CUSTOMER_NOT_REGISTERED`
  - `DATE_OUT_OF_RANGE`
  - `DEPRECATED_FIELD_OR_VALUE`
  - `INVALID_CHARACTERS`
  - `INVALID_CONTACT_REF`
  - `INVALID_DATE_FORMAT`
  - `INVALID_DESTINATION_LOADING_TYPE`
  - `INVALID_EMAIL`
  - `INVALID_FAX_NUMBER`
  - `INVALID_ID_VALUE`
  - `INVALID_MOBILE_NUMBER`
  - `INVALID_NEGATIVE_DECIMAL_VALUE`
  - `INVALID_PAYMENT_DUE_DAYS`
  - `INVALID_PHONE_NUMBER`

- INVALID_POSTAL_CODE
- INVALID_PRICE_AMOUNT
- INVALID_START_LOADING_TYPE
- LOADING_DATE_AFTER_UNLOADING_DATE
- MAX_AMOUNT_PRICE_EXCEEDED
- MAX_LENGTH_CITY_EXCEEDED
- MAX_LENGTH_EMAIL_EXCEEDED
- MAX_LENGTH_FIRST_NAME_EXCEEDED
- MAX_LENGTH_FREIGHT_DESCRIPTION_EXCEEDED
- MAX_LENGTH_LAST_NAME_EXCEEDED
- MAX_LENGTH_POSTCODE_EXCEEDED
- MAX_LENGTH_REMARKS_EXCEEDED
- MAX_NUMBER_LOADING_PLACES_EXCEEDED
- MISSING_BACKLINK
- MISSING_CITY
- MISSING_CONTACT
- MISSING_CONTACT_CHANNEL
- MISSING_COUNTRY_CODE
- MISSING_CUG_PUBLICATION_TYPE
- MISSING_CURRENCY_CODE
- MISSING_CUSTOMER_REF
- MISSING_DESTINATION
- MISSING_EMAIL
- MISSING_FAX_NUMBER
- MISSING_FIRST_NAME
- MISSING_FIRST_OR_LAST_COUNTRY_CODE
- MISSING_LAST_NAME
- MISSING_LOADING_PLACE
- MISSING_LOADING_TYPE
- MISSING_MOBILE_NUMBER
- MISSING_PHONE_NUMBER
- MISSING_PRICE_AMOUNT
- MISSING_START_DATE
- MISSING_TITLE
- MISSING_VEHICLE_BODY

- MISSING_VEHICLE_TYPE
- TOTAL_LENGTH_ABOVE_MAXIMUM
- TOTAL_LENGTH_BELOW_MINIMUM
- TOTAL_WEIGHT_ABOVE_MAXIMUM
- TOTAL_WEIGHT_BELOW_MINIMUM
- UNKNOWN_ADDITIONAL_INFORMATION
- UNKNOWN_CONTACT_CHANNEL
- UNKNOWN_COUNTRY_CODE
- UNKNOWN_CUG_PUBLICATION_TYPE
- UNKNOWN_CURRENCY_CODE
- UNKNOWN_LANGUAGE_CODE
- UNKNOWN_LOADING_TYPE
- UNKNOWN_TITLE
- UNKNOWN_VEHICLE_BODY
- UNKNOWN_VEHICLE_EQUIPMENT
- UNKNOWN_VEHICLE_PROPERTY_CATEGORY
- UNKNOWN_VEHICLE_TYPE
- UNSUCCESSFUL_GEOCODING
- UPDATE_IGNORED
- VEHICLE_PROPERTY_VALUE_MOVED

Example 29. StoreCargoOfferRequest

```
<con:StoreCargoOfferRequest
   version="2.6.0"
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:CargoOfferType"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:id>OFFER_REF</con:id>
      <con:customer xsi:type="con:CustomerRefWrapperType">
        <con:customerRef>CUSTOMER_REF</con:customerRef>
      </con:customer>
      <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">
         <con:detail xsi:type="con:ContactPersonType">  ①
            <con:lastName>Doe</con:lastName>
            <con:firstName>Jane</con:firstName>
            <con:title>MRS</con:title>
            <con:email>jane.doe@acme.com</con:email>
            <con:languages>
              <con:language>es</con:language>
              <con:language>de</con:language>
            </con:languages>
            <con:phone>014356765</con:phone>  ②
         </con:detail>
      </con:contactPerson>
      <con:publicRemark></con:publicRemark>
      <con:contactChannels>
         <con:contactChannel>EMAIL_ADDRESS</con:contactChannel>
         <con:contactChannel>PHONE_NUMBER</con:contactChannel>   ②
      </con:contactChannels>
      <con:internalRemark></con:internalRemark>
      <con:trackable>false</con:trackable>
      <con:closedUserGroupSetting
        xsi:type="con:RequestClosedUserGroupSettingType">
         <con:id>CUG_REF</con:id>
         <con:publicationType>EXTERNAL_LATER</con:publicationType>
         <con:remark></con:remark>
         <con:retentionTimeMinutes>22</con:retentionTimeMinutes>
         <con:resetRetentionTime>false</con:resetRetentionTime>
      </con:closedUserGroupSetting>
      <con:vehicleProperties>
         <con:property>
            <con:category>VEHICLE_BODY</con:category>
            <con:values>
               <con:value>CURTAIN_SIDER</con:value>
            </con:values>
         </con:property>
         <con:property>
            <con:category>VEHICLE_EQUIPMENT</con:category>
            <con:values>
```

```
            <con:value>ADR_EQUIPMENT_SET</con:value>
            <con:value>FIXED_LOADING_CRANE</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_SWAP_BODY</con:category>
         <con:values>
            <con:value>FORTY_FIVE_FEET_CONTAINER</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_LOAD_SECURING</con:category>
         <con:values>
            <con:value>TENSION_CHAIN</con:value>
            <con:value>ANTI_SLIP_MATS</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_BODY_PROPERTY</con:category>
         <con:values>
            <con:value>SLIDING_CURTAIN</con:value>
            <con:value>WIDENABLE</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_TYPE</con:category>
         <con:values>
            <con:value>TRAILER</con:value>
         </con:values>
      </con:property>
   </con:vehicleProperties>
   <con:logisticsDocumentTypes>
      <con:documentType>GMP_CERTIFICATE</con:documentType>
   </con:logisticsDocumentTypes>
   <con:additionalInformationList>
      <con:additionalInformation>PART_LOAD</con:additionalInformation>
   </con:additionalInformationList>
   <con:freightDescription></con:freightDescription>
   <con:price>
      <con:amount>700</con:amount>
      <con:currency>USD</con:currency>
   </con:price>
   <con:lengthInMetres>13.6</con:lengthInMetres>
   <con:weightInTons>24.33</con:weightInTons>
   <con:loadingPlaces>
      <con:loadingPlace>
         <con:loadingType>LOADING</con:loadingType>
         <con:address>
            <con:country>FR</con:country>
            <con:postalCode></con:postalCode>
            <con:city>Marseille</con:city>
```

```
            </con:address>
            <con:latestLoadingDate>2017-07-01</con:latestLoadingDate>
            <con:startTime>09:00:00</con:startTime>
            <con:endTime>18:00:00</con:endTime>
         </con:loadingPlace>
         <con:loadingPlace>
            <con:loadingType>LOADING</con:loadingType>
            <con:address/>
         </con:loadingPlace>
         <con:loadingPlace>
            <con:loadingType>UNLOADING</con:loadingType>
            <con:address>
               <con:country>DE</con:country>
               <con:postalCode>10115</con:postalCode>
               <con:city>Berlin</con:city>
            </con:address>
            <con:latestLoadingDate>2017-07-01</con:latestLoadingDate>
            <con:startTime>23:00:00</con:startTime>
            <con:endTime>23:58:00</con:endTime>
         </con:loadingPlace>
      </con:loadingPlaces>
   </con:payload>
</con:StoreCargoOfferRequest>
```

① This is an embedded contact person. It will be deleted together with this offer.

② The chosen contact channel PHONE_NUMBER requires a valid phone number in the embedded contact detail.

Example 30. StoreCargoOfferResponse

```
<con:StoreCargoOfferResponse
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hdltywlvzl</con:requestRef>
   <con:status>STORED</con:status>
   <con:messages>
      <con:message> ①
         <con:messageLevel>INFO</con:messageLevel>
         <con:messageKey>CUG_PUBLICATION_TIME</con:messageKey>
         <con:logMessage>Offer will become public at 2017-08-
01T14:24:09.113Z</con:logMessage>
      </con:message>
   </con:messages>
</con:StoreCargoOfferResponse>
```

① The publication time for all TIMOCOM customers of a Closed User Group offer will be returned as INFO level message. You can also get this publication date time by a GetCargoOfferRequest.

7.3.7. Deleting a Cargo Offer

- WSDL operation: `DeleteCargoOffer`
- Request type: `DeleteCargoOfferRequest`
- Request payload type: `CustomerReferencingEntityKeyType`
- Response type: `DeleteCargoOfferResponse`
- Response payload type: `null`
- Successful response status: `DELETED`
- Possible Message keys in response messages list:
  - `ALREADY_DELETED`
  - `CUSTOMER_NOT_REGISTERED`

Example 31. DeleteCargoOfferRequest

```
<con:DeleteCargoOfferRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
     xsi:type="con:CustomerReferencingEntityKeyType"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:id>OFFER_REF</con:id>
      <con:customerRef>CUSTOMER_REF</con:customerRef>
   </con:payload>
</con:DeleteCargoOfferRequest>
```

Example 32. DeleteCargoOfferResponse

```
<con:DeleteCargoOfferResponse
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hebvbnti4l</con:requestRef>
   <con:status>DELETED</con:status>
   <con:messages/>
</con:DeleteCargoOfferResponse>
```

7.4. Price Proposal Endpoints

7.4.1. Proposition: Getting a Price Proposal by Its Primary Key

- WSDL operation: `GetPriceProposal`
- Request type: `GetPriceProposalRequest`

- Request payload (filter) type: `CustomerReferencingEntityFilterType`
- Response type: `GetPriceProposalResponse`
- Response payload type: `PriceProposalType`
- Successful response status: `OK`

Example 33. GetPriceProposalRequest

```
<con:GetPriceProposalRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:CustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:publicId>PUBLIC_PRICE_PROPOSAL_ID</con:publicId>
   </con:payload>
</con:GetPriceProposalRequest>
```

Example 34. GetPriceProposalResponse.payload

```
<ns2:payload xsi:type="con:PriceProposalType" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance">
    <con:publicId>gAzx0U7jSQuu6p7kcPPaXA</con:publicId>
    <con:creationDateTime>2022-09-01T09:28:39.425555Z</con:creationDateTime>
    <con:customer xsi:type="con:CustomerRefWrapperType">
        <con:customerRef>CUSTOMER_REF</con:customerRef>
    </con:customer>
    <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">
        <con:detail>
            <con:lastName>Doe</con:lastName>
            <con:firstName>Jane</con:firstName>
            <con:title>MRS</con:title>
            <con:email>jane.doe@acme.com</con:email>
            <con:languages>
                <con:language>de</con:language>
                <con:language>es</con:language>
            </con:languages>
            <con:phone>014356765</con:phone>
        </con:detail>
    </con:contactPerson>
    <con:publicRemark>My Test Remark</con:publicRemark>
    <con:status>ACTIVE</con:status>
    <con:offerId>OFFER_ID</con:offerId>
    <con:price>
        <con:amount>750.00</con:amount>
        <con:currency>EUR</con:currency>
    </con:price>
    <con:expirationDateTime>2022-09-09T09:28:39.331298Z</con:expirationDateTime>
</con:payload>
```

7.4.2. Proposition: Storing a Price Proposal

- WSDL operation: `StorePriceProposal`
- Request type: `StorePriceProposalRequest`
- Request payload type: `PriceProposalType`
- Response type: `StorePriceProposalResponse`
- Response payload type: `EntityKeyType`
- Successful response status: `STORED`
- Possible Message keys in response messages list:
  - `CONTACT_NOT_REGISTERED`
  - `CUSTOMER_NOT_REGISTERED`
  - `DATE_OUT_OF_RANGE`

- INVALID_EMAIL
- INVALID_MOBILE_NUMBER
- INVALID_PHONE_NUMBER
- MAX_AMOUNT_PRICE_EXCEEDED
- MAX_LENGTH_EMAIL_EXCEEDED
- MAX_LENGTH_FIRST_NAME_EXCEEDED
- MAX_LENGTH_LAST_NAME_EXCEEDED
- MAX_LENGTH_REMARKS_EXCEEDED
- MISSING_CONTACT
- MISSING_CURRENCY_CODE
- MISSING_DATE_TIME
- MISSING_FIRST_NAME
- MISSING_LAST_NAME
- MISSING_PRICE
- MISSING_PRICE_AMOUNT
- MISSING_PUBLIC_OFFER_ID
- OFFER_NO_LONGER_PUBLISHED
- UNKNOWN_TITLE

Example 35. StorePriceProposalRequest

```
<con:StorePriceProposalRequest
    version="2.6.0"
    xmlns:con="http://webservice.timocom.com/schema/connect/v2">
    <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
    <con:payload xsi:type="con:PriceProposalType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <con:customer xsi:type="con:CustomerRefWrapperType">
            <con:customerRef>CUSTOMER_REF</con:customerRef>
        </con:customer>
        <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">
            <con:detail xsi:type="con:ContactPersonType">  ①
                    <con:lastName>Doe</con:lastName>
                    <con:firstName>Jane</con:firstName>
                    <con:title>MRS</con:title>
                    <con:email>jane.doe@acme.com</con:email>
                    <con:languages>
                        <con:language>es</con:language>
                        <con:language>de</con:language>
                    </con:languages>
                    <con:phone>014356765</con:phone>  ②
                    </con:detail>
        </con:contactPerson>
        <con:contactChannels>
            <con:contactChannel>EMAIL_ADDRESS</con:contactChannel>
            <con:contactChannel>PHONE_NUMBER</con:contactChannel>  ②
        </con:contactChannels>
        <con:publicRemark>My Test Remark</con:publicRemark>
        <con:offerId>OFFER_REF</con:offerId>
        <con:price>
            <con:amount>666</con:amount>
            <con:currency>USD</con:currency>
        </con:price>
        <con:expirationDateTime>2022-09-
09T09:28:39.331298Z</con:expirationDateTime>
    </con:payload>
</con:StorePriceProposalRequest>
```

① This is an embedded contact person. It will be deleted together with this price proposal.

② The chosen contact channel PHONE_NUMBER requires a valid phone number in the embedded contact detail.

Example 36. StorePriceProposalResponse

```
<con:StorePriceProposalResponse
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hdltywlvzl</con:requestRef>
   <con:status>STORED</con:status>
   <con:messages/>
   <con:payload xsi:type="con::EntityKeyType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
     <con:publicId>9tCPa227QOqikrTsx6icRw</con:publicId>  ①
   </con:payload>
</con:StorePriceProposalResponse>
```

① The public id of the stored price proposal will be returned within the payload of the StorePriceProposalResponse.

7.4.3. Deleting (Withdrawing) a Price Proposal

- WSDL operation: `DeletePriceProposal`
- Request type: `DeletePriceProposalRequest`
- Request payload type: `CustomerReferencingEntityKeyType`
- Response type: `DeletePriceProposalResponse`
- Response payload type: `null`
- Successful response status: `DELETED`
- Possible Message keys in response messages list:
  - `ALREADY_DELETED`
  - `PRICE_PROPOSAL_NO_LONGER_ACTIVE`
  - `UNAUTHORIZED_ACCESS`

Example 37. DeletePriceProposalRequest

```
<con:DeletePriceProposalRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
     xsi:type="con:CustomerReferencingEntityKeyType"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:publicId>PUBLIC_PRICE_PROPOSAL_ID</con:publicId>
   </con:payload>
</con:DeletePriceProposalRequest>
```

Example 38. DeletePriceProposalResponse

```
<con:DeletePriceProposalResponse
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hebvbnti4l</con:requestRef>
   <con:status>DELETED</con:status>
   <con:messages/>
</con:DeletePriceProposalResponse>
```

7.4.4. Receival: Finding Price Proposals for Your Cargo Offer

- WSDL operation: `FindPriceProposals`
- Request type: `FindPriceProposalsRequest`
- Request payload (filter) type: `PriceProposalByOfferFilterType`
- Response type: `FindPriceProposalsResponse`
- Response payload type: `PriceProposalListType`
- Successful response status: `OK`

Example 39. FindPriceProposalsRequest

```
<v2:FindPriceProposalsRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:PriceProposalByOfferFilterType" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:offerRef>OFFER_REF</con:offerRef>
   </con:payload>
</v2:FindPriceProposalsRequest>
```

Example 40. FindPriceProposalsResponse.payload

```
<con:payload xsi:type="con:PriceProposalListType" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance">
    <con:entity>
        <con:publicId>dwVAuv-PQ8-AtXchw6aE-A</con:publicId>
        <con:creationDateTime>2022-09-01T09:28:39.425555Z</con:creationDateTime>
        <con:customer xsi:type="con:CustomerDetailWrapperType">
            <con:detail>
                <con:publicId>104</con:publicId>
                <con:name>Timotrans International GmbH &amp;amp; Co. KG</con:name>
                <con:phone>+49 211 74000 0</con:phone>
                <con:fax>+49 211 74000 -47</con:fax>
                <con:companyAddress>
                    <con:country>DE</con:country>
                    <con:postalCode>40699</con:postalCode>
                    <con:city>Erkrath</con:city>
                    <con:streetOrPostbox>>Timocom-Platz 1</con:streetOrPostbox>
                </con:companyAddress>
                <con:taxId>DE272039727</con:taxId>
            </con:detail>
        </con:customer>
        <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">
            <con:detail>
                <con:lastName>Doe</con:lastName>
                <con:firstName>Jane</con:firstName>
                <con:title>MRS</con:title>
                <con:email>jane.doe@acme.com</con:email>
                <con:languages>
                    <con:language>de</con:language>
                    <con:language>es</con:language>
                </con:languages>
                <con:phone>014356765</con:phone>
            </ns2:detail>
        </ns2:contactPerson>
        <ns2:publicRemark>My Test Remark</ns2:publicRemark>
        <ns2:status>ACTIVE</ns2:status>
        <ns2:offerId>OFFER_REF</ns2:offerId>
        <ns2:price>
            <ns2:amount>666.00</ns2:amount>
            <ns2:currency>USD</ns2:currency>
        </ns2:price>
        <ns2:expirationDateTime>2022-09-
09T09:28:39.331298Z</ns2:expirationDateTime>
    </ns2:entity>
</ns2:payload>
```

7.4.5. Receival: Accepting a Price Proposal

- WSDL operation: `DeleteCargoOffer`
- Request type: `DeleteCargoOfferRequest`
- Request payload type: `DeleteCargoOfferPayloadType`
- Response type: `DeleteCargoOfferResponse`
- Response payload type: `null`
- Successful response status: `DELETED`
- Possible Message keys in response messages list:
  - `ALREADY_DELETED`
  - `CUSTOMER_NOT_REGISTERED`

Example 41. DeleteCargoOfferRequest

```
<con:DeleteCargoOfferRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
     xsi:type="con:DeleteCargoOfferPayloadType"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:id>OFFER_REF</con:id>
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:handling>ACCEPT</con:handling>
      <con:acceptedPriceProposalId>
PUBLIC_PRICE_PROPOSAL_ID</con:acceptedPriceProposalId>
   </con:payload>
</con:DeleteCargoOfferRequest>
```

Example 42. DeleteCargoOfferResponse

```
<con:DeleteCargoOfferResponse
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hebvbnti4l</con:requestRef>
   <con:status>DELETED</con:status>
   <con:messages/>
</con:DeleteCargoOfferResponse>
```

7.5. Loading Space Offer Endpoints

Truck offer is a synonym for loading space offer.

7.5.1. Getting a Loading Space Offer by Its Primary Key

- WSDL operation: `GetTruckOffer`
- Request type: `GetTruckOfferRequest`
- Request payload (filter) type: `CustomerReferencingEntityFilterType`
- Response type: `GetTruckOfferResponse`
- Response payload type: `TruckOfferType`
- Successful response status: `OK`

Example 43. GetTruckOfferRequest

```
<con:GetTruckOfferRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:CustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:id>OFFER_REF</con:id>
   </con:payload>
</con:GetTruckOfferRequest>
```

Example 44. GetTruckOfferResponse.payload

```
<con:payload
  xsi:type="con:TruckOfferType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:id>OFFER_REF</con:id>
   <con:creationDateTime>2020-06-01T11:52:56Z</con:creationDateTime>
   <con:customer xsi:type="con:CustomerRefWrapperType">
     <con:customerRef>CUSTOMER_REF</con:customerRef>
   </con:customer>
   <con:contactPerson xsi:type="con:ContactRefWrapperType">
     <con:contactRef>CONTACT_REF</con:contactRef>
   </con:contactPerson>
   <con:publicRemark></con:publicRemark>
   <con:contactChannels>
      <con:contactChannel>EMAIL_ADDRESS</con:contactChannel>
      <con:contactChannel>PHONE_NUMBER</con:contactChannel>
   </con:contactChannels>
   <con:internalRemark></con:internalRemark>
   <con:trackable>false</con:trackable>
   <con:closedUserGroupSetting xsi:type="con:ResponseClosedUserGroupSettingType">
      <con:id>1000</con:id>
      <con:publicationType>EXTERNAL_LATER</con:publicationType>
      <con:remark></con:remark>
      <con:retentionTimeMinutes>22</con:retentionTimeMinutes>
      <con:publicationDate>2020-06-01T12:14:56Z</con:publicationDate>
   </con:closedUserGroupSetting>
   <con:vehicleProperties>
      <con:property>
         <con:category>VEHICLE_BODY</con:category>
         <con:values>
            <con:value>CURTAIN_SIDER</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_EQUIPMENT</con:category>
         <con:values>
            <con:value>ADR_EQUIPMENT_SET</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_SWAP_BODY</con:category>
         <con:values>
            <con:value>FORTY_FIVE_FEET_CONTAINER</con:value>
         </con:values>
      </con:property>
      <con:property>
         <con:category>VEHICLE_TYPE</con:category>
         <con:values>
            <con:value>WAGGON_AND_DRAG</con:value>
```

```
            </con:values>
         </con:property>
      </con:vehicleProperties>
      <con:logisticsDocumentTypes>
         <con:documentType>GMP_CERTIFICATE</con:documentType>
      </con:logisticsDocumentTypes>
      <con:truckWeightInTons>2.3</con:truckWeightInTons>
      <con:trailerWeightInTons>1.2</con:trailerWeightInTons>
      <con:truckLengthInMetres>5.22</con:truckLengthInMetres>
      <con:trailerLengthInMetres>2.34</con:trailerLengthInMetres>
      <con:startDate>2020-08-01</con:startDate>
      <con:start>
         <con:country>FR</con:country>
         <con:postalCode>13281</con:postalCode>
         <con:city>Marseille</con:city>
         <con:geocoded>true</con:geocoded>
      </con:start>
      <con:destination>
         <con:area>
            <con:country>IT</con:country>
            <con:postalCode>50139</con:postalCode>
            <con:city>Fiorenze</con:city>
            <con:geocoded>true</con:geocoded>
            <con:areaInKilometres>50</con:areaInKilometres>
         </con:area>
      </con:destination>
   </con:payload>
```

7.5.2. Finding Truck Offer Keys

- WSDL operation: `FindTruckOfferKeys`
- Request type: `FindTruckOfferKeysRequest`
- Request payload (filter) type: `ExtendedCustomerReferencingEntityFilterType`
- Response type: `FindTruckOfferKeysResponse`
- Response payload type: `CustomerReferencingEntityKeyListType`
- Successful response status: `OK`

Example 45. FindTruckOfferKeysRequest

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:FindTruckOfferKeysRequest version="2.6.0">
        <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
        <con:payload
          xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <con:customerRef>99</con:customerRef>
            <con:createdAfter>2017-08-16T15:00:00Z</con:createdAfter>
            <con:createdBefore>2017-08-16T17:00:00Z</con:createdBefore>
        </con:payload>
      </con:FindTruckOfferKeysRequest>
   </soap:Body>
</soap:Envelope>
```

Example 46. FindTruckOfferKeysResponse.payload

```
<con:payload
   xsi:type="con:CustomerReferencingEntityKeyListType"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:key>
      <con:id>LSO-97123</con:id>
      <con:customerRef>99</con:customerRef>
   </con:key>
   <!-- further keys omitted -->
</con:payload>
```

7.5.3. Finding Loading Space Offers Inside of Your Customer Group

- WSDL operation: `FindTruckOffers`
- Request type: `FindTruckOffersRequest`
- Request payload (filter) type: `ExtendedCustomerReferencingEntityFilterType`
- Response type: `FindTruckOffersResponse`
- Response payload type: `TruckOfferListType`
- Successful response status: `OK`

Example 47. FindTruckOffersRequest

```
<con:FindTruckOffersRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
      xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:customerRef>CUSTOMER_REF</con:customerRef>
      <con:createdAfter>2020-07-26T15:00:00Z</con:createdAfter>
      <con:createdBefore>2020-07-27T17:00:00Z</con:createdBefore>
   </con:payload>
</con:FindTruckOffersRequest>
```

Example 48. FindTruckOffersResponse.payload

```
<con:payload xsi:type="con:TruckOfferListType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
      <con:id>OFFER_REF</con:id>
      <con:creationDateTime>2020-05-31T12:41:54.808Z</con:creationDateTime>
      <con:customer xsi:type="con:CustomerRefWrapperType">
        <con:customerRef>CUSTOMER_REF</con:customerRef>
      </con:customer>
      <!-- further fields omitted here, see TruckOfferType above -->
   </con:entity>
   <con:entity>
      <!-- fields omitted here-->
      <con:vehicleProperties>
      <!-- fields omitted here-->
      </con:vehicleProperties>
      <con:logisticsDocumentTypes>
          <con:documentType>GMP_CERTIFICATE</con:documentType>
      </con:logisticsDocumentTypes>
      <!-- fields omitted here-->
   </con:entity>
</con:payload>
```

7.5.4. Finding Loading Space Offers Outside of Your Customer Group

This search is restricted to authorized customer groups, only.

- WSDL operation: `FindTruckOffers`
- Request type: `FindTruckOffersRequest`
- Request payload (filter) type: `TruckOfferFilterType`

- Response type: `FindTruckOffersResponse`
- Response payload type: `TruckOfferListType`
- Successful response status: `OK`
- Possible Message keys in response messages list:
  - `DATE_OUT_OF_RANGE`
  - `DETACHED_DELETED_ENTITIES`
  - `INVALID_AREA_SIZE`
  - `INVALID_DATE_INTERVAL`
  - `INVALID_ENDPOINT`
  - `INVALID_LOWER_BOUND_DATE_TIME`
  - `INVALID_NEGATIVE_DECIMAL_VALUE`
  - `INVALID_POSTAL_CODE`
  - `INVALID_QUERY_DATE_TIME`
  - `INVALID_QUERY_INTERVAL`
  - `INVALID_RESULT_SIZE`
  - `INVALID_SORTING`
  - `MAX_NUMBER_COUNTRY_SEARCH_LINES_EXCEEDED`
  - `MAX_NUMBER_DATES_EXCEEDED`
  - `MAX_NUMBER_POSTAL_CODES_EXCEEDED`
  - `MAX_NUMBER_VEHICLE_BODIES_EXCEEDED`
  - `MAX_NUMBER_VEHICLE_TYPES_EXCEEDED`
  - `MISSING_AREA_SIZE`
  - `MISSING_COUNTRY_CODE`
  - `MISSING_DATE`
  - `MISSING_DESTINATION_LOCATION`
  - `MISSING_LOCATION_SEARCH_CHOICE`
  - `MISSING_QUERY_DATE_TIME`
  - `MISSING_START_LOCATION`
  - `MUTUALLY_EXCLUSIVE_DATE_CHOICES`
  - `RESET_QUERY_DATE_TIME`
  - `UNAUTHORIZED_SEARCH_FILTER`
  - `UNKNOWN_COUNTRY_CODE`
  - `UNKNOWN_VEHICLE_BODY`
  - `UNKNOWN_VEHICLE_TYPE`
  - `UNSUCCESSFUL_GEOCODING_FOR_AREA_SEARCH`

- VEHICLE_PROPERTY_VALUE_BACKWARDS_MAPPED

Example 49. FindTruckOffersRequest Outside Search

```
<con:FindTruckOffersRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:TruckOfferFilterType"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:updatedAfterDateTime>2020-05-
01T01:00:00.123Z</con:updatedAfterDateTime>
      <con:queryDateTime>2020-05-01T12:00:00.123Z</con:queryDateTime>
      <con:firstResult>0</con:firstResult>
      <con:maxResults>10</con:maxResults>
      <con:sortings>
         <con:sorting>
            <con:field>creationDateTime</con:field>
            <con:ascending>false</con:ascending>
         </con:sorting>
         <con:sorting>
            <con:field>startDate</con:field>
            <con:ascending>false</con:ascending>
         </con:sorting>
      </con:sortings>
      <con:date>
         <con:individualDates>
           <con:date>2020-05-01</con:date>  ①
           <con:date>2020-05-07</con:date>
           <con:date>2020-05-09</con:date>
         </con:individualDates>
      </con:date>
      <con:vehicleProperties>
         <con:property>
            <con:category>VEHICLE_EQUIPMENT</con:category>
            <con:values>
               <con:value>ADR_EQUIPMENT_SET</con:value>
               <con:value>ESCORT_VEHICLE_TYPE_3</con:value>
            </con:values>
         </con:property>
         <con:property>
            <con:category>VEHICLE_SWAP_BODY</con:category>
            <con:values>
               <con:value>FORTY_FEET_CONTAINER</con:value>
            </con:values>
         </con:property>
         <con:property>
            <con:category>VEHICLE_BODY</con:category>
            <con:values>
               <con:value>CURTAIN_SIDER</con:value>
               <con:value>VAN_CAR</con:value>
            </con:values>
         </con:property>
         <con:property>
```

```
            <con:category>VEHICLE_TYPE</con:category>
            <con:values>
               <con:value>TRAILER</con:value>
            </con:values>
         </con:property>
      </con:vehicleProperties>
      <con:startLocation>
         <con:countrySearch>  ②
            <con:searchLine>
               <con:country>FR</con:country>
               <con:postalCodes>
                 <con:postalCode>68</con:postalCode>
               </con:postalCodes>
            </con:searchLine>
         </con:countrySearch>
      </con:startLocation>
      <con:destinationLocation>
         <con:areaSearch>  ③
            <con:country>ES</con:country>
            <con:postalCode>11033</con:postalCode>
            <con:city>San Roque</con:city>
         </con:areaSearch>
      </con:destinationLocation>
   </con:payload>
</con:FindTruckOffersRequest>
```

① Instead of individual dates it is possible to provide a date interval, see FindCargoOffersRequest example.

② The `countrySearch` in the `startLocation` means that we are looking for loading space offers from France with postal codes starting with '68'.

③ The `areaSearch` in the `destinationLocation` means that we are looking for loading space offers with destination 11933 San Roque, Spain. Note that the `areaInKilometres` field is missing as loading space offers' destinations are submitted with an area size.

Example 50. FindTruckOffersResponse.payload Outside Search

```
<con:payload xsi:type="con:TruckOfferListType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
     <con:publicId>MASpTY6_RI2zqJ5Ox_7f4w</con:publicId>
     <con:creationDateTime>2020-03-15T13:42:32Z</con:creationDateTime>
     <con:customer xsi:type="con:CustomerDetailWrapperType">   ①
        <con:detail>
           <con:publicId>104</con:publicId>
           <con:name>Timotrans International GmbH &amp;amp; Co. KG</con:name>
           <con:phone>+49 211 740 00 -0</con:phone>   ③
           <con:fax>+49 211 740 00 -47</con:fax>    ③
           <con:companyAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1</con:streetOrPostbox>
           </con:companyAddress>
           <con:postalAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1</con:streetOrPostbox>
           </con:postalAddress>
        <con:taxId>DE272039727</con:taxId>
        </con:detail>
      </con:customer>
      <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">  ②
        <con:detail>
           <con:lastName>Doe</con:lastName>
           <con:firstName>John</con:firstName>
           <con:title>MR</con:title>
           <con:email></con:email>   ③
           <con:languages>
              <con:language>de</con:language>
              <con:language>es</con:language>
           </con:languages>
           <con:position></con:position>
           <con:phone>111222333</con:phone>   ③
           <con:phonePrivate></con:phonePrivate>  ③
           <con:phoneMobile></con:phoneMobile>  ③
           <con:fax></con:fax>   ③
           <con:faxPrivate></con:faxPrivate>  ③
        </con:detail>
      </con:contactPerson>
      <!-- further fields omitted here, see TruckOfferType above -->
      <con:deepLink>
https://my.timocom.com/fakedeeplink/PBrO34gdScmJ2kVfNNTnKA</con:deepLink>
      <!-- further fields omitted here, see TruckOfferType above -->
```

```
      <con:publicRemark>Public remark</con:publicRemark>
      <!-- further fields omitted here, see TruckOfferType above -->
   </con:entity>
   <con:entity>
      <!-- fields omitted here-->
   </con:entity>
</con:payload>
```

① You need a dedicated authorization if you want to see the customer details of each offer.

② You need a dedicated authorization if you want to see the contact person details of each offer.

③ Field is filled according to the contact channel setting for this offer.

7.5.5. Looking Up Loading Space Offers Outside of Your Customer Group

This search is restricted to authorized customer groups, only.

- WSDL operation: `LookupTruckOffers`
- Request type: `LookupTruckOffersRequest`
- Request payload (filter) type: `PublicIdsFilterType`
- Response type: `LookupTruckOffersResponse`
- Response payload type: `TruckOfferListType`
- Successful response status: `OK`
- Possible Message keys in response messages list:
  - `INVALID_ENDPOINT`
  - `INVALID_ID_VALUE`
  - `MAX_NUMBER_IDS_EXCEEDED`
  - `MISSING_ID_VALUE`

Example 51. LookupTruckOffersRequest Outside Search

```
<con:LookupTruckOffersRequest version="2.6.0">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:PublicIdsFilterType"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
       <con:publicIds>
          <con:publicId>timocom_public_offer_id_1</con:publicId>
          <con:publicId>timocom_public_offer_id_2</con:publicId>
       </con:publicIds>
   </con:payload>
</con:LookupTruckOffersRequest>
```

Example 52. LookupTruckOffersResponse.payload Outside Search

```
<con:payload xsi:type="con:TruckOfferListType"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <con:entity>
     <con:publicId>timocom_public_offer_id_1</con:publicId>
     <con:creationDateTime>2020-03-15T13:42:32Z</con:creationDateTime>
     <con:customer xsi:type="con:CustomerDetailWrapperType">   ①
        <con:detail>
           <con:publicId>104</con:publicId>
           <con:name>Timotrans International GmbH &amp;amp; Co. KG</con:name>
           <con:phone>+49 211 740 00 -0</con:phone>   ③
           <con:fax>+49 211 740 00 -47</con:fax>    ③
           <con:companyAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1</con:streetOrPostbox>
           </con:companyAddress>
           <con:postalAddress>
              <con:country>DE</con:country>
              <con:postalCode>40699 </con:postalCode>
              <con:city>Erkrath</con:city>
              <con:streetOrPostbox>Timocom-Platz 1</con:streetOrPostbox>
           </con:postalAddress>
        <con:taxId>DE272039727</con:taxId>
        </con:detail>
      </con:customer>
      <con:contactPerson xsi:type="con:ContactPersonDetailWrapperType">  ②
        <con:detail>
           <con:lastName>Doe</con:lastName>
           <con:firstName>John</con:firstName>
           <con:title>MR</con:title>
           <con:email></con:email>   ③
           <con:languages>
              <con:language>de</con:language>
              <con:language>es</con:language>
           </con:languages>
           <con:position></con:position>
           <con:phone>111222333</con:phone>   ③
           <con:phonePrivate></con:phonePrivate>  ③
           <con:phoneMobile></con:phoneMobile>  ③
           <con:fax></con:fax>   ③
           <con:faxPrivate></con:faxPrivate>  ③
        </con:detail>
      </con:contactPerson>
      <con:publicRemark>Public remark</con:publicRemark>
      <!-- further fields omitted here, see TruckOfferType above -->
   </con:entity>
   <con:entity>
```

```
      <!-- 2nd offer w/ fields omitted here -->
   </con:entity>
</con:payload>
```

① You need a dedicated authorization if you want to see the customer details of each offer.

② You need a dedicated authorization if you want to see the contact person details of each offer.

③ Field is filled according to the contact channel setting for this offer.

7.5.6. Storing a Loading Space Offer

- WSDL operation: `StoreTruckOffer`
- Request type: `StoreTruckOfferRequest`
- Request payload type: `TruckOfferType`
- Response type: `StoreTruckOfferResponse`
- Response payload type: `null`
- Successful response status: `STORED`
- Possible Message keys in response messages list:
  - `AMBIGUOUS_DESTINATION_TYPE_CHOICES`
  - `AREA_SIZE_SET_TO_DEFAULT`
  - `AREA_SIZE_SET_TO_NULL`
  - `CONTACT_NOT_REGISTERED`
  - `CUG_CHANGED_TO_MANDATORY`
  - `CUG_IGNORED_AND_NO_MANDATORY_FOUND`
  - `CUG_PUBLICATION_TIME`
  - `CUG_RETENTION_TIME_CHANGED`
  - `CUSTOMER_NOT_REGISTERED`
  - `DATE_OUT_OF_RANGE`
  - `DEPRECATED_FIELD_OR_VALUE`
  - `ILLEGAL_TRAILER_LENGTH_FOR_VEHICLE_TYPE`
  - `ILLEGAL_TRAILER_WEIGHT_FOR_VEHICLE_TYPE`
  - `ILLEGAL_TRUCK_LENGTH_FOR_VEHICLE_TYPE`
  - `ILLEGAL_TRUCK_WEIGHT_FOR_VEHICLE_TYPE`
  - `INVALID_AREA_SIZE`
  - `INVALID_CHARACTERS`
  - `INVALID_CONTACT_REF`
  - `INVALID_EMAIL`

- INVALID_FAX_NUMBER
- INVALID_ID_VALUE
- INVALID_MOBILE_NUMBER
- INVALID_NEGATIVE_DECIMAL_VALUE
- INVALID_PHONE_NUMBER
- INVALID_POSTAL_CODE
- INVALID_WEIGHT_FOR_VEHICLE_TYPE
- MAX_LENGTH_CITY_EXCEEDED
- MAX_LENGTH_EMAIL_EXCEEDED
- MAX_LENGTH_FIRST_NAME_EXCEEDED
- MAX_LENGTH_LAST_NAME_EXCEEDED
- MAX_LENGTH_POSTCODE_EXCEEDED
- MAX_LENGTH_REMARKS_EXCEEDED
- MISSING_BACKLINK
- MISSING_CITY
- MISSING_CONTACT
- MISSING_CONTACT_CHANNEL
- MISSING_COUNTRY_CODE
- MISSING_CUG_PUBLICATION_TYPE
- MISSING_CUSTOMER_REF
- MISSING_DESTINATION
- MISSING_EMAIL
- MISSING_FAX_NUMBER
- MISSING_FIRST_NAME
- MISSING_LAST_NAME
- MISSING_MOBILE_NUMBER
- MISSING_PHONE_NUMBER
- MISSING_TITLE
- MISSING_VEHICLE_BODY
- MISSING_VEHICLE_TYPE
- TOTAL_LENGTH_ABOVE_MAXIMUM
- TOTAL_LENGTH_BELOW_MINIMUM
- TOTAL_WEIGHT_ABOVE_MAXIMUM
- TOTAL_WEIGHT_BELOW_MINIMUM
- TRAILER_LENGTH_BELOW_MINIMUM

- TRAILER_WEIGHT_BELOW_MINIMUM
- TRUCK_LENGTH_BELOW_MINIMUM
- TRUCK_WEIGHT_BELOW_MINIMUM
- UNKNOWN_CONTACT_CHANNEL
- UNKNOWN_COUNTRY_CODE
- UNKNOWN_CUG_PUBLICATION_TYPE
- UNKNOWN_LANGUAGE_CODE
- UNKNOWN_TITLE
- UNKNOWN_VEHICLE_BODY
- UNKNOWN_VEHICLE_EQUIPMENT
- UNKNOWN_VEHICLE_PROPERTY_CATEGORY
- UNKNOWN_VEHICLE_TYPE
- UNSUCCESSFUL_GEOCODING
- UPDATE_IGNORED
- VEHICLE_PROPERTY_VALUE_MOVED

Example 53. StoreTruckOfferRequest

```
<con:StoreTruckOfferRequest
   version="2.6.0"
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload xsi:type="con:TruckOfferType"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:id>OFFER_REF</con:id>
      <con:customer xsi:type="con:CustomerRefWrapperType">
        <con:customerRef>CUSTOMER_REF</con:customerRef>
      </con:customer>
      <con:contactPerson xsi:type="con:ContactRefWrapperType">
        <con:contactRef>CONTACT_REF</con:contactRef>
      </con:contactPerson>
      <con:publicRemark></con:publicRemark>
      <con:contactChannels>
         <con:contactChannel>EMAIL_ADDRESS</con:contactChannel>
         <con:contactChannel>PHONE_NUMBER</con:contactChannel>
      </con:contactChannels>
      <con:internalRemark></con:internalRemark>
      <con:trackable>false</con:trackable>
      <con:closedUserGroupSetting
        xsi:type="con:RequestClosedUserGroupSettingType">
         <con:id>CUG_REF</con:id>
         <con:publicationType>EXTERNAL_LATER</con:publicationType>
         <con:remark></con:remark>
         <con:retentionTimeMinutes>22</con:retentionTimeMinutes>
         <con:resetRetentionTime>false</con:resetRetentionTime>
      </con:closedUserGroupSetting>
      <con:vehicleProperties>
         <con:property>
            <con:category>VEHICLE_BODY</con:category>
            <con:values>
               <con:value>CURTAIN_SIDER</con:value>
            </con:values>
         </con:property>
         <con:property>
            <con:category>VEHICLE_EQUIPMENT</con:category>
            <con:values>
               <con:value>ADR_EQUIPMENT_SET</con:value>
            </con:values>
         </con:property>
         <con:property>
            <con:category>VEHICLE_SWAP_BODY</con:category>
            <con:values>
               <con:value>FORTY_FIVE_FEET_CONTAINER</con:value>
            </con:values>
         </con:property>
         <con:property>
```

```
            <con:category>VEHICLE_TYPE</con:category>
            <con:values>
               <con:value>WAGGON_AND_DRAG</con:value>
            </con:values>
         </con:property>
      </con:vehicleProperties>
      <con:logisticsDocumentTypes>
          <con:documentType>GMP_CERTIFICATE</con:documentType>
      </con:logisticsDocumentTypes>
      <con:truckWeightInTons>2.3</con:truckWeightInTons>
      <con:trailerWeightInTons>1.2</con:trailerWeightInTons>
      <con:truckLengthInMetres>5.22</con:truckLengthInMetres>
      <con:trailerLengthInMetres>2.34</con:trailerLengthInMetres>
      <con:startDate>2020-08-01</con:startDate>
      <con:start>
         <con:country>FR</con:country>
         <con:postalCode>13281</con:postalCode>
         <con:city>Marseille</con:city>
      </con:start>
      <con:destination>
         <con:area>
            <con:country>IT</con:country>
            <con:postalCode>50139</con:postalCode>
            <con:city>Firenze</con:city>
            <con:areaInKilometres>50</con:areaInKilometres>
         </con:area>
      </con:destination>
   </con:payload>
</con:StoreTruckOfferRequest>
```

Example 54. StoreTruckOfferResponse

```
<con:StoreTruckOfferResponse
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hdltywlvzl</con:requestRef>
   <con:status>STORED</con:status>
   <con:messages>
      <con:message> ①
         <con:messageLevel>INFO</con:messageLevel>
         <con:messageKey>CUG_PUBLICATION_TIME</con:messageKey>
         <con:logMessage>Offer will become public at 2020-08-
01T14:24:09.113Z</con:logMessage>
      </con:message>
   </con:messages>
</con:StoreTruckOfferResponse>
```

① The publication time for all TIMOCOM customers of a Closed User Group offer will be returned as INFO level message. You can also get this publication date time by a GetTruckOfferRequest.

7.5.7. Deleting a Loading Space Offer

- WSDL operation: `DeleteTruckOffer`
- Request type: `DeleteTruckOfferRequest`
- Request payload type: `CustomerReferencingEntityKeyType`
- Response type: `DeleteTruckOfferResponse`
- Response payload type: `null`
- Successful response status: `DELETED`
- Possible Message keys in response messages list:
  - `ALREADY_DELETED`
  - `CUSTOMER_NOT_REGISTERED`

Example 55. DeleteTruckOfferRequest

```
<con:DeleteTruckOfferRequest
  version="2.6.0"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
   <con:payload
     xsi:type="con:CustomerReferencingEntityKeyType"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <con:id>OFFER_REF</con:id>
      <con:customerRef>CUSTOMER_REF</con:customerRef>
   </con:payload>
</con:DeleteTruckOfferRequest>
```

Example 56. DeleteTruckOfferResponse

```
<con:DeleteTruckOfferResponse
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <con:requestRef>8hebvbnti4l</con:requestRef>
   <con:status>DELETED</con:status>
   <con:messages/>
</con:DeleteTruckOfferResponse>
```

7.6. Reference Data Endpoint

7.6.1. Getting all Reference Data

- WSDL operation: `GetReferencedData`
- Request type: `GetReferencedDataRequest`
- Request payload (filter) type: `FilterType` or omitted
- Response type: `GetReferenceDataResponse`
- Response payload type: `ReferencedDataType`
- Successful response status: `OK`

SOAP request for getting all valid reference data

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:GetReferencedDataRequest version="2.6.0">
         <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
         <con:payload xsi:type="con:FilterType" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"/>
      </con:GetReferencedDataRequest>
   </soap:Body>
</soap:Envelope>
```

GetReferenceDataResponse.payload (extract)

```
<con:payload xsi:type="con:ReferencedDataType" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance">
   <con:releaseNotes>
      <con:release>
         <con:referencedDataVersion>2.0.0</con:referencedDataVersion>
         <con:schemaVersion>2.6.0</con:schemaVersion>
         <con:releaseDate>2020-08-01</con:releaseDate>
         <con:note>What has changed</con:note>
      </con:release>
   </con:releaseNotes>
   <con:vehicleBodies>
      <con:referencedValue>
         <con:value>CURTAIN_SIDER</con:value>
         <con:translations>
            <con:translation lang="en">Curtainsider</con:translation>
            <con:translation lang="de">Plane</con:translation>
         </con:translations>
      </con:referencedValue>
      <con:referencedValue>
         <con:value>BOX</con:value>
         <con:translations>
            <con:translation lang="en">Box</con:translation>
            <con:translation lang="de">Koffer</con:translation>
         </con:translations>
      </con:referencedValue>
      <con:referencedValue>
         <con:value>THERMO</con:value>
         <con:translations>
            <con:translation lang="en">Thermo</con:translation>
            <con:translation lang="de">Thermo</con:translation>
         </con:translations>
      </con:referencedValue>
      <!-- further values omitted here -->
   </con:vehicleBodies>
   <!-- further soft enum types omitted here -->
</con:payload>
```

Chapter 8. Types of the XML Schema Definition

8.1. Entities and Their Fields

This sections describes the fields of entities, i.e. object with a life cycle, or their base types, only.

8.1.1. EntityType

The base type of this type is PayloadType.

Table 6. EntityType Fields

Field	Type	Mandatory	Description
id	EntityIdType	false	Unique key transmitted by the client. It is mandatory for store requests.
publicId	PublicIdType	false	Unique ID on TIMOCOM side. Only set in result of a search request.
creationDateTime	xsd:dateTime	false	Filled in response objects, only. Ignored in store requests

8.1.2. CustomerType

The base type of this type is EntityType.

Table 7. CustomerType Fields

Field	Type	Mandatory	Description
name	String50Type	true	Name of the company
phone	String50Type	false	
fax	String50Type	false	
companyAddress	PostalAddressType	true	Address of the company
postalAddress	PostalAddressType	false	Postal address of the company
taxId	String50Type	false	Tax ID of the company

8.1.3. ContactPersonType

The base type of this type is AbstractCustomerReferencingType.

Table 8. ContactPersonType Fields

Field	Type	Mandatory	Description
lastName	String50Type	true	Last name of the contact person
firstName	String50Type	true	First name of the contact person
title	TitleType	true	Form of address of the contact person
email	String100Type	false	E-mail address of the contact person
languages	TCLanguageISOListType	true	List of languages of the contact person
position	String50Type	false	Position of the contact person (e.g. Head of Sales)
phone	PhoneNumberType	true	Telephone number of the contact person
phonePrivate	PhoneNumberType	false	Private telephone number of the contact person
phoneMobile	PhoneNumberType	false	
fax	PhoneNumberType	false	
faxPrivate	PhoneNumberType	false	

8.1.4. ContactPersonBaseType

Either a wrapper for ContactRef string or for a complex object of type ContactPersonType

This type does not have any fields apart from its inheriting or base types.

8.1.5. ContactRefWrapperType

The base type of this type is ContactPersonBaseType. To be used on client side if the contact person is sent per reference (contactRef) within the offer.

Table 9. ContactRefWrapperType Fields

Field	Type	Mandatory	Description
contactRef	EntityIdType	true	Referred contact person of the entity, ie id of the ContactPersonType.

8.1.6. ContactPersonDetailWrapperType

The base type of this type is ContactPersonBaseType. To be used on client side if the contact person is sent embedded within the offer.

Table 10. ContactPersonDetailWrapperType Fields

Field	Type	Mandatory	Description
detail	ContactPersonType	true	Embedded contact person object

8.1.7. AbstractOfferType

The base type of this type is [AbstractBusinessInitiationEntityType].

Table 11. AbstractOfferType Fields

Field	Type	Mandatory	Description
contactChannels	ContactChannelListType	false	Specifies which contact channels of your contact and customer will be presented in the offer detail. If ommitted, all contact channels are assumed to be selected and will be set.
contactRef	EntityIdType	false	Deprecated field, use the contactPerson field. Referred contact person of the entity, ie id of the ContactPersonType.
internalRemark	String50Type	false	Remark, internally visible for users of the publishing customer
trackable	xsd:boolean	true	Localisation with TC eMap Tracking
closedUserGroupSetting	ClosedUserGroupSettingType	false	CUG settings. Relevant for customers with CUG contract, only
vehicleEquipments	VehicleEquipmentListType	false	Deprecated field, use vehicleProperties. Present / required vehicle equipment

Field	Type	Manda-tory	Description
vehicleBody	VehicleBodyType	false	Deprecated field, use vehicleProperties. Present / required vehicle body
vehicleProperties	VehiclePropertiesType	false	Bundles all vehicle properties: vehicle body, vehicle equipment, vehicle type, swap bodies, vehicle body characteristics and load securing
deepLink	String100Type	false	URL deep link to the TIMOCOM offer details. Will be set by TIMOCOM and in search results, only.
logisticsDocumentTypes	LogisticsDocumentTypeListTyp e	false	Vehicle/freight documents

8.1.8. CargoOfferType

The base type of this type is AbstractOfferType.

Table 12. CargoOfferType Fields

Field	Type	Manda-tory	Description
otherBodyPossible	xsd:boolean	true	True, if another type of vehicle body is acceptable - ATTENTION: This attribute is not supported anymore!
additionalInformationList	AdditionalInformationListType	true	
freightDescription	String50Type	false	Description of the freight
price	PriceType	false	(Proposed) price of the freight offer
paymentDueWithinDays	PaymentDaysType	false	Payment is due within specified number of days
lengthInMetres	DimensionType	true	Length of the freight
weightInTons	DimensionType	true	Weight of the freight
vehicleTypes	VehicleTypeListType	false	Deprecated field, use vehicleProperties. Accepted vehicle types for the freight
loadingPlaces	LoadingPlaceListType	true	Loading places of the freight

Field	Type	Manda-tory	Description
distanceInKilometres	KilometreType	false	Distance of the whole tour. This field is set in already published offers only (Reponse only field).
acceptPriceProposals	xsd:boolean	true	True, if price proposals for this freight offer are accepted and will be handled

8.1.9. TruckOfferType

The base type of this type is AbstractOfferType.

Table 13. TruckOfferType Fields

Field	Type	Manda-tory	Description
vehicleType	VehicleTypeType	false	Deprecated field, use vehicleProperties. Vehicle type
truckWeightInTons	DimensionType	false	Possible truck payload weight. Summed up with trailerWeightInTons has to be greater than 0. If the vehicle type is TRAILER, the value must be 0.
trailerWeightInTons	DimensionType	false	Possible trailer payload weight. Summed up with truckWeightInTons has to be greater than 0.
truckLengthInMetres	DimensionType	false	Possible truck payload length. Summed up with trailerLengthInMetres has to be greater than 0. If the vehicle type is TRAILER, the value must be 0.
trailerLengthInMetres	DimensionType	false	Possible trailer payload length. Summed up with truckLengthInMetres has to be greater than 0.
startDate	xsd:date	true	Loading date (current date + max 31 days)

Field	Type	Mandatory	Description
start	AddressType	true	Place of departure
destination	DestinationType	true	Destination area

8.1.10. PriceProposalType

The base type of this type is [AbstractBusinessInitiationEntityType].

Table 14. PriceProposalType Fields

Field	Type	Mandatory	Description
status	PriceProposalStatusType	true	Status of the price proposal. Relevant in the response of a Get/Find Price Proposal request
offerId	PublicIdType	true	Unique ID on TIMOCOM side of the referenced freight offer
price	PriceType	true	(Proposed) price of the freight offer
expirationDateTime	xsd:dateTime	true	

8.2. Closed User Group Objects and Their Fields

8.2.1. ClosedUserGroupSettingType

Table 15. ClosedUserGroupSettingType Fields

Field	Type	Mandatory	Description
id	xsd:int	true	CUG ID, has been assigned to you by TIMOCOM
publicationType	CugPublicationTypeType	true	Publication type: Visible within, only or publish outside of the CUG after expired retention time
remark	String150Type	false	Remark visible to members of this Closed User Group only
retentionTimeMinutes	xsd:int	false	Retention time for remaining in CUG, visible for CUG members, only

8.2.2. RequestClosedUserGroupSettingType

The base type of this type is ClosedUserGroupSettingType.

Table 16. RequestClosedUserGroupSettingType Fields

Field	Type	Mandatory	Description
resetRetentionTime	xsd:boolean	true	Relevant if an offer is to be updated by a request: Controls whether the retention time of an offer within in the CUG is to be recalculated by the sent value of the field retentionTimeMinutes.true = The countdown for the period in the CUG will be re-started after an update.false = The countdown for the period in the CUG will not be affected by an update. This is the default behaviour.

8.2.3. ResponseClosedUserGroupSettingType

The base type of this type is ClosedUserGroupSettingType.

Table 17. ResponseClosedUserGroupSettingType Fields

Field	Type	Mandatory	Description
publicationDate	xsd:dateTime	false	At this calculated timestamp the offer will be published outside of the CUG

8.3. Filter Types and Their Fields

8.3.1. FilterType

The base type of this type is PayloadType. PayloadType and base class for all filters. Each ReadRequestType needs a filter which specifies the search parameters.

This type does not have any fields apart from its inheriting or base types.

Outside Lookup

“Outside Lookup” means looking up offers by a given list of public Ids. The publicId is a TIMOCOM identifier of an offer and provided in each offer of a outside search result set.

8.3.2. PublicIdsFilterType

The base type of this type is FilterType. Lookup entities by publicIds (provided by a list)

Table 18. PublicIdsFilterType Fields

Field	Type	Mandatory	Description
publicIds	PublicIdListType	true	

Outside Search

8.3.3. AbstractOfferFilterType

The base type of this type is FilterType. Base class for search for offers outside of your customer group. Usage is granted to authorized customer groups only.

Table 19. AbstractOfferFilterType Fields

Field	Type	Mandatory	Description
firstResult	xsd:unsignedShort	true	Pagination: Sets the offset position in the result set to start pagination, starting with 0. firstResult = (pageNumber-1) * maxResults
maxResults	xsd:unsignedShort	true	Pagination: Maximum number of returned element of result set. Value is additionally bounded above (currently to 30) on server side.
updatedAfterDateTime	DateTimeWithTimeZone	false	Offers must be created or updated after this timestamp. This is the (left) open lower bound of a search interval in matters of the update timestamp. See queryDateTime as right upper bound.

Field	Type	Mandatory	Description
queryDateTime	DateTimeWithTimeZone	true	Timestamp of the first request with these parameters, necessary for proper pagination. A value more than 8 hours in the past will be rejected (status UNPROCESSABLE_ENTITY) and the response will contain a message key INVALID_QUERY_DATE_TIME of message level ERROR. A value in the future is set to current timestamp on server side and the response will contain a message key RESET_QUERY_DATE_TIME of message level INFO. This value is also interpreted as upper bound for the update timestamp of offers under search. In particular if updatedAfterDateTime is set, too, this field is the right upper bound of a search interval in matters of the update timestamp.
sortings	SortingListType	false	Sorting order of the result list
startDate	xsd:date	false	DEPRECATED: use date.individualDates. Corresponds with the startDate of a TruckOfferType resp. the date of the first loading place of a CargoOfferType.
date	DateSearchType	false	Search for specific or interval of loading/start dates
vehicleTypes	VehicleTypeListType	false	DEPRECATED, use vehicleProperties. List of vehicle types, logically composed by a disjunction.

Field	Type	Manda-tory	Description
vehicleBodies	VehicleBodyListType	false	DEPRECATED, use vehicleProperties. List of vehicle bodies, logically composed by a disjunction.
weightInTons	DimensionRangeType	false	Accepted (freight) resp available (loading space) weight interval in tons
lengthInMetres	DimensionRangeType	false	Accepted (freight) resp available (loading space) loading metres interval
startLocation	LocationSearchType	true	Start location / area
destinationLocation	LocationSearchType	true	Destination location / area
vehicleProperties	VehiclePropertiesType	false	Bundles all vehicle properties: vehicle body, vehicle equipment, vehicle type, swap bodies, vehicle body characteristics and load securing

8.3.4. CargoOfferFilterType

The base type of this type is AbstractOfferFilterType. Search filter for freight offers outside of your customer group. Usage is granted to authorized customer groups only.

This type does not have any fields apart from its inheriting or base types.

8.3.5. TruckOfferFilterType

The base type of this type is AbstractOfferFilterType. Search filter for loading space offers outside of your customer group. Usage is granted to authorized customer groups only.

This type does not have any fields apart from its inheriting or base types.

8.3.6. SortingType

Sorting of the result list by a field and if it is ascending or not

Table 20. SortingType Fields

Field	Type	Mandatory	Description
field	SortingFieldType	true	Field name to be sorted by. Valid values are currently “startDate” and “creationDateTime”
ascending	xsd:boolean	true	

8.3.7. SortingListType

Sortings, first by the first element of the list and if equal, by the second and so on

Table 21. SortingListType Fields

Field	Type	Mandatory	Description
sorting	SortingType	false	

8.3.8. DateSearchType

Filter parameter for date. One and only one choice elements must be set.

Table 22. DateSearchType Fields

Field	Type	Mandatory	Description
individualDates	IndividualDatesSearchType	false	Any of individual dates
dateInterval	DateIntervalType	false	Any of the dates within this interval

8.3.9. IndividualDatesSearchType

List of dates (logically composed by a disjunction), compared with loading interval/date (cargo) resp. start date (truck)

Table 23. IndividualDatesSearchType Fields

Field	Type	Mandatory	Description
date	xsd:date	true	At least one and up to 5 date elements must be set.

8.3.10. DateIntervalType

Table 24. DateIntervalType Fields

Field	Type	Mandatory	Description
start	xsd:date	true	Left lower bound (inclusive)
end	xsd:date	true	Right upper bound (inclusive)

8.3.11. LocationSearchType

Filter parameter for areas/countries. One and only one choice elements must be set.

Table 25. LocationSearchType Fields

Field	Type	Mandatory	Description
countrySearch	CountrySearchType	false	Countries with postal codes
areaSearch	AreaType	false	Area around a point. For a loading space offer's destination the areaInKilometres must be omitted: You are looking for a truck for your freight to a certain location.

8.3.12. CountrySearchType

Filter parameter for countries with optionally up to 5 postal code areas

Table 26. CountrySearchType Fields

Field	Type	Mandatory	Description
searchLine	CountrySearchLineType	true	At least one and up to 5 searchLine elements must be set.

8.3.13. CountrySearchLineType

A Country with optionally a list of postal code fragments

Table 27. CountrySearchLineType Fields

Field	Type	Mandatory	Description
country	TCCountryISO	true	
postalCodes	PostalCodeList	false	

8.3.14. PostalCodeList

Table 28. PostalCodeList Fields

Field	Type	Mandatory	Description
postalCode	PostalCodeType	false	Postal code (minimum is the first character). Up to 3 such fragments may be set.

8.3.15. DimensionRangeType

Range (lower and upper bound inclusive) for length and weight dimensions

Table 29. DimensionRangeType Fields

Field	Type	Mandatory	Description
minimum	DimensionType	true	
maximum	DimensionType	true	

8.4. Response Handling Types and Their Fields

8.4.1. MessageType

Server side message regarding processing. If its messageLevel is `ERROR` you should react and raise an exception on client side.

Table 30. MessageType Fields

Field	Type	Mandatory	Description
messageLevel	MessageLevelType	true	Escalation level. See Message Level
messageKey	MessageKeyType	true	Identifier for detailed message description. See Message Key
propertyPath	String100Type	false	Optional object tree property path pointing to the field the message refers to. Example `start.postalCode` for an invalid postal code at the start address of a truck offer.
logMessage	String150Type	false	Optional loggable message.

8.5. Further Value Types or Abstract Types and Their Fields

This sections describes the fields of further value objects, i.e. objects without a life cycle.

8.5.1. AbstractCustomerReferencingType

The base type of this type is EntityType. Base class for entities referencing a customer.

Table 31. AbstractCustomerReferencingType Fields

Field	Type	Mandatory	Description
customer	CustomerBaseType	false	In case of a StoreRequest or as part of a result of a search inside of the same customer group: Referred customer of the entity, ie the id of the CustomerType. As part of a result of a search outside of your customer group: Filled complex Customer object.
customerRef	EntityIdType	false	Deprecated field, use customer field. Referred customer of the entity, ie id of the CustomerType.

8.5.2. PayloadType

The payload is the base type of the business content of a request or response.

Table 32. PayloadType Fields

Field	Type	Mandatory	Description
extensionPoint	xsd:anyType	false	Generic extension point

8.5.3. VehiclePropertiesType

Generic type for all Vehicle properties

Table 33. VehiclePropertiesType Fields

Field	Type	Mandatory	Description
property	VehiclePropertyType	false	

8.5.4. VehiclePropertyType

Generic type for a certain Vehicle property category with its values. Category VEHICLE_TYPE: Mandatory category: Exactly one value is required for Truck and at least one value is required for Cargo. Category VEHICLE_BODY: Mandatory category: Exactly one value is required for Truck and at least one value is required for Cargo. Category VEHICLE_EQUIPMENT: Optional category Category VEHICLE_SWAP_BODY: Optional category Category VEHICLE_BODY_PROPERTY: Optional category Category VEHICLE_LOAD_SECURING: Optional category

Table 34. VehiclePropertyType Fields

Field	Type	Mandatory	Description
category	VehiclePropertyCategoryType	true	
values	VehiclePropertyValueListType	true	

8.5.5. AddressType

Table 35. AddressType Fields

Field	Type	Mandatory	Description
country	TCCountryISO	true	
postalCode	PostalCodeType	false	
city	String100Type	false	
geocoded	xsd:boolean	false	Relevant for reponses, only. True if address could be geocoded by TIMOCOM
geoCoordinate	GeoCoordinateType	false	

8.5.6. AreaType

The base type of this type is AddressType.

Table 36. AreaType Fields

Field	Type	Mandatory	Description
areaInKilometres	AreaKilometresType	false	Defines the size of the area around the address

8.5.7. DestinationType

Either an area around an address or a list of countries

Table 37. DestinationType Fields

Field	Type	Mandatory	Description
area	AreaType	true	Destination area
countries	TCCountryISOListType	true	Destination countries

8.5.8. LoadingPlaceType

Table 38. LoadingPlaceType Fields

Field	Type	Mandatory	Description
loadingType	LoadingTypeType	true	Loading type. Must be LOADING for first loading place and UNLOADING for last loading place
address	AddressType	false	Address of the loading place
earliestLoadingDate	xsd:date	false	Earliest (un)loading date of the loading place. If no value is known, it is recommended to set it to the value of the 'latestLoadingDate'.
latestLoadingDate	xsd:date	false	Latest (un)loading date of the loading place. Mandatory fieldin first loading place (start place)
date	xsd:date	false	Deprecated, use 'latestLoadingDate'. Clients using API version 2.2.7+ are not able to use this field. Latest (un)loading date of the loading place
startTime	xsd:time	false	(Un)loading start time of the loading place
endTime	xsd:time	false	(Un)loading end time of the loading place

8.5.9. PhoneNumberType

Type for landline, mobile phone and fax numbers. The characters +-/() are ignored during length restriction validation.

This type does not have any fields apart from its inheriting or base types.

8.5.10. PublicIdType

Type for TIMOCOM internal IDs

This type does not have any fields apart from its inheriting or base types.

Chapter 9. Reference Data Index

9.1. Country Codes ISO 3166-2

Table 39. TCCountryISOEnumType values

Enum Value	Meaning
AD	Andorra
AE	United Arab Emirates
AF	Afghanistan
AL	Albania
AM	Armenia
AT	Austria
AZ	Azerbaijan
BA	Bosnia-Herzegovina
BE	Belgium
BG	Bulgaria
BH	Bahrain
BY	Belarus
CH	Switzerland
CN	China
CY	Cyprus
CZ	Czech Republic
DE	Germany
DK	Denmark
DZ	Algeria
EE	Estonia
EG	Egypt
ER	Eritrea
ES	Spain
ET	Ethiopia
FI	Finland
FO	Faroe Islands
FR	France
GB	United Kingdom
GE	Georgia

Enum Value	Meaning
GI	Gibraltar
GR	Greece
HR	Croatia
HU	Hungary
IE	Ireland
IL	Israel
IN	India
IQ	Iraq
IR	Iran
IS	Iceland
IT	Italy
JO	Jordan
KG	Kyrgyzstan
KW	Kuwait
KZ	Kazakhstan
LB	Lebanon
LI	Liechtenstein
LT	Lithuania
LU	Luxembourg
LV	Latvia
LY	Libya
MA	Morocco
MC	Monaco
MD	Moldavia
ME	Montenegro
MK	North Macedonia
MN	Mongolia
MT	Malta
NL	Netherlands
NO	Norway
NP	Nepal
OM	Oman
PK	Pakistan

Enum Value	Meaning
PL	Poland
PT	Portugal
QA	Qatar
RO	Romania
RS	Serbia
RU	Russia
SA	Saudi Arabia
SE	Sweden
SI	Slovenia
SK	Slovakia
SM	San Marino
SY	Syria
TJ	Tajikistan
TM	Turkmenistan
TN	Tunisia
TR	Turkey
UA	Ukraine
UZ	Uzbekistan
VA	Vatican City
YE	Yemen

9.2. Currency Codes ISO 4217

Table 40. TCCurrencyISOEnumType values

Enum Value	Meaning
ALL	Albanian Lek
AMD	Armenian Dram
AZN	Azerbaijani Manat
BAM	Bosnia Herzegovina Convertible Mark
BGN	Bulgarian Lew
BYN	Belarusian Ruble
CHF	Swiss Franc
CZK	Czech Crown
DKK	Danish Crown

Enum Value	Meaning
EUR	Euro
GBP	British Pound
GEL	Georgian Lari
HRK	Croatian Kuna
HUF	Hungarian Forint
ISK	Icelandic Króna
KZT	Kazakhstani Tenge
MDL	Moldovan Leu
MKD	Macedonian Dinar
NOK	Norwegian Crown
PLN	Polish Złoty
RON	Rumanian Leu
RSD	Serbian Dinar
RUB	Russian Ruble
SEK	Swedish Crown
TRY	Turkish Lira
UAH	Ukrainian Hryvnia
USD	US Dollar

9.3. Language Codes ISO 639-1

Table 41. TCLanguageISOEnumType values

Enum Value	Meaning
af	Afrikaans
ar	Arabic
az	Azerbaijani
be	Belorusian
bg	Bulgarian
bs	Bosnian
ca	Catalan
ce	Chechen
co	Corsican
cs	Czech
da	Danish

Enum Value	Meaning
de	German
el	Modern Greek
en	English
eo	Esperanto
es	Spanish
et	Estonian
eu	Basque
fa	Persian
fi	Finnish
fo	Faroese
fr	French
fy	Western Frisian
ga	Irish
gd	Scottish Gaelic
gl	Galician
he	Hebrew
hr	Croatian
hu	Hungarian
hy	Armenian
id	Indonesian
is	Icelandic
it	Italian
ja	Japanese
ka	Georgian
kk	Kazakh
ku	Kurdish
kw	Cornish
ky	Kirghiz
lb	Luxembourgish
li	Limburgan
lt	Lithuanian
lv	Latvian
mk	Macedonian

Enum Value	Meaning
mn	Mongolian
mt	Maltese
nl	Dutch
no	Norwegian
oc	Occitan
pl	Polish
ps	Pashto
pt	Portuguese
rm	Romansh
ro	Romanian
ru	Russian
sc	Sardinian
sk	Slovak
sl	Slovenian
sq	Albanian
sr	Serbian
sv	Swedish
sw	Swahili
th	Thai
tr	Turkish
uk	Ukrainian
uz	Uzbek
vi	Vietnamese
wa	Walloon
zh	Chinese

9.4. Vehicle Property Category

Table 42. VehiclePropertyCategoryEnumType values

Enum Value	Meaning
VEHICLE_BODY	Vehicle body
VEHICLE_BODY_PROPERTY	Body characteristics
VEHICLE_EQUIPMENT	Equipment
VEHICLE_LOAD_SECURING	Load securing

Enum Value	Meaning
VEHICLE_SWAP_BODY	Swap bodies
VEHICLE_TYPE	Vehicle type

9.5. Vehicle Body (Vehicle Property Category)

Table 43. VehicleBodyEnumType values

Enum Value	Meaning
BOX	Box
CAR_TRANSPORTER	Car transporter
CHASSIS	Container chassis
COIL_WELL	Coil trough
CURTAIN_SIDER	Curtain
DRAWER	Skip loader
DUMP_TRAILER	Tipper
EXTENDABLE_TRAILER	Extendable trailer
FLATBED	Flatbed truck
INLOADER	Inloader
JUMBO	Jumbo
LOW_LOADER	Low loader
MEGA	Mega
MOVING_FLOOR	Walking floor
MOVING_FLOOR_FILL_MATERIAL	Walking floor (Bulk material)
PLATFORM	Drop side
REFRIGERATOR	Refrigerator
SEMI_TRAILER_WITH_INCLINED_TABLE	Semi-trailer with inclined table
SILO	Silo trailer
SPECIAL_TRUCK	Special truck
SWAP_BODY_TRUCK	Swap body truck
TANK	Tank trailer
TAUTLINER	Tautliner
THERMO	Thermo
TIPPER_ROLL_OFF	Roll-on roll-off tipper
TRACTOR_UNIT	Tractor
VAN_CAR	Panel van

9.6. Vehicle Body Property (Vehicle Property Category)

Table 44. VehicleBodyPropertyTypeEnumType values

Enum Value	Meaning
AIR_SUSPENDED	Air suspension
BACK_TIPPER	Back tipper
CODE_XL	Code XL
DOUBLE_EVAPORATOR	Dual evaporator
DOUBLE_FLOOR	Double floor
ELEVATING_ROOF	Lifting roof
FOLDING_SIDE_BOX	Folding side box
HANGING_GARMENT_CONTAINER	Hanging garment container
LONG_TRUCK	Euro combi
LOW_FLOOR	Low floor
LOW_LOADING_RAILS	Steel rails for Joloda rollers
REMOVAL_VAN	Removal truck
SIDE_TIPPER	Side tipper
SLIDING_CURTAIN	Curtainsider
SLIDING_ROOF	Sliding roof
WIDENABLE	Widenable

9.7. Vehicle Swap Body (Vehicle Property Category)

Table 45. VehicleSwapBodyTypeEnumType values

Enum Value	Meaning
FORTY_FEET_CONTAINER	40ft Container
FORTY_FIVE_FEET_CONTAINER	45ft Container
HALFPIPE_DUMPER	Skip
ROLL_ON_ROLL_OFF_CONTAINER	Roll-on roll-off container
SWAP_BODY	Demountable body
TWENTY_FEET_CONTAINER	20ft Container

9.8. Vehicle Type (Vehicle Property Category)

Table 46. VehicleTypeEnumType values

Enum Value	Meaning
TRAILER	Articulated truck
VEHICLE_UP_TO_12_T	Vehicle up to 12 t
VEHICLE_UP_TO_3_5_T	Vehicle up to 3.5 t
VEHICLE_UP_TO_7_5_T	Vehicle up to 7.5 t
WAGGON_AND_DRAG	Rigid truck

9.9. Vehicle Equipment (Vehicle Property Category)

Table 47. VehicleEquipmentEnumType values

Enum Value	Meaning
ACCESS_RAMP	Access ramp
ADR_EQUIPMENT_SET	ADR
A_PLATE	Waste carrier licence
CUSTOMS_SEAL_STRING	Customs seal string
ESCORT_VEHICLE_TYPE_3	Escort vehicle type 3
ESCORT_VEHICLE_TYPE_4	Escort vehicle type 4
FIXED_LOADING_CRANE	Fixed loading crane
MEAT_HOOK	Meat hooks
PARTITION_WALL	Partition wall
PORTABLE_FORKLIFT	Portable forklift
PORTABLE_PUMP_TRUCK	Pallet lifter
SATELLITE_TRACKING	Satellite tracking
SECOND_DRIVER	2nd driver
TAIL_LIFT	Tail lift
TARPAULIN_COVER	Tarpaulin cover
WOOD_STANCHIONS	Log stanchions

9.10. Vehicle Load Securing (Vehicle Property Category)

Table 48. VehicleLoadSecuringTypeEnumType values

Enum Value	Meaning
ANTI_SLIP_MATS	Anti slip mats
BOARD_WALL	Side panels
EDGE_PROTECTION	Edge protection

Enum Value	Meaning
LASHING_STRAPS	Lashing straps
LOCKING_BAR	Locking bar
PALLET_STOP_BAR	Pallet retaining bar
PERFORATED_BATTEN	Perforated batten
STANCHIONS	Stanchions
TENSION_CHAIN	Lashing chains

9.11. Logistics Document Types

Table 49. LogisticsDocumentTypeEnumType values

Enum Value	Meaning
GMP_CERTIFICATE	GMP Certificate

9.12. Additional Cargo Information

Table 50. AdditionalInformationEnumType values

Enum Value	Meaning
CRANE_LOADING_UNLOADING	Crane loading / unloading
FULL_LOAD	Full load
HEAVY_LOAD	Heavy load
LOADING_EQUIPMENT_EXCHANGE	Loading equipment exchange
LOAD_FOR_DIRECT_DELIVERY	Load for direct delivery
NO_ADDITIONAL_LOAD	No additional load
NO_TRANSHIPMENT	No transhipment
OVERSIZE_HEIGHT	Oversize - height
OVERSIZE_LENGTH	Oversize - length
OVERSIZE_WIDTH	Oversize - width
PART_LOAD	Part load
ROUNDTRIP	Round trip
SIDE_LOADING	Side loading
SPECIAL_TRANSPORT_TIME_SENSITIVE	Special transport time-sensitive

9.13. Contact channel to be shown in offer detail

Table 51. ContactChannelEnumType values

Enum Value	Meaning
BACKLINK	Backlink
EMAIL_ADDRESS	E-mail address
FAX_NUMBER	Fax number
MOBILE_NUMBER	Mobile number
PHONE_NUMBER	Phone number

9.14. Closed User Group Publication Type

Table 52. CugPublicationTypeEnumType values

Enum Value	Meaning
EXTERNAL_LATER	Publish later
INTERNAL_ONLY	Closed user group (CUG) only

9.15. Loading Type

Table 53. LoadingTypeEnumType values

Enum Value	Meaning
LOADING	Loading place
UNLOADING	Unloading place

9.16. Form of Address (Person)

Table 54. TitleEnumType values

Enum Value	Meaning
MR	Mr
MRS	Ms

9.17. Status of a price proposal (read only)

Table 55. PriceProposalStatusEnumType values

Enum Value	Meaning
ACCEPTED	Accepted
ACTIVE	Active
EXPIRED	Expired
REJECTED	Rejected
WITHDRAWN	Withdrawn

9.18. Message Level

Table 56. MessageLevelEnumType values

Enum Value	Meaning
ERROR	ERROR
INFO	INFO
WARN	WARN

9.19. Response Status

Table 57. StatusEnumType values

Enum Value	Meaning
BAD_REQUEST	Bad request
CONFLICT	Conflict during processing
DELETED	Deleted
INTERNAL_SERVER_ERROR	Internal server side error
NOT_FOUND	Requested resource does not exist
OK	OK
STORED	Imported
UNAUTHORIZED	Unauthorized request
UNPROCESSABLE_ENTITY	Validation failure

9.20. Message Key

Table 58. MessageKeyEnumType values

Enum Value	Meaning	Message Level
ALREADY_DELETED	Entity was already deleted	WARN
AMBIGUOUS_DESTINATION_TYPE_CHOICES	Ambiguous destination: Choose either countries or area.	ERROR
AREA_SIZE_SET_TO_DEFAULT	The destination is geocodable, thus we set the missing area size (areaInKilometres) to the default value 50 km.	INFO
AREA_SIZE_SET_TO_NULL	Unable to geocode the destination, thus the area size (areaInKilometres) was set to null.	INFO
CONFLICT	Conflict during storage of an entity	ERROR

CONTACT_NOT_DELETABLE	Contact person is system user and thus not deletable	ERROR
CONTACT_NOT_MODIFIABLE	Contact person is system user and thus not modifiable	ERROR
CONTACT_NOT_REGISTERED	Contact person unknown	ERROR
CONTACT_USER_LOCKED	Contact person’s system user locked	ERROR
CUG_CHANGED_TO_MANDATORY	The offer was published with a mandatory Closed User Group, not with the submitted one.	WARN
CUG_IGNORED_AND_NO_MANDATORY_FOUND	The customer is not member of the submitted CUG and there is no mandatory CUG as fallback.	WARN
CUG_PUBLICATION_TIME	The offer will be published outside of the Closed User Group at the timestamp given in the log message.	INFO
CUG_RETENTION_TIME_CHANGED	The retention time in the Closed User Group was changed.	WARN
CUSTOMER_NOT_REGISTERED	Customer unknown or locked	ERROR
DATE_OUT_OF_RANGE	Loading or start date must not be in the past nor more than 31 days in the future	ERROR
DEPRECATED_FIELD_OR_VALUE	A field or value is deprecated and might not be supported any longer in the future	WARN
DETACHED_DELETED_ENTITIES	In the meantime deleted entities have been detached from the result set at the requested time	INFO
ILLEGAL_TRAILER_LENGTH_FOR_VEHICLE_TYPE	The trailer length must not be set for the given vehicle type	ERROR
ILLEGAL_TRAILER_WEIGHT_FOR_VEHICLE_TYPE	The trailer weight must not be set for the given vehicle type	ERROR
ILLEGAL_TRUCK_LENGTH_FOR_VEHICLE_TYPE	The truck length must not be set for the given vehicle type	ERROR
ILLEGAL_TRUCK_WEIGHT_FOR_VEHICLE_TYPE	The truck weight must not be set for the given vehicle type	ERROR
INSUFFICIENT_VERSION	A feature required a higher submitted version	ERROR
INVALID_AREA_SIZE	Invalid area size for geocodable destination location	ERROR
INVALID_BACKLINK_URL	Invalid backlink URL	ERROR
INVALID_CHARACTERS	Removed invalid characters from city	WARN

INVALID_CONTACT_REF	Invalid ContactRef (Contact person reference id)	ERROR
INVALID_CUSTOMER_REF	Invalid customer reference	ERROR
INVALID_DATE_FORMAT	Invalid date format	ERROR
INVALID_DATE_INTERVAL	The dateInterval is invalid	ERROR
INVALID_DESTINATION_LOADING_TYPE	Invalid loading type of last loading place	ERROR
INVALID_EMAIL	Invalid e-mail address	ERROR
INVALID_ENDPOINT	Invalid endpoint	ERROR
INVALID_FAX_NUMBER	Invalid fax number	ERROR
INVALID_FIELD_VALUE	Invalid field value	ERROR
INVALID_ID_VALUE	Invalid ID	ERROR
INVALID_LENGTH	The minimal or maximal length is invalid	ERROR
INVALID_LOADING_DATE_INTERVAL	First loading date must not be before latest loading date	ERROR
INVALID_LOWER_BOUND_DATE_TIME	The lower search bound “updatedAfterDateTime” must not be older than 31 days nor in the future	ERROR
INVALID_MOBILE_NUMBER	Invalid mobile phone number	ERROR
INVALID_NEGATIVE_DECIMAL_VALUE	Invalid negative decimal value	ERROR
INVALID_OFFER_REF	Invalid offer reference	ERROR
INVALID_PAYMENT_DUE_DAYS	Payment due within days: Invalid value	ERROR
INVALID_PHONE_NUMBER	Invalid telephone number	ERROR
INVALID_POSTAL_CODE	Invalid postal code for the specified country	ERROR
INVALID_PRICE_AMOUNT	Invalid price amount	ERROR
INVALID_PRICE_PROPOSAL_HANDLING	Invalid combination price proposal handling type/PublicPriceProposalId	ERROR
INVALID_PUBLIC_OFFER_ID	Invalid PublicOfferId of the referenced freight offer	ERROR
INVALID_PUBLIC_PRICE_PROPOSAL_ID	Invalid PublicPriceProposalId	ERROR
INVALID_QUERY_DATE_TIME	Invalid query date time	ERROR
INVALID_QUERY_INTERVAL	The submitted search interval (updatedAfterDateTime,queryDateTime] is invalid	ERROR
INVALID_REQUEST_PAYLOAD	Invalid or missing request payload	ERROR
INVALID_RESULT_SIZE	The value of the pagination parameter “maxResults” is invalid	ERROR

INVALID_SORTING	Invalid or incomplete sorting parameters	ERROR
INVALID_START_LOADING_TYPE	Invalid loading type of first loading place	ERROR
INVALID_TIMOCOM_ID	Invalid TimoComID	ERROR
INVALID_VERSION	Deprecated or unknown API version	ERROR
INVALID_WEIGHT	The minimal or maximal weight is invalid	ERROR
INVALID_WEIGHT_FOR_VEHICLE_TYPE	The truck weight is above the maximum for the given vehicle type	ERROR
LENGTH_REQUIREMENT_FREIGHT_DESCRIPTION_NOT_MET	The length of the freightDescription must be between 3 and 50 characters	ERROR
LOADING_DATE_AFTER_UNLOADING_DATE	Loading date must not be before unloading date	ERROR
MAX_AMOUNT_PRICE_EXCEEDED	Amount exceeded for price	ERROR
MAX_LENGTH_CITY_EXCEEDED	Field length exceeded for city	ERROR
MAX_LENGTH_EMAIL_EXCEEDED	Field length exceeded for e-mail	ERROR
MAX_LENGTH_FIRST_NAME_EXCEEDED	Field length exceeded for first name	ERROR
MAX_LENGTH_FREIGHT_DESCRIPTION_EXCEEDED	Field length exceeded for freight description	ERROR
MAX_LENGTH_LAST_NAME_EXCEEDED	Field length exceeded for surname	ERROR
MAX_LENGTH_POSTCODE_EXCEEDED	Field length exceeded for postal code	ERROR
MAX_LENGTH_REMARKS_EXCEEDED	Field length exceeded for remarks	ERROR
MAX_NUMBER_COUNTRY_SEARCH_LINES_EXCEEDED	Maximum number of county search lines exceeded	ERROR
MAX_NUMBER_DATES_EXCEEDED	Search by maximum 5 individualDates	ERROR
MAX_NUMBER_IDS_EXCEEDED	Maximum number of submitted IDs exceeded	ERROR
MAX_NUMBER_LOADING_PLACES_EXCEEDED	Maximum number of loading places exceeded	ERROR
MAX_NUMBER_POSTAL_CODES_EXCEEDED	Maximum number of postal code fragments exceeded	ERROR
MAX_NUMBER_VEHICLE_BODIES_EXCEEDED	Maximum number of vehicle bodies exceeded	ERROR
MAX_NUMBER_VEHICLE_BODY_PROPERTIES_EXCEEDED	Maximum number of vehicle body properties exceeded	ERROR
MAX_NUMBER_VEHICLE_EQUIPMENT_EXCEEDED	Maximum number of vehicle equipment values exceeded	ERROR
MAX_NUMBER_VEHICLE_LOAD_SECURING_EXCEEDED	Maximum number of vehicle load securing values exceeded	ERROR

MAX_NUMBER_VEHICLE_SWAP_BODIES_EXCEEDED	Maximum number of vehicle swap bodies exceeded	ERROR
MAX_NUMBER_VEHICLE_TYPES_EXCEEDED	Maximum number of vehicle types exceeded	ERROR
MISSING_AREA	Missing area	ERROR
MISSING_AREA_SIZE	Missing area size for area search	ERROR
MISSING_BACKLINK	Missing backlink URL	ERROR
MISSING_CITY	Missing city	ERROR
MISSING_CONTACT	Missing contact person (reference or detail)	ERROR
MISSING_CONTACT_CHANNEL	An embedded contact person must have at least one contact channel field set	ERROR
MISSING_COUNTRY_CODE	Missing country code	ERROR
MISSING_CUG_PUBLICATION_TYPE	Missing Closed User Group publication type	ERROR
MISSING_CURRENCY_CODE	Missing currency	ERROR
MISSING_CUSTOMER_REF	Missing customer reference	ERROR
MISSING_DATE	Missing date in dateInterval	ERROR
MISSING_DATE_TIME	Missing date time value	ERROR
MISSING_DESTINATION	Missing destination (either countries or area)	ERROR
MISSING_DESTINATION_LOCATION	Missing destination location	ERROR
MISSING_EMAIL	Missing e-mail address	ERROR
MISSING_FAX_NUMBER	Missing fax number	ERROR
MISSING_FIELD_VALUE	Missing field value	ERROR
MISSING_FIRST_NAME	Missing first name	ERROR
MISSING_FIRST_OR_LAST_COUNTRY_CODE	Missing country code at first or last loading place	ERROR
MISSING_ID_VALUE	Missing ID	ERROR
MISSING_LAST_NAME	Missing last name	ERROR
MISSING_LOADING_PLACE	Missing loading place or unloading place	ERROR
MISSING_LOADING_TYPE	Missing loading place type	ERROR
MISSING_LOCATION_SEARCH_CHOICE	Either area search or country search must be set	ERROR
MISSING_MOBILE_NUMBER	Missing mobile phone number	ERROR
MISSING_OFFER_REF	Missing offer reference	ERROR

MISSING_PHONE_NUMBER	Missing business phone number	ERROR
MISSING_POSTAL_CODE_AREAS	Missing postal code areas	ERROR
MISSING_PRICE	Missing price	ERROR
MISSING_PRICE_AMOUNT	Missing price amount	ERROR
MISSING_PRICE_PROPOSAL_HANDLING_TYPE	Missing price proposal handling type	ERROR
MISSING_PUBLIC_OFFER_ID	Missing offerId referencing the freight offer	ERROR
MISSING_PUBLIC_PRICE_PROPOSAL_ID	Missing PublicPriceProposalId	ERROR
MISSING_QUERY_DATE_TIME	Missing query date time	ERROR
MISSING_START_DATE	Missing start date	ERROR
MISSING_START_LOCATION	Missing start location	ERROR
MISSING_TITLE	Missing title	ERROR
MISSING_VEHICLE_BODY	Missing vehicle body	ERROR
MISSING_VEHICLE_TYPE	Missing vehicle type	ERROR
MISSING_VERSION	Missing API version	WARN
MUTUALLY_EXCLUSIVE_DATE_CHOICES	It is not possible to search by individualDates and at the same time by dateInterval	ERROR
NEW_FEATURE	We rolled out new features, so we recommend upgrading to our new extended API	INFO
OBSOLETE_FIELD_OR_VALUE	A field or value is not supported any longer	ERROR
OFFER_NOT_FOUND	The referenced offer could not be found	ERROR
OFFER_NO_LONGER_PUBLISHED	The referenced offer is no longer published	ERROR
PRICE_PROPOSAL_NOT_ACCEPTED	The referenced freight offer does not accept price proposals	ERROR
PRICE_PROPOSAL_NOT_FOUND	The price proposal could not be found	ERROR
PRICE_PROPOSAL_NO_LONGER_ACTIVE	Price proposal is not active anymore	ERROR
PROTOCOL_VIOLATION	HTTPS is mandatory	ERROR
READ_ONLY_FIELD	Value of this field may be set only by TIMOCOM.	ERROR
RESET_QUERY_DATE_TIME	The submitted query date time has been reset to the current time	INFO
TOTAL_LENGTH_ABOVE_MAXIMUM	The total length is above the maximum	ERROR

TOTAL_LENGTH_BELOW_MINIMUM	The total length is below the minimum	ERROR
TOTAL_WEIGHT_ABOVE_MAXIMUM	The total weight is above the maximum	ERROR
TOTAL_WEIGHT_BELOW_MINIMUM	The total weight is below the minimum	ERROR
TRAILER_LENGTH_BELOW_MINIMUM	The trailer length is below minimum	ERROR
TRAILER_WEIGHT_BELOW_MINIMUM	The trailer weight is below minimum	ERROR
TRUCK_LENGTH_BELOW_MINIMUM	The truck length is below minimum	ERROR
TRUCK_WEIGHT_BELOW_MINIMUM	The truck weight is below minimum	ERROR
UNAUTHORIZED_ACCESS	Unauthorized access to a resource	ERROR
UNAUTHORIZED_CONTACT_CHANNEL	Authorization missing for a contact channel	ERROR
UNAUTHORIZED_SEARCH_FILTER	Authorization missing for this search filter	ERROR
UNKNOWN_ADDITIONAL_INFORMATIO N	Additional information unknown	ERROR
UNKNOWN_CONTACT_CHANNEL	Unknown contact option	ERROR
UNKNOWN_COUNTRY_CODE	Country unknown	ERROR
UNKNOWN_CUG_PUBLICATION_TYPE	Closed User Group publication type unknown	ERROR
UNKNOWN_CURRENCY_CODE	Currency unknown	ERROR
UNKNOWN_LANGUAGE_CODE	Language unknown	ERROR
UNKNOWN_LOADING_TYPE	Loading place type unknown	ERROR
UNKNOWN_LOGISTICS_DOCUMENT_TYP E	Document type unknown	ERROR
UNKNOWN_PRICE_PROPOSAL_HANDLIN G_TYPE	Unknown price proposal handling type for freight offer removal	ERROR
UNKNOWN_PRICE_PROPOSAL_STATUS	Price proposal status unknown	ERROR
UNKNOWN_TITLE	Title unknown	ERROR
UNKNOWN_VEHICLE_BODY	Vehicle body unknown	ERROR
UNKNOWN_VEHICLE_BODY_PROPERTY	Body characteristics unknown	ERROR
UNKNOWN_VEHICLE_EQUIPMENT	Vehicle equipment unknown	ERROR
UNKNOWN_VEHICLE_LOAD_SECURING	Load securing unknown	ERROR
UNKNOWN_VEHICLE_PROPERTY_CATEG ORY	Unknown vehicle property category	ERROR
UNKNOWN_VEHICLE_SWAP_BODY	Swap body unknown	ERROR
UNKNOWN_VEHICLE_TYPE	Vehicle type unknown	ERROR
UNSPECIFIC_CONSTRAINT_VIOLATED	Unspecific constraint violated	ERROR

UNSUCCESSFUL_GEOCODING	Unsuccessful geocoding	WARN
UNSUCCESSFUL_GEOCODING_FOR_AREA_SEARCH	Geocoding for area search with submitted data not possible	ERROR
UPDATE_IGNORED	No changes in submitted entity. Update was ignored.	WARN
VEHICLE_PROPERTY_VALUE_BACKWARDS_MAPPED	For keeping compatibility a value was backwards mapped	WARN
VEHICLE_PROPERTY_VALUE_MOVED	A vehicle property value has been moved to its new category	WARN

Chapter 10. Getting Started

This chapter will show you how to use the *API* Web Service (current XML schema version 2.0), the Web Service interface to the *Apps*. The chapter is a guideline for software developers who want to implement a web service client for the *API*. You should be familiar with XML and have a good knowledge of a web service framework of your favourite programming language. The chapter does neither prescribe any specific programming language for implementing the client, nor does it show in detail how an implementation is written e.g. in Java.

The chapter is not the definition of the web service interface. It should help you to get the web service running for the first time and verify that access is working in general.

See API Methods (Endpoints) for a more detailed definition of the web service.

10.1. Migrating From Older *API* Versions Before 2.6.0

10.1.1. Migrating From *API* Version before 2.6.0

- The field `customerRef` cannot be used anymore. Use `CustomerRefWrapperType` instead. See Migrating From *API* Version 2.0.
- The field `contactRef` cannot be used anymore. Use `ContactRefWrapperType` instead. See Migrating From *API* Version 2.0.
- The field `startDate` in `CargoOfferFilterType` and `TruckOfferFilterType` (search for offers) cannot be used anymore. See Deprecated Fields (since 2.3.0).

10.1.2. Migrating From *API* Version 2.2.7

Regrouped VehicleProperties

As already mentioned in the section Vehicle Properties, it is possible that we have to regroup ("repot") some vehicle property values. So, as we are introducing new categories *load securing*, *swap body* and *body property* (a.k.a. *body characteristics*) between versions 2.2.7 and 2.3.0 we regrouped a notable number of vehicle properties. This means that many values have been moved into the new categories.

The categories match the view in the *Apps* except *Stacking & lifting* whose values can still be found within the *vehicle equipment* category.[1]

The following table contains the columns:

- Value, i.e. old and in most cases new value of a vehicle property
- Old category valid until v2.2.7
- New category valid since v2.3.0
- New value if re-named; empty means value remains the same.

More hints:

- N/A means the value is not in use anymore.
- The former vehicle equipment `GMP` is not a vehicle property anymore and has been moved to the new enumeration field `logisticsDocumentTypes`.

Value	Category (until 2.2.7)	Category (since 2.3.0)	New Value
ANTI_SLIP_MATS	VEHICLE_EQUIPMENT	VEHICLE_LOAD_SECURING	
BACK_TIPPER	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
BOARD_WALL	VEHICLE_EQUIPMENT	VEHICLE_LOAD_SECURING	
CAR	VEHICLE_BODY	VEHICLE_BODY	VAN_CAR
CODE_XL	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
CONTAINER	VEHICLE_BODY	VEHICLE_BODY	CHASSIS
CURTAIN_SIDER	VEHICLE_BODY	VEHICLE_BODY	
CURTAIN_SIDER	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	SLIDING_CURTAIN
DOUBLE_FLOOR	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
DRAWER	VEHICLE_EQUIPMENT	VEHICLE_BODY	
EDGE_PROTECTION	VEHICLE_EQUIPMENT	VEHICLE_LOAD_SECURING	
FOLDING_SIDE_BOX	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
FORTY_FEET_CONTAINER	VEHICLE_EQUIPMENT	VEHICLE_SWAP_BODY	
FORTY_FIVE_FEET_CONTAINER	VEHICLE_EQUIPMENT	VEHICLE_SWAP_BODY	
FREEZER	VEHICLE_EQUIPMENT	N/A	N/A
FRIDGE	VEHICLE_EQUIPMENT	N/A	N/A
GMP	VEHICLE_EQUIPMENT	value in new field logisticsDocumentTypes	
HALFPIPE_DUMPER	VEHICLE_EQUIPMENT	VEHICLE_SWAP_BODY	
HANGING_GARMENT_CONTAINER	VEHICLE_BODY	VEHICLE_BODY_PROPERTY	
LASHING_STRAPS	VEHICLE_EQUIPMENT	VEHICLE_LOAD_SECURING	

Value	Category (until 2.2.7)	Category (since 2.3.0)	New Value
LOCKING_BAR	VEHICLE_EQUIPMENT	VEHICLE_LOAD_SECURING	
LOW_LOADING_RAILS	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
REMOVAL_VAN	VEHICLE_BODY	VEHICLE_BODY_PROPERTY	
ROLL_ON_ROLL_OFF_CONTAINER	VEHICLE_BODY	VEHICLE_BODY	TIPPER_ROLL_OFF
ROLL_ON_ROLL_OFF_CONTAINER	VEHICLE_EQUIPMENT	VEHICLE_SWAP_BODY	
SIDE_TIPPER	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
SLIDING_ROOF	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	
STANCHIONS	VEHICLE_EQUIPMENT	VEHICLE_LOAD_SECURING	
SWAP_BODY	VEHICLE_BODY	VEHICLE_BODY	SWAP_BODY_TRUCK
SWAP_BODY_TRUCK	VEHICLE_BODY	VEHICLE_BODY	
SWAP_BODY	VEHICLE_EQUIPMENT	VEHICLE_SWAP_BODY	
TELEPHONE	VEHICLE_EQUIPMENT	N/A	N/A
VAN	VEHICLE_BODY	VEHICLE_BODY	VAN_CAR
TWENTY_FEET_CONTAINER	VEHICLE_EQUIPMENT	VEHICLE_SWAP_BODY	
VAN_UP_TO_3_POINT_5_TONS	VEHICLE_TYPE	VEHICLE_TYPE	VEHICLE_UP_TO_3_5_T
VEHICLE_UP_TO_12_TONS	VEHICLE_TYPE	VEHICLE_TYPE	VEHICLE_UP_TO_12_T
VEHICLE_UP_TO_7_POINT_5_TONS	VEHICLE_TYPE	VEHICLE_TYPE	VEHICLE_UP_TO_7_5_T
WIDENABLE	VEHICLE_EQUIPMENT	VEHICLE_BODY_PROPERTY	

Outdated Fields

The usage of the former deprecated fields is not permitted anymore

- vehicleBody (deprecated since 2.2 and now outdated)
- vehicleEquipments (deprecated since 2.2 and now outdated)
- vehicleTypes (Cargo, deprecated since 2.2 and now outdated)

- vehicleType (Truck, deprecated since 2.2 and now outdated)
- date (Cargo loading place, since 2.2.7 and now outdated)
- vehicleTypes (CargoOfferFilterType, TruckOfferFilterType, since 2.3.0)
- vehicleBodies (CargoOfferFilterType, TruckOfferFilterType, since 2.3.0)
- customerRef (Cargo and Truck, since 2.6.0)
- contactRef (Cargo and Truck, since 2.6.0)
- startDate (CargoOfferFilterType, TruckOfferFilterType, deprecated since 2.3.0 and now since 2.6.0 outdated)

Deprecated Fields

The usage of the following field becomes now deprecated

- AbstractOfferFilterType.startDate: Use date.individualDates instead.

10.1.3. Migrating From *API* Version 2.2.6

Loading Dates

Until v2.2.6 the loading place of a cargo offer had just the field `date`. From v2.2.7 on there are two new fields `earliestLoadingDate` and `latestLoadingDate`. The field `date` is now deprecated, use the new fields `earliestLoadingDate` (i.e. the earliest possible) and `latestLoadingDate` (i.e. the latest possible loading date). If you are still using `date`, we copy its value to the new `earliestLoadingDate` and `latestLoadingDate` fields but return with a Message element with level WARN and a deprecation message.

10.1.4. Migrating From *API* Version 2.1

The fields `vehicleBody`, `vehicleEquipments`, `vehicleTypes` (Cargo) and `vehicleType` (Truck) are deprecated and should not be used anymore. Instead, fill the values appropriately into the generic `vehicleProperties`.

Example XML:

```
<con:vehicleProperties>
    <con:property>
       <con:category>VEHICLE_BODY</con:category>
       <con:values>
          <con:value>CURTAIN_SIDER</con:value>
       </con:values>
    </con:property>
    <con:property>
       <con:category>VEHICLE_EQUIPMENT</con:category>
       <con:values>
          <con:value>ADR_EQUIPMENT_SET</con:value>
          <con:value>FORTY_FIVE_FEET_CONTAINER</con:value>
       </con:values>
    </con:property>
    <con:property>
       <con:category>VEHICLE_TYPE</con:category>
       <con:values>
          <con:value>TRAILER</con:value>
       </con:values>
    </con:property>
 </con:vehicleProperties>
```

10.1.5. Migrating From *API* Version 2.0

The field `customerRef` and `contactRef` are deprecated and should not be used anymore. Instead, use the wrapper types.

Example XML:

```
<con:customer xsi:type="con:CustomerRefWrapperType">
   <con:customerRef>myCustomerRef</con:customerRef>
</con:customer>
<con:contactPerson xsi:type="con:ContactRefWrapperType">
   <con:contactRef>myContactRef</con:contactRef>
</con:contactPerson>
```

10.1.6. Migrating From *API* Version 1.x

This section contains a couple of hints if you are already successfully using the *API* (formerly called *TC Connect*) with an *API* version 1.x.

Technical Changes

- The URLs of the web service have changed. See URLs.
- SOAP version 1.1 is no longer supported. You have to migrate to version 1.2
  - The HTTP header `SOAPAction` must *not* be used

- The header `Content-Type` must be `application/soap+xml`.
- The SOAP namespace changed:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
   </soap:Body>
</soap:Envelope>
```

- Completely new WSDL and XSDs. This means that there will be a new (generated from the XSDs) data model, and you will have to adapt your client code accordingly
- *API* version 2 switches to resource based processing: A store or delete request now handles exactly one entity resource. This means if a contact person of an offer does not yet exist in TIMOCOM's system you will have to send a store request for this contact before the store request of an offer which references this contact.
- Reference data types like countries, vehicle bodies are handled as strings (soft enums). There is no need for a new *API* version release if new values are available. See chapter Reference Data.
- Inheritance in XML data types: The XSDs of the v1.x *API* versions did not use inheritance and thus no abstract types which made the code sometimes cumbersome and redundant. The *API* v2 make use of inheritance for XSD types, e.g. `AbstractOfferType` for any offer type, `TCConnectRequestType` for any request or the `FilterType` for any search type.

Functional Changes

- Contact person: Last name, first name and title are now mandatory. You should also provide an e-mail or phone number.
- CargoOffer: More than two loading places with optional address information.
- TruckOffer: The destination of a truck offer now may be either a list of countries, or an area (address with kilometres)
- Cargo and TruckOffer:
  - New field for internal remark (remark is visible within the same customer, only)
  - Feedback about unsuccessful geocoding attempts of the submitted address(es) as a WARNING-level message:

```
            <con:message>
               <con:messageLevel>WARN</con:messageLevel>
               <con:messageKey>UNSUCCESSFUL_GEOCODING</con:messageKey>
               <con:logMessage>country=FR,city=Unknown Town,
                 postCode=99999</con:logMessage>
            </con:message>
```

Note that other users may not find your offer if they search around a coordinate (or postal code) if

it is not geocoded.

- More detailed error message and error codes

```
            <con:message>
               <con:messageLevel>ERROR</con:messageLevel>
               <con:messageKey>INVALID_POSTAL_CODE</con:messageKey>
               <con:propertyPath>loadingPlaces[0].address</con:propertyPath>
            </con:message>
```

- New reference data for vehicle body, vehicle equipment, vehicle type, cargo additional information amongst others. The current valid values are available by a SOAP request, see Getting all Reference Data and chapter Reference Data.
- Closed User Groups: The date/time when the offer will become public is returned as an INFO-level message:

```
         <con:messages>
            <con:message>
               <con:messageLevel>INFO</con:messageLevel>
               <con:messageKey>CUG_PUBLICATION_TIME</con:messageKey>
               <con:logMessage>Offer will become public at
                2017-11-02T09:32:57.845Z</con:logMessage>
            </con:message>
         </con:messages>
```

10.2. Prerequisites

To access the *API* Web Service you will need some basic information at hand as well as some source documents to generate your client implementation. All these should have been provided by your contact person at TIMOCOM:

- The *address* of the web service test system (aka *Certification platform*) at TIMOCOM. Usually, the address of the Certification platform is:

  ```
  http://ws-test.timocom.com/tcconnect/ws_v2/soap1_2
  ```

Please check with your contact person at TIMOCOM that the address is still valid.

- Your *authentication information* for the *API* Web Service, a principal name and a password (basic security level) resp. a principal name and Web Service Security information (high security level) to be provided within the request (Soap body resp. Soap header and body).
- A *customerRef* for testing. It will be a string only valid on testing environment.
- The *schema documents* of the *API* Web Service Version 2.0. The complete definition of the schema consists of the following documents:

    - TCBasicTypes.xsd
    - TCClosedUserGroupTypes.xsd
    - TCConnectDocument.xsd
    - TCEntityTypes.xsd
    - TCGeoTypes.xsd
    - TCLocaleTypes.xsd
    - TCMessageTypes.xsd
    - TCPriceProposal.xsd
    - TCReferenceDataPayload.xsd
    - TCRequestsCargoOffer.xsd
    - TCRequestsContactPerson.xsd
    - TCRequestsCustomer.xsd
    - TCRequestsReferencedData.xsd
    - TCRequestsShared.xsd
    - TCRequestsTruckOffer.xsd
    - TCSearchFilter.xsd
    - TCVehicleDescriptionTypes.xsd
- The *WSDL file* of the *API* Web Service.
- If you want to use this WSDL file to generate a web service client, as it is recommended, you must probably modify it slightly. See section Hints for Creating Web Service Clients for details.

10.3. Scope of Your Client Implementation

If you want to be able to submit offers to the TIMOCOM *Smart Logistics System* your web service client must implement the entire functionality of the *API* Web Service.

The *API* Web Service offers you a complete set of functions for managing your truck and cargo offers as well as contact and customer data in the *API*.

The *API* was designed this way to avoid conflicts which could arise when the same offer is modified or deleted by the web interface and the *API* in parallel. Because there is no other way to modify or delete an offer created by the *API* Web Service, it is essential to implement these functions of the web service interface in your client as well.

You may also have to implement the functionality for querying data, because some parts of the information you need, e.g. the IDs (aka contactRefs) of your contact persons, will not be visible on the web interface.

To avoid problems, you must implement and test a complete web service client with all the functionality offered by the *API*. TIMOCOM will provide assistance during testing and implementation wherever possible. But, we cannot compensate for missing functionality of your

client implementation that results from an incomplete implementation of the interface. In particular, TIMOCOM cannot delete or modify your data in the *API* on your behalf, and we also cannot deliver or manage the IDs of your contact persons.

10.4. Authentication and Security

TIMOCOM offers multiple possible levels of security for accessing the web service:

Basic (a.k.a. “IP and Password” resp. “Static Password Token”) and High (a.k.a. “Web Service Security”).

All levels need a principal name which has to be sent in each request. The principal name will identify you as a customer group. If your customer group name is e.g. *My Logistic Group* it will be something like *My-Logistic-Group (The username should not contain spaces). Or it will be _MLG* if this is the common acronym for *My Logistic Group*.

10.4.1. HTTPS

In order to prevent a third party from eavesdropping, requests to the production system have to be sent via HTTPS.

10.4.2. Security Level Basic: Username and Password

Credentials set up:

1. Retrieve principal name and password from TIMOCOM
2. Let TIMOCOM register your IP range(s)

The password will be generated by TIMOCOM. Both, principal id and password, must be transmitted with every request that is sent to the web service as there is no authenticated session on TIMOCOM side.

Client With Static IP Ranges

As another important security mechanism the access to the web service is restricted to registered IP addresses. You have to provide us with a set of IP addresses which should be registered for you. Only these addresses may access the web service.

The IP address to be registered must be the address of the machine on which the web service client is running. Please consider that in some network configurations, network address translation (NAT) occurs e.g. in firewalls when sending network traffic to external hosts. Therefore, the external address which is visible for our web service may be different from the internal address you see in your local network. If you are in doubt which address is correct, please contact your local IT support.

Also make sure, that the external address of your system remains stable. If your system uses different addresses (dynamic IP), possibly from a huge address space, we would have to register your entire address space in our system. This is usually not feasible.

Client With Dynamic IP

If your system uses dynamic IP, and your IT support cannot provide you with a static address ask your TIMOCOM contact person for security level *Static Password Token*.

The generated password is longer (more than 20 characters), randomly generated and will work in combination with HTTPS protocol, only – even on the testing stage.

This level is quite new on TIMOCOM side and it is possible that it will be extended to a kind of *Dynamic Password Token*, i.e. the lifetime of the access token will be limited.

10.4.3. Security Level High: Web Service Security

Web Service Security (https://en.wikipedia.org/wiki/WS-Security) is an open standard for SOAP security aspects. In particular for the *API* it is used for submitting an encrypted signature for client authentication.

Credentials set up:

1. Create your security certificate
2. Submit the public certificate file to TIMOCOM for registering

How to Create a Security Certificate

The following example creates a certificate by using the JDK's command line keytool utility program (to be found under `$JAVA_HOME/bin`). Of course, you may use other tools like Portecle (http://portecle.sourceforge.net) which may have more comfortable user interfaces.

Hint: This is an example for RSA; but DSA is accepted, too.

1. Bear in mind that you have to deploy the keystore with your client application, so it may be a good idea to start with a fresh keystore.
2. Create a keypair and add it to the local keystore. If the keystore does not exist it will be created:

```
keytool -genkey -validity 9999 -alias myAlias -keystore pathto/keystore.jks
                          -keyalg RSA -keysize 1024 -sigalg SHA1WithRSA
```

3. Export a public certificate:

```
keytool -keystore pathto/keystore.jks -exportcert
                          -alias myAlias -file targetname.cer -rfc
```

Usage of the Certificate in Your Requests

Web Service Security means that the SOAP message's Header element contains all necessary information.

Please consider the following configuration needs for the Header:

- The Security element must contain a Timestamp
- The Key Identifier Type must be *Binary Security Token*
- The Signature (Method) Algorithm must be http://www.w3.org/2000/09/xmldsig#rsa-sha1 (resp. http://www.w3.org/2000/09/xmldsig#dsa-sha1 for DSA).
- The Signature Canonicalization must be http://www.w3.org/2001/10/xml-exc-c14n#
- The request must additionally contain the authentication element with the provided principal name (a.k.a. customer group name) within the SOAP body (see XSD).

Please consider that the *API*'s response is not signed.

Example of an enriched SOAP request with WS Security information:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
               xmlns:con="http://webservice.timocom.com/schema/connect/v2">
  <soap:Header>
    <wsse:Security
       xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
secext-1.0.xsd"
       xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
      <wsse:BinarySecurityToken
         EncodingType="http://docs.oasis-open.org/wss/2004/
            01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"
         ValueType="http://docs.oasis-open.org/wss/2004/
            01/oasis-200401-wss-x509-token-profile-1.0#X509v3"
         wsu:Id="X509-B77B136C55FC2767141456325095322140">
        MIIC9jCCArSgAwIBAgIETNPshortenedduetolengthk28BqU1jnGXfZA1M/axhAsA/nQ
      </wsse:BinarySecurityToken>
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
         Id="SIG-B77B136C55FC2767141456325095323144">
        <ds:SignedInfo>
          <ds:CanonicalizationMethod
              Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
            <ec:InclusiveNamespaces
                xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
                PrefixList="soap v1"/>
          </ds:CanonicalizationMethod>
          <ds:SignatureMethod
              Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
          <ds:Reference URI="#TS-B77B136C55FC2767141456325095320139">
            <ds:Transforms>
              <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                <ec:InclusiveNamespaces
                   xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
                   PrefixList="wsse soap v1"/>
              </ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <ds:DigestValue>JgOTEyRBTrw6uedvrX1U06b9/AM=</ds:DigestValue>
```

```
          </ds:Reference>
          <ds:Reference URI="#id-B77B136C55FC2767141456325095322143">
            <ds:Transforms>
              <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                <ec:InclusiveNamespaces
                   xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#"
                   PrefixList="v1"/>
              </ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <ds:DigestValue>UTwIFRH2IEbqEgUDmXKGUqmBaxW4=</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>XrwH2cJA==</ds:SignatureValue>
        <ds:KeyInfo Id="KI-B77B136C55FC2767141456325095322141">
          <wsse:SecurityTokenReference
             wsu:Id="STR-B77B136C55FC2767141456325095322142">
            <wsse:Reference URI="#X509-B77B136C55FC2767141456325095322140"
               ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
x509-token-profile-1.0#X509v3"/>
          </wsse:SecurityTokenReference>
        </ds:KeyInfo>
      </ds:Signature>
      <wsu:Timestamp wsu:Id="TS-B77B136C55FC2767141456325095320139">
        <wsu:Created>2016-02-24T14:44:55.320Z</wsu:Created>
        <wsu:Expires>2016-02-24T15:01:35.320Z</wsu:Expires>
      </wsu:Timestamp>
    </wsse:Security>
  </soap:Header>
  <soap:Body
     xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd"
     wsu:Id="id-B77B136C55FC2767141456325095322143">
    <con:GetCargoOfferRequest version="2.6.0"
       xmlns:con="http://webservice.timocom.com/schema/connect/v2">
      <con:authentication>PRINCIPAL_NAME</con:authentication>
      <con:payload
         xsi:type="con:CustomerReferencingEntityFilterType"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
         <con:customerRef>CUSTOMER_REF</con:customerRef>
         <con:id>OFFER_REF</con:id>
      </con:payload>
    </con:GetCargoOfferRequest>
  </soap:Body>
</soap:Envelope>
```

Integration of Web Service Security Into Your Application (Java)

If you are building a Java application we recommend using the Spring Webservices' `org.springframework.ws.client.core.WebServiceTemplate`. Below you find two code snippets for

setting up this WebServiceTemplate.

Example for a client security config file `client_securityPolicy.xml` which has to be in the classpath of your application:

```
<!-- For configuration see:
http://download.oracle.com/docs/cd/E17802_01/webservices/webservices/docs
/2.0/tutorial/doc/
 Schema is available as file: xwssconfig.wsd
-->
<xwss:SecurityConfiguration dumpMessages="true"
    xmlns:xwss="http://java.sun.com/xml/ns/xwss/config">
    <!-- Note that in the <Sign> operation, a Timestamp is exported
         in the security header and signed by default.
    -->
  <xwss:Sign includeTimestamp="true">
     <!-- keystore.jks:
        configure here the alias you declared when defining your keypair:
     -->
    <xwss:X509Token certificateAlias="myAlias" />

    <!-- For DSA keys use instead:
           http://www.w3.org/2000/09/xmldsig#dsa-sha1 -->
    <xwss:SignatureMethod algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
  </xwss:Sign>
</xwss:SecurityConfiguration>
```

Example for a Spring Webservice dependency injection configuration of a WebServiceTemplate. It needs the `client_securityPolicy.xml` classpath resource defined above:

```
import java.io.IOException;
import java.security.GeneralSecurityException;
import java.security.KeyStore;

import javax.security.auth.callback.CallbackHandler;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.core.io.ClassPathResource;
import org.springframework.oxm.jaxb.Jaxb2Marshaller;
import org.springframework.ws.client.core.WebServiceTemplate;
import org.springframework.ws.client.support.interceptor.ClientInterceptor;
import org.springframework.ws.soap.SoapVersion;
import org.springframework.ws.soap.saaj.SaajSoapMessageFactory;
import org.springframework.ws.soap.security.support.KeyStoreFactoryBean;
import org.springframework.ws.soap.security.xwss.XwsSecurityInterceptor;
import org.springframework.ws.soap.security.xwss.callback.KeyStoreCallbackHandler;

@Configuration
```

```
public class WebserviceSecurityClientProducer {

    @Bean
    public WebServiceTemplate secureWebserviceTemplate() throws Exception {
        Jaxb2Marshaller xmlMarshaller = new Jaxb2Marshaller();
        // configured value depends on the package name of the generated classes:
        xmlMarshaller.setContextPaths(
            "com.timocom.schema.tcconnect");
        xmlMarshaller.afterPropertiesSet();

        WebServiceTemplate bean = new WebServiceTemplate();

        bean.setMessageFactory(createSoap12MessageFactory());
        bean.setInterceptors(new ClientInterceptor[] {createClientInterceptor()});
        bean.setMarshaller(xmlMarshaller);
        bean.setUnmarshaller(xmlMarshaller);

        bean.setDefaultUri("https://ws-test.timocom.com/tcconnect/ws_v2/soap1_2");
        bean.afterPropertiesSet();
        return bean;
    }

    ClientInterceptor createClientInterceptor() throws Exception {
        XwsSecurityInterceptor xwsSecurityInterceptor = new XwsSecurityInterceptor();

        xwsSecurityInterceptor.setPolicyConfiguration(
            new ClassPathResource("/client_securityPolicy.xml"));
        xwsSecurityInterceptor.setCallbackHandler(createKeyStoreHandler());
        xwsSecurityInterceptor.afterPropertiesSet();
        return xwsSecurityInterceptor;

    }

    CallbackHandler createKeyStoreHandler() throws Exception {
        KeyStoreCallbackHandler keyStoreCallbackHandler =
            new KeyStoreCallbackHandler();
        keyStoreCallbackHandler.setKeyStore(createKeyStore());
        keyStoreCallbackHandler.setPrivateKeyPassword("your_private_key_password");
        keyStoreCallbackHandler.afterPropertiesSet();
        return keyStoreCallbackHandler;
    }

    KeyStore createKeyStore() {
        KeyStoreFactoryBean keyStoreFactoryBean = new KeyStoreFactoryBean();
        keyStoreFactoryBean.setLocation(new ClassPathResource("keystore.jks"));
        keyStoreFactoryBean.setPassword("your_keystore_password");
        keyStoreFactoryBean.afterPropertiesSet();
        return keyStoreFactoryBean.getObject();
    }

    SaajSoapMessageFactory createSoap12MessageFactory() {
```

```
        SaajSoapMessageFactory soap12MsgFactory = new SaajSoapMessageFactory();
        soap12MsgFactory.setSoapVersion(SoapVersion.SOAP_12);
        soap12MsgFactory.afterPropertiesSet();
        return soap12MsgFactory;
    }
}
```

The example above needs at least the following dependencies (for detailed information please refer to the Spring WS Security documentation):

- org.springframework.ws:spring-ws-core:3.0.7.RELEASE
- org.springframework.ws:spring-ws-security:3.0.7.RELEASE
- com.sun.xml.wss:xws-security:3.0
- javax.xml:xmldsig:1.0
- javax.xml.soap:saaj-api:1.3.5
- com.sun.xml.messaging.saaj:saaj-impl:1.5.1
- wsdl4j:wsdl4j:1.6.3

10.5. Useful Tools

The challenge of implementing a web service client involves two aspects:

- Getting a connection over the internet to the web service and sending the correct information.
- Writing an implementation in a specific language that abstracts from the details of XML etc.

It is often useful to consider these aspects one after another. There are several generic test tools which can be used to access a web service without implementing the client at first. With these tools, you can directly assemble XML messages, send these messages to the web service and view the result.

In this way, you can identify network or authentication problems before you start the client implementation. This greatly reduces debugging effort. As soon as you made sure the access to the web service works on the network level, you are ready to start implementing the client.

One of the tools that can be used to test web services is SOAP-UI. See https://www.soapui.org/ for further information.

10.6. The First Request

In the following, you will see how a first example request must be constructed to access the *API* Web Service. It is assumed that you have set up your credentials successfully as described above. If you use Web Service Security the Authentication element contains your username, only.

You should start with an ID list request which looks like this:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
   xmlns:con="http://webservice.timocom.com/schema/connect/v2">
   <soap:Header/>
   <soap:Body>
      <con:FindCargoOfferKeysRequest version="2.6.0">
         <con:authentication>PRINCIPAL_NAME:PASSWORD</con:authentication>
         <con:payload
            xsi:type="con:ExtendedCustomerReferencingEntityFilterType"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
             <con:customerRef>99</con:customerRef>
         </con:payload>
      </con:FindCargoOfferKeysRequest>
   </soap:Body>
</soap:Envelope>
```

The request asks the system to return the IDs of all data objects which are stored in the *API* for customer group *My-Logistic-Company*. If the call to the web service is successful, you should receive the following response:

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
   <env:Header/>
   <env:Body>
      <con:FindCargoOfferKeysResponse
        xmlns:con="http://webservice.timocom.com/schema/connect/v2">
         <con:requestRef>8qyattrkxdx</con:requestRef>
         <con:status>OK</con:status>
         <con:messages/>
         <con:payload xsi:type="con:CustomerReferencingEntityKeyListType"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      </con:FindCargoOfferKeysResponse>
   </env:Body>
</env:Envelope>
```

The web service response does not contain any IDs (the payload is empty) because you have not uploaded data to the *API* yet. Nevertheless, the `status` value `OK` shows that the system has processed the request successfully.

10.7. What “Successful Processing” Means

It is important to understand that there is a difference between a request that is successfully processed by the system, and a request that is free of (semantic) errors.

A request will be successfully processed by the *API* if it is a valid web service request according to the definition of the *API* Web Service and if the authentication is correct.

When the *API* processes the request, it will perform some operations based on the data element of the request. For example, it will try to create a new cargo offer in the system. If all data elements

are correct, this operation will work and there will be no errors.

However, some data elements of the request may be incorrect. For example, a value is outside the boundaries of an element's data range, or there is a reference to a non existing entity in the database, or a required element is missing. In this case, the *API* will report an error status (in most cases `UNPROCESSABLE_ENTITY`) in the response.

Nevertheless, the request was successfully processed (and as a result of the processing, the *API* told you what is wrong in the request.) It is up to you what you will do with this information. Perhaps you will give some feedback to the user and point out which part of the data he just entered caused the error. It is also a good idea to check the data already on the client side against the XML schema before sending it to the *API*.

Just keep in mind that a request which is processed successfully and returns an expected response status means that the *API* and your client implementation are working fine. If the response contains error messages, there is a problem with the data you have sent, and the *API* will tell you as detailed as possible where the problem is.

See also chapter Response Handling.

10.8. Hints for Creating Web Service Clients

To access the *API* Web Service, you have to implement a matching client which is capable of sending requests to the web service. The request must conform to the WSDL and schema definitions of the *API*.

As said before your client should implement all operations of the web service, especially if you want to store offers you must implement the operation for deleting offers as well. Offers which are created via the *API* are visible in the *Apps*, but they cannot be deleted or updated there. If you want to delete or update an offer, you must send a corresponding request through the *API*. However, each offers will be deleted on TIMOCOM side if the offer is expired at a certain date which is usually the day after its expiration date (which is the startDate if it is a loading space offer, or the first loading date of a freight offer).

Though it is possible to implement a client from scratch, it is highly recommended using a framework and to let your IDE generate the client implementation for you. Modern IDEs, e.g. Eclipse or Microsoft Visual Studio, offer wizards for generating web service clients. Another approach is, let your build tool (like Maven or Gradle) generate the transport classes out of the XSDs and use frameworks like Spring which have a similar approach. Instead of going through a set of dialogs in a wizard, you have to describe all parameters of the web service in an XML file or `@Configuration` annotated Java class. The framework will interpret this description and interact with the web service for you.

Frameworks for web service access take the WSDL description of the service as input and interpret this description. Very often, they take the URLs in this description literally and try to access these URLs, e.g. to load additional schema files directly from the web service site. Unfortunately, this does not work for the *API*, because the schema documents are not publicly accessible. XML has a mechanism called *XML catalogs* which allows you to map a remote schema location to a local one, e.g. a file in you filesystem. See Java JAX-WS Client for a Java example using *XML catalog*. If your

IDE does not implement catalogs, you have to adapt the WSDL of the *API* Web Service. Especially if you use Microsoft Visual Studio, this adaptation is necessary:

- Use your web browser to download the WSDL file from the TIMOCOM web service test server to your local machine.

The URL of the WSDL file is:

```
http://ws-test.timocom.com/tcconnect/ws_v2/soap1_2/TCConnect_v2_6_0_soap_1_2.wsdl
```

- Open the WSDL file in a text editor. At the bottom of the WSDL file, change the following line

```
<soap12:address location="/ws_v2/soap1_2/"/>
```

into:

```
<soap12:address location="http://ws-test.timocom.com/tcconnect/ws_v2/soap1_2/"/>
```

- Put the modified WSDL file and all XML Schema documents you received from TIMOCOM in a directory on your local machine. When your development environment asks you to specify the location of the WSDL file, do not enter the original URL, but enter the location of the modified WSDL file in the directory. Due to the modifications you made above, the wizard will load all required schema documents locally from the same directory, and the generated client implementation will correctly access the test system.

Depending on your IDE, you may have to repeat the second step when you want to access the production system later. The URL of the production system is:

```
https://webservice.timocom.com/tcconnect/ws_v2/soap1_2/
```

Normally this is not required, because your framework should allow you to change the web service URL after the client has been generated.

If you encounter problems with the multiple XSD files you may use the provided merged wsdl whose name ends with `fully_embedded_xsd.wsdl`. It has all XSD definitions inline defined and thus does not include any XSD file.

10.8.1. Java Spring Web Services Client

See the directory `example-client-java` in the distributed zip file. See https://spring.io/projects/spring-ws for information about Spring Web Services.

10.8.2. Java JAX-WS Client

This chapter describes step by step how to generate the necessary classes from the provided WSDL

and its XSD files for a `javax.xml.ws.WebServiceClient`. It uses for convenience a Maven plugin `org.codehaus.mojo:jaxws-maven-plugin` but of course you may call directly the `wsimport` executable of your JDK.

Step 1: Prepare Maven POM

Add the jaxws-maven-plugin to your pom.xml

```
    <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>jaxws-maven-plugin</artifactId>
        <version>2.4.1</version>
        <executions>
            <execution>
                <id>client_jaxws</id>
                <goals>
                    <goal>wsimport</goal>
                </goals>
                <configuration>
                    <wsdlDirectory>src/main/resources/META-INF/wsdl</wsdlDirectory>
                    <wsdlFiles>
                        <wsdlFile>TCConnect_v2_6_0_soap_1_2.wsdl</wsdlFile>
                    </wsdlFiles>
                    <packageName>com.timocom.tcconnect.client</packageName>
                    <vmArgs>
                        <vmArg>-Djavax.xml.accessExternalSchema=all</vmArg>
                    </vmArgs>
                    <sourceDestDir>target/generated-sources/java</sourceDestDir>

<wsdlLocation>http://localhost/timocom/TCConnect_v2_6_0_soap_1_2.wsdl</wsdlLocation>
                    <extension>true</extension>
                </configuration>
            </execution>
        </executions>
    </plugin>
```

Step 2: Copy WSDL and XSD files into your project

Copy the files from the unpacked directory `wsdl` to your project's wsdlDirectory (see pom.xml) `src/main/resources/META-INF/wsdl`.

Step 3: Create the XML catalog file

Create a file called `jax-ws-catalog.xml` under `src/main/resources/META-INF` with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog
   xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
   prefer="system">
   <system systemId="http://localhost/timocom/TCConnect_v2_6_0_soap_1_2.wsdl"
           uri="wsdl/TCConnect_v2_6_0_soap_1_2.wsdl"/>
</catalog>
```

This serves as a redirect to the local WSDL instead of connecting to a real URL. Note that TIMOCOM does not provide the complete WSDL with all its XSDs online.

Step 4: Java code snippet using the generated `javax.xml.ws.WebServiceClient`

```
// Get the correct stage dependent value, here it is just hard-coded:
final String timocomUrl = "https://ws-test.timocom.com/tcconnect/ws_v2/soap1_2";

final TCCClientService timocomClient =
        new TCCClientService(null, new javax.xml.namespace.QName(
"http://webservice.timocom.com/schema/connect/v2", "TCCClientService"));

final TCCClientPortType port = timocomClient.getPort(TCCClientPortType.class);
final javax.xml.ws.BindingProvider bindingProvider = (javax.xml.ws.BindingProvider
)port;
bindingProvider.getRequestContext()
               .put(BindingProvider.ENDPOINT_ADDRESS_PROPERTY, timocomUrl);

final GetReferencedDataRequest referencedDataRequest = new GetReferencedDataRequest();
referencedDataRequest.setAuthentication("PRINCIPAL_NAME:PASSWORD");
referencedDataRequest.setPayload(new FilterType());
referencedDataRequest.setVersion("2.6.0");

final GetReferencedDataResponse referencedDataResponse = port.getReferencedData
(referencedDataRequest);

// further code goes here
```

10.9. Implementing a Response Handler

A response handler should first check if the `status` of the response is the expected one in consideration of the request type (Get, Find, Store, Delete)

In a second step the message type should be evaluated, in particular its `messageKey` field. If you look at Message Key you will find a bunch of (currently) possible keys which are represented by a soft enum.

10.9.1. First Step: Considering the Expected Response Status

The following Java example snippet implements a response handler with a method which checks if the response is considered successful or should raise an exception.

Possible Simple Exception Class Hosting the Messages of the Response

```
package com.timocom.tcconnect.client.shared;

import java.text.MessageFormat;
import java.util.List;

import com.timocom.tcconnect.client.model.v2.MessageType;
import com.timocom.tcconnect.client.model.v2.StatusEnumType;

public class ConnectException extends RuntimeException {

    private Origin origin;
    private String requestRef;
    private String status;
    private List<MessageType> messages;

    public ConnectException(final String requestRef, final String status,
        List<MessageType> messages) {
        super(toMessage(status));
        this.requestRef = requestRef;
        this.origin = toCause(status);
        this.messages = messages;
        this.status = status;
    }

    private static Origin toCause(final String status) {
        return StatusEnumType.INTERNAL_SERVER_ERROR.value().equals(status) ?
            Origin.SERVER : Origin.CLIENT;
    }

    private static String toMessage(final String status) {
        return MessageFormat.format("{0} ({1} error)", status, toCause(status));
    }

    /**
     * Omitted getters and setters
     */


    public enum Origin {
        CLIENT,
        SERVER
    }
}
```

Of course, you may want to create a more detailed exception type hierarchy based on the status/messageKey combination.

Possible Corresponding Response Handler Class

```
package com.timocom.tcconnect.client.shared;

import java.util.Collections;

import com.timocom.tcconnect.client.model.v2.DeleteEntityResponseType;
import com.timocom.tcconnect.client.model.v2.ReadResponseType;
import com.timocom.tcconnect.client.model.v2.MessageType;
import com.timocom.tcconnect.client.model.v2.StatusEnumType;
import com.timocom.tcconnect.client.model.v2.StoreEntityResponseType;
import com.timocom.tcconnect.client.model.v2.TCConnectResponseType;

public class ResponseHandler {

    public void checkResponse(final TCConnectResponseType response) {
        if (!isResponseStatusExpected(response)) {
            throw new ConnectException(
                response.getRequestRef(),
                response.getStatus(),
                response.getMessages() != null ?
                    response.getMessages().getMessages() :
                    Collections.<MessageType>emptyList());
        }
    }

    private boolean isResponseStatusExpected(TCConnectResponseType response) {
        if (response == null) {
            return false;
        }
        if (ReadResponseType.class.isAssignableFrom(response.getClass())) {
            return (StatusEnumType.OK.name().equals(response.getStatus())
                || StatusEnumType.NOT_FOUND.name().equals(response.getStatus()));
        }
        if (DeleteEntityResponseType.class.isAssignableFrom(response.getClass())) {
            return StatusEnumType.DELETED.name().equals(response.getStatus());
        }
        if (StoreEntityResponseType.class.isAssignableFrom(response.getClass())) {
            return StatusEnumType.STORED.name().equals(response.getStatus());
        }

        return false;
    }
}
```

10.9.2. Optional Second Step: Incorporate Message Type's Message Key

The message key within each message is a soft enum (see above) and may be translated at runtime via the Reference Data Endpoint.

So you may log the translated message to the client's log file or just use it during development for a better explanation of the returned message.

We first outline a service class which enables you to translate a soft enum:

RefereneDataService Providing Method for Translation of a MessageKey

```
package com.timocom.tcconnect.client.referencedata;

import java.util.List;
import java.util.Optional;
import java.util.function.Function;

import javax.inject.Inject;

import com.timocom.tcconnect.client.model.v2.ReferencedDataType;
import com.timocom.tcconnect.client.model.v2.ReferencedValue;
import com.timocom.tcconnect.client.model.v2.ReferencedValueListType;
import com.timocom.tcconnect.client.model.v2.TranslationListType;
import com.timocom.tcconnect.client.model.v2.TranslationType;

public class ReferenceDataService {

    @Inject
    ReferenceDataRepository repository; ①

    /**
     * You should cache the result
     *
     * @param enumerationValue
     * @param isoLanguageCode
     * @return
     */
    public String translateMessageKey(
        final String enumerationValue, final String isoLanguageCode) {
        return translate(
            enumerationValue,
            isoLanguageCode,
            ReferencedDataType::getValidationMessages); ②
    }

    private String translate(
        final String enumerationValue,
        final String isoLanguageCode,
        final Function<ReferencedDataType,ReferencedValueListType> listGetter) {
```

```
        final List<ReferencedValue> values = getReferencedValues(listGetter);

        final Optional<ReferencedValue> referenceValue =
            values.stream().filter(it ->
                enumerationValue.equals(it.getValue())).findFirst();

        if (!referenceValue.isPresent()) {
            return null;
        }

        final TranslationListType translations =
            referenceValue.get().getTranslations();

        final Optional<String> translation = translations.getTranslations().stream()
            .filter(t -> isoLanguageCode.equals(t.getLang()))
            .map(TranslationType::getValue).findFirst();
        return translation.orElse(null);
    }

    private List<ReferencedValue> getReferencedValues(
        final Function<ReferencedDataType, ReferencedValueListType> listGetter) {
        final ReferencedDataType referencedData =
            this.repository.getReferenceData(); ①

        return listGetter.apply(referencedData).getReferencedValues();
    }
}
```

① This repository is injected by dependency injection framework and responsible for sending the `GetReferenceDataRequest` to the *API* and then return the payload.

② The last parameter is a Java 8 method reference.

The following snippet shows how you may evaluate the message keys of each returned message:

How to Translate the Returned MessageKeys:

```
package com.timocom.tcconnect.client.offer.cargo;

import static java.lang.System.out;

import javax.inject.Inject;

import com.timocom.tcconnect.client.model.v2.CargoOfferType;
import com.timocom.tcconnect.client.model.v2.MessageType;
import com.timocom.tcconnect.client.referencedata.ReferenceDataService;
import com.timocom.tcconnect.client.shared.ConnectException;

public class CargoOfferService {

    @Inject
    ReferenceDataService referenceDataService;

    @Inject
    CargoOfferRepository repository;


    public void createOffer() {
        CargoOfferType cargoOffer = createYourOffer();
        try {
            this.repository.storeOffer(cargoOffer);
        } catch (ConnectException exception) {

            for (final MessageType message : exception.getMessages()) {
                // print all messages:

                out.println("Message level: " + message.getMessageLevel());
                out.println("Message key = " + message.getMessageKey());

                out.println("Message translation: "
                    + this.referenceDataService.translateMessageKey(
                        message.getMessageKey(), "de"));
                out.println("Optional property path: " + message.getPropertyPath());
                out.println("Optional log message: " + message.getLogMessage());
            }
        }
    }

}
```

10.10. From Testing to Production

This section describes how the transition from the test system to the production system is done.

Before you can switch to the production system, you must have completed your tests on the test

system successfully. There can be no further tests on the production system. All data which you send to the production system will be processed and, given there are no errors, be published immediately.

If you feel ready for switching to the production system, contact TIMOCOM and agree upon a start date. We will prepare your account and provide you with a new password for the production system.

You will also have to tell us which customer ids (TIMOCOM customer numbers) you want to access on the production system. If your (corporate) company consists of several subsidiaries, each of these probably has its own customer id. We will activate customer ids only on request, and each offer you send must be related to one of these ids in order to be accepted by the *API*.

If you will use “IP and password” authentication do not forget to inform TIMOCOM about the IP address which should have access to the web service on the production system. Do these addresses differ from the test system?

On the agreed date, you should start sending a small amount of requests at first. Inform TIMOCOM that you have sent your first requests. We will review our log files and confirm that everything looks good from our side. In the meantime, you can have a look at the web interface of the *Apps* and see if your offer was published as you expected.

If the first requests are looking good for both sides, you can commence regular operation of your web service client.

[1] So far *Stacking & lifting* has no category representation neither in the *Apps* nor the *API* backend, i.e. in the *Apps* view it is just an emphasized subgroup representation of *vehicle equipment*.

Chapter 11. History

API version	Server version	Date	Comment
2.6.0	6.3	2024-01	Fix documentation (about modifying/withdrawing offers via the web interface)
2.6.0	5.8	2023-05	Fix documentation (example StoreContactPerson)
2.6.0	5.8	2023-04	Fix documentation (example DeleteCargoOffer)
2.6.0	5.6	2022-09	New endpoints `StorePriceProposalRequest`, `GetPriceProposalRequest` and `FindPriceProposalsRequest` to handle store, read and find operations for price proposals
2.4.1	5.0	2021-12	New field `distanceInKilometres` in returned CargoOfferTypes
2.4.1	5.0	2021-12	Allow just one post-code character (prefix string) in outside search filter
2.4.0	4.20	2021-06	New endpoints LookupCargoOffersRequest / LookupTruckOffersRequest for looking up offers by ID
2.3.0	4.17	2020-09	Re-group vehicleProperties and introduce 3 new categories
2.3.0	4.17	2020-09	Deprecation field checks (Outside search filter): vehicleBodies, vehicleTypes
2.3.0	4.17	2020-09	Deprecation field check for offer: vehicleType(s), loadingPlace.date, vehicleEquipments, vehicleBody
2.2.7	4.15	2020-05	Introduce multi-dates for freight loading places (earliestLoadingDate, latestLoadingDate)
2.2.6	4.14	2020-05	Introduce embedded contact persons in offers.
2.2.6	4.14	2020-05	Field CargoOfferType.otherBodyPossible is now deprecated.
2.2.5	4.13	2020-01	Add contact channels field to offer type.
2.2.4	4.12	2019-11	“Outside Search”: Add search by date interval or individual dates.
2.2.3	4.10	2019-08	Replace section about generating client code with Visual Studio by Java JAX-WS dynamic proxy example.
2.2.2	4.9	2019-04	“Outside Search”: Each result offer contains a deep link to its offer details in the TIMOCOM *Smart Logistics System*
2.2.2	4.9	2019-04	“Outside Search”: Search interval: Allow setting of a left lower bound.
2.2.1	4.8	2019-02	“Outside Search”: Allow sorting of the result set by the creation date.
2.2.0	4.7	2018-09	More generic `vehicleProperties` replace dedicated fields `vehicleBody`, `vehicleType`, `vehicleEquipments`. Allow more than one vehicle body.

API version	Server version	Date	Comment
2.1.1	4.5	2018-05	Introduce PhoneNumberType and validate mandatory phone field in ContactPersonType
2.1.0	4.4	2018-03	Introduce “Outside Search”
2.0.2	4.3	2017-11	Add section “Generate Client Interface Classes from WSDL-Files With Visual Studio”
2.0.2	4.3	2017-11	Add section “Migrating from API version 1.x”
2.0.2	4.3	2017-11	New field paymentDueWithinDays for CargoOfferType.
2.0.1	4.3	2017-11	Additional vehicle equipment values (SECOND_DRIVER, ACCESS_RAMP, ESCORT_VEHICLE_TYPE_3, ESCORT_VEHICLE_TYPE_4, MEAT_HOOK, WOOD_STANCHIONS, TARPAULIN_COVER) and vehicle body values (SWAP_BODY_TRUCK, TRACTOR_UNIT).
2.0.1	4.2	2017-11	Section about security level “Static Password Token”
2.0.0	4.0	2017-07	First version for new v2 API