N-410

Camiuns (NETS) Interface Specification

Status as at 01.08.2026, v1.3

Versioning:

Version	Date	Section	Change
1.0	24.02.2024		First published version
1.1	01.09.2024	various	List XSD Message Schemas in Table Add validation error code in TollDeclarationResponse Add new issue code for tollDeclarationId is unknown in Acknowledge Add new Scenario TollDeclaration (CORRECTION_VIN) Removed issueText from Acknowledge Changed Example in BAZG Message Definition (messageDateTime in UTC, removed Signature und Certificate Information Add further Information TollDeclaration (Regular Message) to the field declarationPeriod. Add acceptance URL BAZG B2B-Hub-Connectivity
1.2	01.03.2025	5.4 6.13.2	Camiuns replies with a REGISTARTION_STATUS for every REGISTARTION_BEGIN notice
1.3	01.08.2026	 2.3. / 2.4	New Validation Error New Endpoints v3

Table of Contents

1. Purpose and audience 5

2. NETS Interaction with BAZG 6
2.1. BAZG Interaction possibilities 6
2.2. BAZG B2B-Hub-Access Point 7
2.2.1. Design Approach 7
2.2.2. Security 7
2.2.2.1. Transport 7
2.2.2.2. Authentication 7
2.2.2.3. Authorization 8
2.2.2.4. Integrity 8
2.3. Partner API v3 (Businesspartner) 8
2.3.1. Changes since API v2 8
2.3.2. Open API Specification 8
2.3.3. Endpoints 13
2.3.4. Usage 14
2.3.4.1. Message Identifier 14
2.3.4.2. Usage Variants 14
2.3.5. Topics 15
2.3.6. Generic Header Attributes 15
2.3.7. Functions 16
2.3.7.1. Fetch message(s) 16
2.3.7.1.1. List message(s) 16
2.3.7.1.2. Fetch single message 16
2.3.7.1.3. Fetch next message 17
2.3.7.2. Submit message 17
2.4. BAZG B2B-Hub-Connectivity Setup 18
2.4.1. Registration business partner BAZG 18
2.4.2. Complete user registration 18
2.5. Access Token 19
2.5.1. Overview 19
2.5.2. Access Token refresh 20
2.5.2.1. Example request (with curl): 20
2.5.3. Connectivity-Test 20
2.6. BAZG Message Definition 20
2.6.1. Message Structure 21
2.6.2. Example 21
2.6.2.1. Nets Notice 21
2.7. Data Integrity 22
2.7.1. Enveloped Signature 23
2.7.2. Certificates 24
2.7.2.1. General 24
2.7.2.2. BAZG 24
2.7.2.3. NETS provider 25
2.7.3. Glossary 25
2.8. Transport level / Functional level 26
2.8.1. Example 1: Error at transport level 26
2.8.2. Example 2: Error on functional level 26

3. NETS Information Model 27
3.1. TollDeclaration (Message + Interaction Pattern) 27
3.1.1. TollDeclaration Request 27
3.1.1.1. TollDeclaration Request Types (Regular Declaration) 28
3.1.1.2. TollDeclaration Request Types (Manual Correction) 28
3.1.2. TollDeclaration Response 29
3.1.2.1. TollDeclarationResponse Response Types 30
3.1.2.2. Technical errors 30
3.1.3. Acknowledge 31
3.1.3.1. BAZG → Provider 31
3.1.3.2. Provider → BAZG 31
3.1.4. Interaction Pattern 31
3.1.4.1. Regular Declaration Message 31
3.1.4.2. Manual Correction Message 31
3.1.4.3. Message Handling Rules 32

3.1.4.3.1. for regular declaration messages: 32
3.1.4.3.2. for manual correction messages: 32
3.1.5. Correlation and Message Processing 33
3.1.5.1. Correlation (to one or part of) a regular message 33
3.1.5.2. Correlation to regular messages and manuel corrections 34
3.1.5.3. Correlation to exactly one regular message 34
3.1.5.4. Message Processing 34
3.2. Notice (Message + Interaction Pattern) 34
3.2.1. Notice Request 34
3.2.2. Acknowledge 35
3.2.3. Interaction Pattern 35
3.2.4. Correlation 35

4. NETS Conversation Handling 35
4.1. TollDeclarations 36
4.1.1. Provider → BAZG 36
4.1.2. BAZG → Provider 37
4.2. Notices 38
4.2.1. Provider → BAZG 38
4.2.2. BAZG → Provider 38

5. NETS Messages 39
5.1. Message 39
5.2. Message Content Header 39
5.3. Message Content Body 39
5.4. Notice 40
5.5. TollDeclaration (Regular Message) 41
5.6. TollDeclaration (Manual Correction Message) 43
5.6.1. Manual Declaration Reasons 46
5.7. TollDeclarationResponse 50
5.7.1. ResponseReasons 51
5.7.2. ResponseReasonsText 53
5.7.3. Validation Error Codes 53
5.8. Acknowledge 54
5.8.1. Issue Codes 55
5.9. Schemas 55
5.9.1. General Message Schema Files 55
5.9.2. Message Content Schema Files 55
5.9.3. Versions 56

6. NETS Messages-Exchange Szenarios 56
6.1. TollDeclaration (OK) 56
6.1.1. With result ACCEPTED 56
6.2. TollDeclaration (CORRECTION_JOURNEY) 56
6.2.1. Preconditions 57
6.2.2. Sequence diagram 57
6.3. TollDeclaration (CORRECTION_TRACK) 57
6.3.1. Preconditions 58
6.3.2. Sequence diagram 58
6.4. TollDeclaration (CORRECTION_TRAILER) 58
6.4.1. Preconditions 58
6.4.2. Correction timespan 59
6.4.3. Sequence 59
6.5. TollDeclaration (DEADLINE_MISSED) 59
6.5.1. With result REFUSED 59
6.6. TollDeclaration (DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS) 60
6.6.1. With result REFUSED 60
6.7. TollDeclaration (DECLARATION_ID_NOT_UNIQUE) 60
6.7.1. With result REFUSED 60
6.7.1.1. resending with result ACCEPTED 61
6.8. TollDeclaration, Notice (ISSUER_ID_UNKNOWN) 61
6.8.1. With result NOK (1002) 61
6.8.1.1. resending with result ACCEPTED 62
6.9. TollDeclaration (NO_REGISTRATION_FOR_VIN) 62
6.9.1. TollDeclaration (NO_REGISTRATION_FOR_VIN - vin not found) 62

6.9.1.1. With result ACCEPTED ERROR / REFUSED 62
6.9.1.2. ... manual correction with result ACCEPTED 63
6.9.1.3. ... manual correction with result REFUSED 63
6.9.1.4. ... correction not timely received 64
6.10. TollDeclaration (CORRECTION_VIN) 64
6.10.1. Background 64
6.10.2. Scenarios 65
6.10.2.1. Data Entry Error 65
6.10.2.1.1. Behaviour on day 1 (Day of the VIN correction) 65
6.10.2.1.2. Behaviour on day 2 (Day after the VIN correction) 66
6.10.2.2. Vehicle Registration Errors 66
6.10.2.3. Toll Declaration Lifecycle in case of VIN Correction (Example) 67
6.11. TollDeclaration, Notice (SIGNATURE_INVALID) 69
6.11.1. With result NOK (1001) 69
6.11.2. resending with result ACCEPTED 69
6.12. TollDeclaration, Notice (VALIDATION_ERROR) 70
6.12.1. TollDeclaration, Notice (VALIDATION_ERROR Schema Level) 70
6.12.1.1. With result NOK (1000) 70
6.12.2. TollDeclaration (VALIDATION_ERROR Application Level) 71
6.12.2.1. With result REFUSED 71
6.13. Notice (REGISTRATION) 72
6.13.1. Notice (REGISTRATION_BEGIN) 72
6.13.2. Notice (REGISTRATION_STATUS) 72
6.13.3. Notice (REGISTRATION_END) 73
6.13.3.1. Notice (REGISTRATION_END - Termitated by BAZG) 73
6.13.3.2. Notice (REGISTRATION_END - Terminated by the holder via NETS provider) 73

1. Purpose and audience

This specification describes the technical requirements for providers of the National Electronic Toll Service (NETS) who integrate their IT system with BAZG or the backend system CAMIUNS for the purpose of providing services. The description covers the entire process from obtaining the interface, the underlying information model and the message exchange from a functional and technical point of view.

The technical messages to be used are listed in tabular form.

Message Type	Description	Links

Notice	Notices are used to report the registration or unregistration for vehicles under service	• Domain Model
		  ○ Notice
		• Scenarios
		  ○ Notice (REGISTRATION)
		  ○ Notice (SERVICE_ERROR)

TollDeclaration	TollDeclarations contain regular or manual declarations for a vehicle under service	• Domain Model
		  ○ TollDeclaration (Regular Message)
		  ○ TollDeclaration (Manual Correction Message)
		• Scenarios
		  ○ TollDeclaration (OK)
		  ○ TollDeclaration (DEADLINE_MISSED)
		  ○ TollDeclaration (DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS)
		  ○ TollDeclaration (DECLARATION_ID_NOT_UNIQUE)
		  ○ TollDeclaration, Notice (ISSUER_ID_UNKNOWN)
		  ○ TollDeclaration (NO_REGISTRATION_FOR_VIN)
		  ○ TollDeclaration, Notice (SIGNATURE_INVALID)
		  ○ TollDeclaration, Notice (VALIDATION_ERROR)

TollDeclarationResponse	Every TollDeclaration will be replied by BAZG with an TollDeclarationResponse.	• Domain Model
		  ○ TollDeclarationResponse
		• Scenarios
		  ○ s. TollDeclaration

Acknowledge	Every Message has to be answered. Some Messages arent answered by an TollDeclarationRe	• Domain Model ○ Acknowledge

Message Type	Description	Links
	sponse. For those cases, the Acknowledge message type is used.	

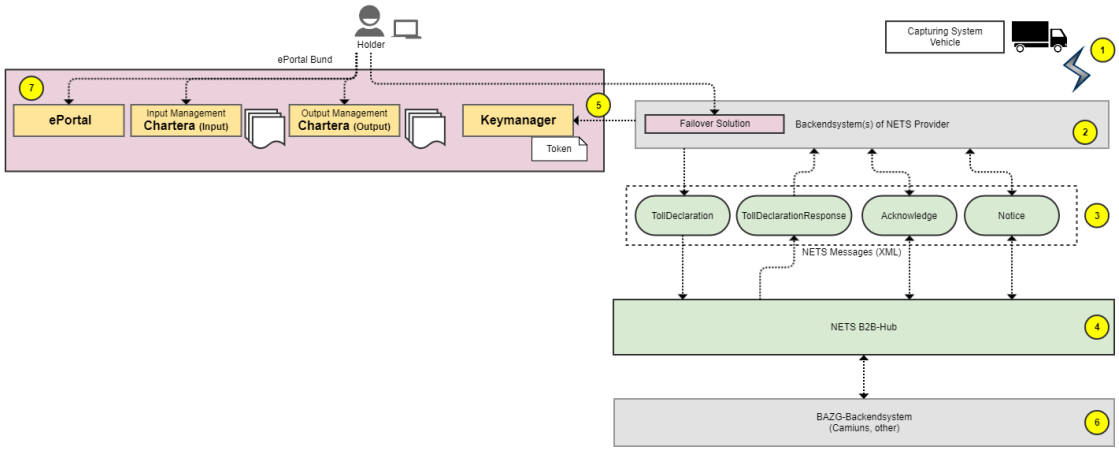
Concrete implementations can be found in section: Schemas. The target audience of this document is the IT departments of the business partners or their architects and developers.

The NETS interface specification will all be all the same for the different groups of NETS providers. Therefore, whenever NETS providers are mentioned in the document it is referred to both national NETS provider (NNA) as well as authorized NETS providers (ZNA).

2. NETS Interaction with BAZG

2.1. BAZG Interaction possibilities

For the use of the infrastructure and connection to individual backend systems BAZG provides a common interface. To use the interface, a provider must register once as a BAZG business partner and be assigned the “HVC provider NETS” business partner role (this role cannot be obtained via self-service, it is assigned by the BAZG). The illustration below shows the principle of how external systems interact with BAZG via "ePortal Bund" and "NETS B2B Hub".



1	A vehicle under service captures its journey and transmits the milage over the air to the backend system of a NETS Provider
2	The NETS provider's backend system must fulfil the following requirements: 1) It must be able to read the mileage of its contracted vehicles 2) This data must be processed by the BAZG prior to transmission in accordance with the specification 3) A failure solution must be provided for all manual interventions by a vehicle owner. These manual corrections can also result in messages that can be exchanged with the BAZG 4) Communication with the BAZG must take place via the B2B Hub. To be able to

	communicate successfully with the B2B Hub, a token lifecycle must be implemented using a key manager
3	NETS messages are used for data exchange with the BAZG. NETS messages are signed XML files.
4	The central system for asynchronous data exchange between a NETS provider and the BAZG is the B2B Hub. It offers a business partner API via which NETS messages can be sent and received.
5	The providers backend system(s) obtail the initial API Token via self-service portal. For the token refresh, that is necessary to exchange NETS messages, the authorization token can be obtained from a keymanager.
6	The messages are processed in the BAZG backend system. This is also where new messages are created, which are provided to a business partner via the B2B Hub.
7	A LSVA holder can retrieve and submit various documents (assessment decisions, invoices, etc.) from the BAZG via the ePortal.

2.2. BAZG B2B-Hub-Access Point

2.2.1. Design Approach

Technically, the backend systems of BAZG should be dependent on the IT systems of the business partners:

- No adaptation to backend system BAZG when new providers are added.
- No error handling on backend system BAZG if the providers systems are not available or cannot be reached.

Technically, this is solved with the B2B-Hub, which abstracts a message exchange between the two systems via a clearly defined interface.

Dataflow	Backend System BAZG	Business Partner
incoming BusinessPartner → Backend-System BAZG	Receives messages	Sends messages (push)
outgoingBackend-System BAZG → Business Partner	Publishes messages	Fetches messages (pull)

2.2.2. Security

2.2.2.1. Transport

Transport is protected by means of HTTPS. The exchange of the Nets messages shall be done using the Transport Layer Security, TLS 1.3 protocol.

2.2.2.2. Authentication

The identity of the business partner is guaranteed by means of an Access Token. Such an Access Token is required to access the B2B-Hub. The application of the business partner has to renew the Access Token by its own using credentials (ClientId/ClientSecret).

2.2.2.3. Authorization

The B2B-Hub decides whether the business partner is authorized for the request based on its identity.

2.2.2.4. Integrity

The body of **all** messages (XML) are signed, which can be used to prove (see BAZG Message Signing):

- that the message was created by this NETS provider
- that the message has not been subsequently modified

2.3. Partner API v3 (Businesspartner)

2.3.1. Changes since API v2

Added partner API v3 with the following changes compared to v2 (note that v2 is unchanged)

- The GET /messages response now includes the following optional elements:
  - `<groupId>`
  - `<partnerTopic>`
- `partnerTopic` is now defined as a query parameter instead of a header value
  - The HTTP headers `partnerTopic` and `partner-topic` have been removed
- Removed Deprecated HTTP Headers
  - `partnerTopic`
  - `partner-topic`
  - `bpId` (use `bp-id` instead)
  - `messageType` (use `message-type` instead)
  - `Messageid` (use `message-id` instead)
- The deprecated query parameter `lastMessageID` (which was a fallback for the correct name `lastMessageId`) has been removed

2.3.2. Open API Specification

```
{
    "openapi": "3.1.0",
    "info": {
        "title": "Message Exchange Service",
        "description": "Rest API to exchange messages with business
partners"
    },
    "externalDocs": {
        "url": ""
    },
    "servers": [
        {
            "url": "",
            "description": "Generated server url"
        }
    ],
    "security": [
        {
            "OIDC": []
        }
    ],
    "paths": {
        "/api/partner/v3/messages/{messageId}": {
            "get": {
                "tags": [
                    "message-partner-v-3-controller"
                ],
                "summary": "Get message",
                "operationId": "getMessage",
                "parameters": [
                    {
                        "name": "messageId",
                        "in": "path",
                        "description": "Unique message identification as
UUID: cc7d5097-4d3f-4fff-af91-fd3680199642",
                        "required": true,
                        "schema": {
                            "type": "string",
                            "format": "uuid"
                        }
                    },
                    {
                        "name": "bp-id",
                        "in": "header",
                        "description": "Partner identification",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "content": {
                            "application/xml": {
                                "schema": {
                                    "type": "string",
                                    "format": "binary"
                                }
                            }
                        }
                    }
                }
            },
```

```
            "put": {
                "tags": [
                    "message-partner-v-3-controller"
                ],
                "summary": "Submits a new message",
                "operationId": "sendMessage",
                "parameters": [
                    {
                        "name": "messageId",
                        "in": "path",
                        "description": "Unique message identification as
UUID: cc7d5097-4d3f-4fff-af91-fd3680199642",
                        "required": true,
                        "schema": {
                            "type": "string",
                            "format": "uuid"
                        }
                    },
                    {
                        "name": "bp-id",
                        "in": "header",
                        "description": "Partner identification",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "message-type",
                        "in": "header",
                        "description": "Business type definition of the
message body",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    }
                ],
                "requestBody": {
                    "description": "XML message",
                    "content": {
                        "application/xml": {
                            "schema": {
                                "type": "string"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "200": {
                        "description": "OK"
                    }
                }
            }
        },
        "/api/partner/v3/messages": {
            "get": {
                "tags": [
                    "message-partner-v-3-controller"
                ],
                "summary": "Returns a list of messages for the given BP
(Business Partner), topicName and / or groupId",
                "operationId": "getMessages",
                "parameters": [
```

```
                    {
                        "name": "bp-id",
                        "in": "header",
                        "description": "Partner identification",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "topicName",
                        "in": "query",
                        "description": "Get only messages from given
topicName",
                        "required": false,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "groupId",
                        "in": "query",
                        "description": "Get only messages with given
groupId",
                        "required": false,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "lastMessageId",
                        "in": "query",
                        "description": "Get only messages which were
published after the given messageId",
                        "required": false,
                        "schema": {
                            "type": "string",
                            "format": "uuid"
                        }
                    },
                    {
                        "name": "partnerTopic",
                        "in": "query",
                        "description": "Partner Topic",
                        "required": false,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "size",
                        "in": "query",
                        "description": "Number of messages returned",
                        "required": false,
                        "schema": {
                            "type": "integer",
                            "format": "int32",
                            "default": 1000
                        }
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "content": {
```

```
                            "application/xml": {
                                "schema": {
                                    "$ref":
"#/components/schemas/messages"
                                }
                            }
                        }
                    }
                }
            }
        },
        "/api/partner/v3/messages/{messageId}/next": {
            "get": {
                "tags": [
                    "message-partner-v-3-controller"
                ],
                "summary": "Get next message",
                "operationId": "getNextMessage",
                "parameters": [
                    {
                        "name": "messageId",
                        "in": "path",
                        "description": "Unique message identification as
UUID: cc7d5097-4d3f-4fff-af91-fd3680199642",
                        "required": true,
                        "schema": {
                            "type": "string",
                            "format": "uuid"
                        }
                    },
                    {
                        "name": "bp-id",
                        "in": "header",
                        "description": "Partner identification",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "partnerTopic",
                        "in": "query",
                        "description": "Partner Topic",
                        "required": false,
                        "schema": {
                            "type": "string"
                        }
                    },
                    {
                        "name": "topicName",
                        "in": "query",
                        "description": "Get only messages from given
topicName",
                        "required": false,
                        "schema": {
                            "type": "string"
                        }
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "content": {
                            "application/xml": {
                                "schema": {
```

```
                                    "type": "string",
                                    "format": "binary"
                                }
                            }
                        }
                    }
                }
            }
        },
        "components": {
            "schemas": {
                "MessageResultDto": {
                    "type": "object",
                    "properties": {
                        "messageId": {
                            "type": "string",
                            "format": "uuid"
                        },
                        "messageType": {
                            "type": "string"
                        },
                        "groupId": {
                            "type": "string"
                        },
                        "partnerTopic": {
                            "type": "string"
                        }
                    }
                },
                "messages": {
                    "type": "object",
                    "properties": {
                        "message": {
                            "type": "array",
                            "items": {
                                "$ref":
    "#/components/schemas/MessageResultDto"
                            }
                        }
                    }
                }
            },
            "securitySchemes": {
                "OIDC": {
                    "type": "openIdConnect",
                    "description": "OAuth2-Authentication",
                    "openIdConnectUrl": ""
                }
            }
        }
    }
```


2.3.3. Endpoints

URL Production Environment	https://nets-b2bhub.bazg.admin.ch/declaration/api/v3/
URL Acceptance Environment	https://abn-nets-b2bhub.bazg.admin.ch/declaration/api/V3/

2.3.4. Usage

Endpoint	Method	Usage
/messages	GET	Returns a list of NETS messages for the specific business partner by bpId, topicName and groupId. The response (XML) contains a list of messageIds with the messageType of each NETS message.
/messages/{messageId}	GET	Returns the NETS message for the specified business partner by bpId and message by messageId. The response (XML) fulfils the corresponding XSD according to the specification of the NETS message.
/messages/{messageId}/next	GET	Returns a message for the specified business partner by bpId and/or topicName, which was published chronologically after the message with the specified messageId and is used to iterate over messages in chronological order. The response (XML) fulfils the corresponding XSD in accordance with the NETS message specification
/messages/{messageId}	PUT	Used to transmit a new message. The request body must be an XML that fulfils the corresponding XSD according to the NETS message specification

2.3.4.1. Message Identifier

The messageId is used to uniquely identify a message that is transmitted via the B2B Hub. This identification is set by the source system and corresponds to a UUID.
It must fulfil the following convention:

[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}

If a messageId is sent several times, the previously sent document is overwritten each time.

2.3.4.2. Usage Variants

A business partner can collect messages from the B2B Hub in two ways:

1. Retrieve a list of messages with /messages and then retrieve each individual message. As soon as all messages have been retrieved, retrieve the next messages with /messages?lastMessageId.
2. Fetch first message /messages?size=1 and then only iterate over all messages with /next until a 404 is reached.

Variant	List & single retreival	Iteration with /next
Initial request	`/messages?size=2` **Response**: <messages> <messageId>1</messageId> <messageId>2</messageId> </messages>	`/messages?size=1` **Response**: <messages> <messageId>1</messageId> </messages>

Variant	List & single retreival	Iteration with /next
Frist message	`/messages/1` **Response:** Header: messageId:1 Body: <xmlContent>	`/messages/1` **Response:** Header: messageId:1 Body: <xmlContent>
Next message	`/messages/2` **Response**: Header: messageId:2 Body: <xmlContent>	`/messages/1/next` **Response**: Header: messageId:2 Body: <xmlContent>
Next message	/messages?size=2&lastMessageId=2 **Response**: <messages> <messageId>3</messageId> <messageId>4</messageId> </messages> /messages/3 **Response**: Header: messageId:3 Body: <xmlContent>	`/messages/2/next` **Response**: Header: messageId:3 Body: <xmlContent>

2.3.5. Topics

By distributing the messages to different topics, we enable the counterpart to allow different software components to access "their" messages directly without the need for forwarding.

The following topics are currently implemented for the traffic duties:

Business Object	Topic	Description
Notice TollDeclaration (Regular Message) TollDeclaration (Manual Correction Message) TollDeclarationResponse Acknowledge	nets-tolldeclaration	all NETS messages share the same topic

Topics are only available for outgoing messages.

2.3.6. Generic Header Attributes

This header attributes should be part of every API Call.

Name	Required	Description	Example Value
bpId	yes	The Business Partner Id. Must match the Business Partner in the token.	`1234567891`

Name	Required	Description	Example Value
Authorization	yes	The token goes here	`Bearer eyJ0eX...`

2.3.7. Functions

2.3.7.1. Fetch message(s)

The BAZG provides 3 different API endpoints for querying and retrieving messages.

2.3.7.1.1. List message(s)

Verb	GET
Endpoint	`/messages/`
Description	Returns a list of messages for the specified business partner, topicName and / or groupId
Direction	BAZG → BP
Header Attributes	Generic header attributes
Path Parameter	
Query Parameter	

Name	Required	Description	Example Value
topicName	no	the name of the topic	nets-tolldeclaration
lastMessageId	no	Filter to only see messages after a previous MessageId	UUID
size	no	Number of messages which are returned: • default: 1000 • minium: 1 • maxium: 1000	1000

2.3.7.1.2. Fetch single message

Verb	GET
Endpoint	`/messages/{messageId}`
Description	Returns the NETS message for the specified business partner by bpId and message by messageId.
Direction	BAZG → BP
Header Attributes	Generic header attributes

Path Parameter	**Name**	**Type**	**Description**	**Example Value**
	messageId	UUID	s. Message Identifier	`12345678-1234-1234-1234-123456789012`
Query Parameter				

2.3.7.1.3. Fetch next message

Verb	GET
Endpoint	`/messages/{messageId}/next`
Description	Returns a message for the specified business partner and/or topicName, which was published chronologically according to the given MessageID.
Direction	BAZG→ BP
Header Attributes	Generic header attributes

Path Parameter	**Name**	**Type**	**Description**	**Example Value**
	messageId	UUID	s. Message Identifier	`12345678-1234-1234-1234-123456789012`
Query Parameter				

2.3.7.2. Submit message

Verb	PUT
Endpoint	`/messages/{messageId}`
Description	Transmits a new message
Direction	BP → BAZG
Header Attributes	Generic header attributes +

Name	Required	Description	Example Value
content-type	yes	the content type has to be application/xml	`application/xml`
messageType	yes	The messageType is used for the unique assignment of a sent message. This is set by the sender of the message. • ○ nets-acknowledge for Acknowledge messages	`nets-acknowledge`

			○ nets-notice for Notice messages ○ nets-manualtolldeclaration for TollDeclaration (Manual Correction Message) messages ○ nets-regulartolldeclaration for TollDeclaration (Regular Message) messages	

Path Parameter	**Name**	**Type**	**Description**	**Example Value**
	messageId	UUID	s. Message Identifier	`12345678-1234-1234-1234-123456789012`
Query Parameter				
Body	The HTTP body must be an XML (content type application/xml) The body is validated for XML conformity by the B2B Hub.			

In case the Message is successfully received and stored by the BAZG, the request will be answered with HTTP (200).

Remarks:

- This operation is idempotent. This means in case of timeout the NETS provider can safely repeat the operation, using exactly the same messageId, until the operation is successful.
- Replacing of a messages is not supported. BAZG will always use the first message successfully received for the messageId specified.

2.4. BAZG B2B-Hub-Connectivity Setup

Each counterpart, which communicates via B2B with LSVA III, must setup his system to operate with Camiuns.

2.4.1. Registration business partner BAZG

A provider must register once as a BAZG business partner and be assigned the “HVC provider NETS” business partner role (this role cannot be obtained via self-service, it is assigned by the BAZG, see "Complete user registration", Step 2).

2.4.2. Complete user registration

Step	Title	Description
1	Register User	All the necessary registration steps are described on the Internet: Registration for the digital services of Swiss Customs on ePortal (Onboarding) (admin.ch)

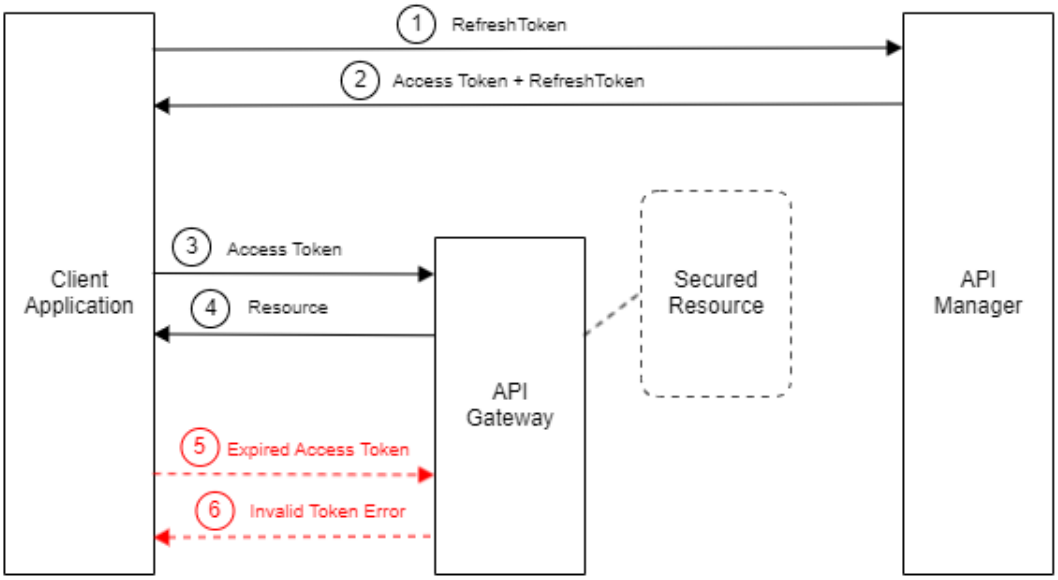
Step	Title	Description
		Registration must be carried out on both the production environment and the acceptance environment. https://eportal-a.admin.ch/start (acceptance) https://eportal.admin.ch/start (production)
2	Register for API Subscription	Get in touch with your contact at BAZG and have yourself authorized to use the NETS B2B Hub. This step must be carried out manually.
3	Subscribe API	After the BAZG registration, the API can be subscribed via self service URL Acceptance Environment: https://abn-nets-b2bhub.bazg.admin.ch/declaration/api/v3/ URL Production Environment: https://nets-b2bhub.bazg.admin.ch/declaration/api/v3/
4	Deposit generated tokens	The token generated in the ePortal is stored in the business partner's business software solution (customer system).

2.5. Access Token

2.5.1. Overview

A valid access token is required to call up a protected API. An access token is only valid for a certain period of time. A new access token can be requested using a refresh token. A new refresh token is also created in the process. This can be used to request another Access Token together with a Refresh Token.

To request a new Access Token, a Client ID and Client Secret are required in addition to the Refresh Token.



1	The client application requests a new access token from the API Manager with a refresh token.
2	If the refresh token is valid, the API Manager issues a new access token and a new refresh token and sends both to the client application.
3	The client application calls the API for a secured resource on the API gateway with a valid access token.
4	If the access token allows access to the secured resource, the API gateway forwards the request and returns the response.
5	The client application calls the API for a secured resource with an expired access token.
6	The API gateway rejects the request with an "Invalid Token Error".

2.5.2. Access Token refresh

To avoid a manual creation of a new set of tokens every 24 hours via API Selfcare, the automatic Access Token Refresh shall be implemented as part of the registration process on the customer system.

URL of the refresh-API from the acceptance-environment	https://keymanager-npr.api.admin.ch/token/v2
URL of the refresh-API from the production-environment	https://keymanager.api.admin.ch/token/v2

2.5.2.1. Example request (with curl):

```
curl -k -u <customerKey>:<secret> -d
"grant_type=refresh_token&refresh_token=<refreshToken>" -H "Content-Type:
application/x-www-form-urlencoded" <refresh-api-url>
```

The response is a new token, which can be used for transmission afterwards:

```
{"access_token":"{Bearer_token_goes_here}","refresh_token":"{Refresh_token
_goes_here}","scope":"default","token_type":"Bearer","expires_in":86400}
```

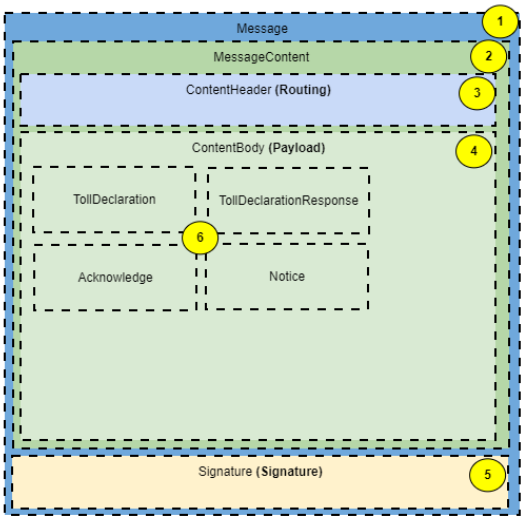
If the bearer token expires, a new bearer token can be generated from the response with the refresh token.

2.5.3. Connectivity-Test

The connectivity test consists of sending a message from the customer system to the BAZG. When the message arrives at the BAZG, the connectivity test is considered passed.

2.6. BAZG Message Definition

A message has a generic structure and allows the definition of all information necessary for the exchange. During message exchange the message as described herafter is encloased in the datafield "body". See also BAZG B2B-Hub-Access Point.



1	The overall generic structure of a message. A message consists of 2 components: • MessageContent • Signature
2	A MessageContent consists of 2 parts: • ContentHeader • ContentBody
3	The ContentHeader contains information about: • Id of the message • Sender of the message • Receiver of the message • Message date • Type of message content
4	The ContentBody contains the specific message (defined by a specific MessagType).
5	The signature ensures the authenticity of the MessageContent (s. BAZG Message Signing).
6	A specific business context is defined by it's information model. Different message types can be used to describe the model.

2.6.1. Message Structure

The structure and characteristics are listed in chapter NETS Messages.

2.6.2. Example

2.6.2.1. Nets Notice

Below an example of a complete message (header + content) of a message of type notice.

```
<?xml version="1.0" encoding="UTF-8"?>
<message xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
    <messageContent>
        <contentHeader>
            <messageId>7a5a323c-6ec6-4889-85af-05cf27351d99</messageId>
            <messageDateTime>2024-01-29T13:11:01.316Z</messageDateTime>
            <informationSenderId>
                <issuerId>1234567891</issuerId>
            </informationSenderId>
            <informationRecipientId>
                <issuerId>1000006447</issuerId>
            </informationRecipientId>
        </contentHeader>
        <contentBody>
            <notice>
                <noticeId>1706530261316</noticeId>
                <noticeType>REGISTRATION_BEGIN</noticeType>
                <dateTimeWhenRecorded>2024-01-
29T13:11:01.316Z</dateTimeWhenRecorded>
                <recordedBy>PROVIDER</recordedBy>
                <registrationBegin>
                    <vin>XLRASH4300G232849</vin>
                    <tollDomain>CH</tollDomain>
                    <effectiveDate>2024-01-29</effectiveDate>
                </registrationBegin>
            </notice>
        </contentBody>
    </messageContent>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:SignedInfo>
            <ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
            <ds:SignatureMethod
Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
            <ds:Reference URI="">
                <ds:Transforms>
                    <ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                    <ds:Transform
Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
                </ds:Transforms>
                <ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>

<ds:DigestValue>b3HAFHg5pQoRHx4jbphBdw00sHKSx7oSG00Q4bK8+LE=</ds:DigestVal
ue>
            </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>{here goes signature value}</ds:SignatureValue>
        <ds:KeyInfo>
            <ds:X509Data>
                <ds:X509Certificate>{here goes
certificate}</ds:X509Certificate>
            </ds:X509Data>
        </ds:KeyInfo>
    </ds:Signature>
</message>
```

2.7. Data Integrity

All messages exchanged between the business partner and BAZG shall be signed. BAZG will archive this message including signature in order to be able to prove:

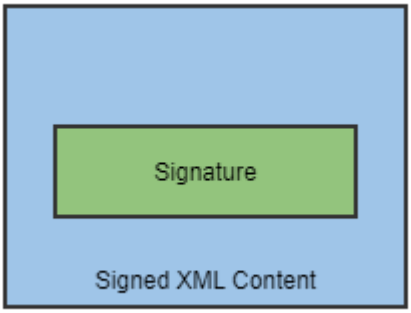
- that the message in the archive was actually created by a specific provider (information origin)
- that the message in the archive has not been subsequently changed (information integrity)

The digital signatures of the NETS messages shall be according to the W3C standard "XML Signature Syntax and Processing Version 1.1" (W3C Recommendation 11 April 2013, https://www.w3.org/TR/2013/REC-xmldsig-core1-20130411). From that standard the following options shall be used:

- Canonicalization of the XML document, Identifier for canonical XML 1.0 (omits comments): http://www.w3.org/2006/12/xml-c14n11
- Signature with envelope, Identifier: http://www.w3.org/2000/09/xmldsig#enveloped-signature
- For the <KeyInfo> element, the option <X509Data> (Identifier http://www.w3.org/2000/09/xmldsig#X509Data) with exactly one attribute of type <X509SubjectName> and <X509Certificate>
- Signature algorithm PKCS#1 v1.5 with digest sha256 (at least), identifier: http://www.w3.org/2001/04/xmldsig-more#rsa-sha256

2.7.1. Enveloped Signature

Means that a <signature> element will be added to the part of the message to be signed. We will sign the whole message, which will embed the signature into the message.



The signed message has the following structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<Message>
    <MessageContent>
        <ContentHeader/>
        <ContentBody/>
    </MessageContent>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:SignedInfo>
            <ds:CanonicalizationMethod
Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
            <ds:SignatureMethod
Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha512"/>
            <ds:Reference URI="">
                <ds:Transforms>
                    <ds:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                    <ds:Transform
Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
                </ds:Transforms>
                <ds:DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>

<ds:DigestValue>ubGfhen7JeSRL0BYVMAce6Do6sBuzx/B6ZcW9mQVdoQ=</ds:DigestVal
ue>
            </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>
            ...
        </ds:SignatureValue>
        <ds:KeyInfo>
            <ds:X509Data>
                <ds:X509Certificate>
                    ...
                </ds:X509Certificate>
                <ds:X509SubjectName>...</ds:X509SubjectName>
            </ds:X509Data>
        </ds:KeyInfo>
    </ds:Signature>
</Message>
```

2.7.2. Certificates

2.7.2.1. General

The used certificates shall be according X.509. The length of the RSA keys in the X.509 certificates shall be at least 2048 bits.

2.7.2.2. BAZG

The currently used BAZG XML document signature certificate shall be published on the BAZG webpage,

Documents
Insert Links to online available documents

2.7.2.3. NETS provider

The NETS provider shall provide the BAZG with the necessary information to validate the message signatures.

This can either be in the form of a public key or a certificate of his choice.

The NETS provider is responsible for notifiying the BAZG if the security of one of the certificates can no longer be guaranteed or if he wants to renew his signing key.

2.7.3. Glossary

Parameter	Applicable Specification	Description
CanonicalizationMethod	http://www.w3.org/2006/12/xml-c14n11	The XML syntax allows a number of options (e.g. single instead of double quotes for attribute values, the order of attributes in a tag, places with empty (or blank) spaces are irrelevant) which can result in a physically different but logically identical document. The purpose of normalization is to define an algorithm that reliably and repeatably transforms any XML document into its canonical (simplest, normalized) form. If the same algorithm is applied to 2 physically different documents, they can then be compared in logical form. The CanonicalizationMethod is applied to the SignedInfo element, not to the data. The data must be transformed with the algorithms listed under Transform before the digest can be calculated.
Reference		Points to the resources (data) that must be signed. In our case, the signature is not restricted to individual resources, so the reference URI is empty.
Transformation	http://www.w3.org/2000/09/xmldsig#enveloped-signature http://www.w3.org/2006/12/xml-c14n11	The following transformation algorithms are applied to the data before the digest is calculated: • Canonicalisation • Enveloped Signature
DigestMethod	http://www.w3.org/2001/04/xmlenc#sha256	For performance reasons, public-key algorithms usually do not sign the entire document, but only a hash of the entire document. This summary is called a digest. The digest is applied via the URI listed under References. The digest is Base-64 encoded. This digest is then integrated into the SignedInfo element to be signed.
DigestValue		The calculated digest (the hash of the entire data).
SignatureMethod	http://www.w3.org/2001/04/xmldsig-more#rsa-sha512	Algorithm which is used to sign the normalized SignedInfo.
SignatureValue		The base-64 encoded result of signing the entire normalized SignedInfo element.
KeyInfo	http://www.w3.org/TR/xmldsig-core/#sec-X509Data	We use X.509 certificates. In the signature X509SubjectName and X509Certificate are expected.

Parameter	Applicable Specification	Description
		<X509Data> <X509SubjectName>CN=My Name,O=Test Certificates Inc.,C=US</X509SubjectName> <X509Certificate>MIIB9zCCAWCgAwIBAgIERZwdkzANBgkqhki...</X509Certificate> </X509Data>

In order for a NETS provider to be able to provide its services, it must connect its IT infrastructure to BAZG systems. The message exchange is based on the specifications defined by BAZG / LSVA III.

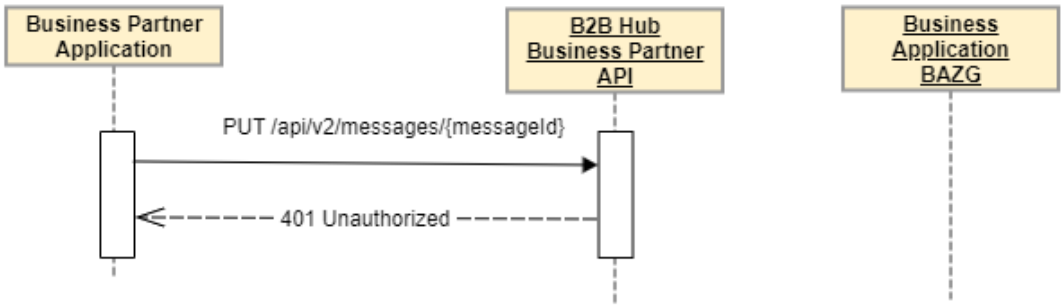
2.8. Transport level / Functional level

Due to the asynchronous structure of the BAZG B2B-Hub-Access Point, a distinction must be made between the (HTTP) transport level and the so-called "functional" level:

- If the business partner transmits a message, he receives a response immediately after transmission (transport layer).
- The message is then processed by the BAZG system and generates a response (functional level).

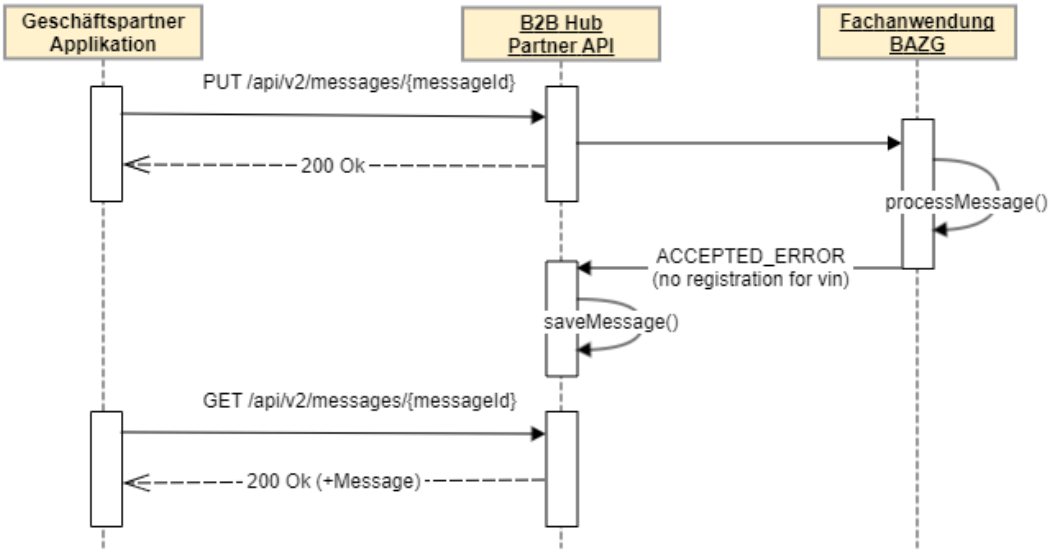
2.8.1. Example 1: Error at transport level

A LSVA III provider transmits a message and forgets to add the bearer token to the request. When the message is transmitted, an HTTP 401 (unauthorized) error is received at the transport level.



2.8.2. Example 2: Error on functional level

Business partner transmits a message (ex. TollDeclaration) which is accepted by the B2B hub. At the transport level, the HVC provider receives an ok (HTTP 200). The message is then processed in the specialized application at the BAZG, where in this specific example it is determined that there is no registration for the vehicle yet. This error on the functional level is reported back to the business partner with a specific message.



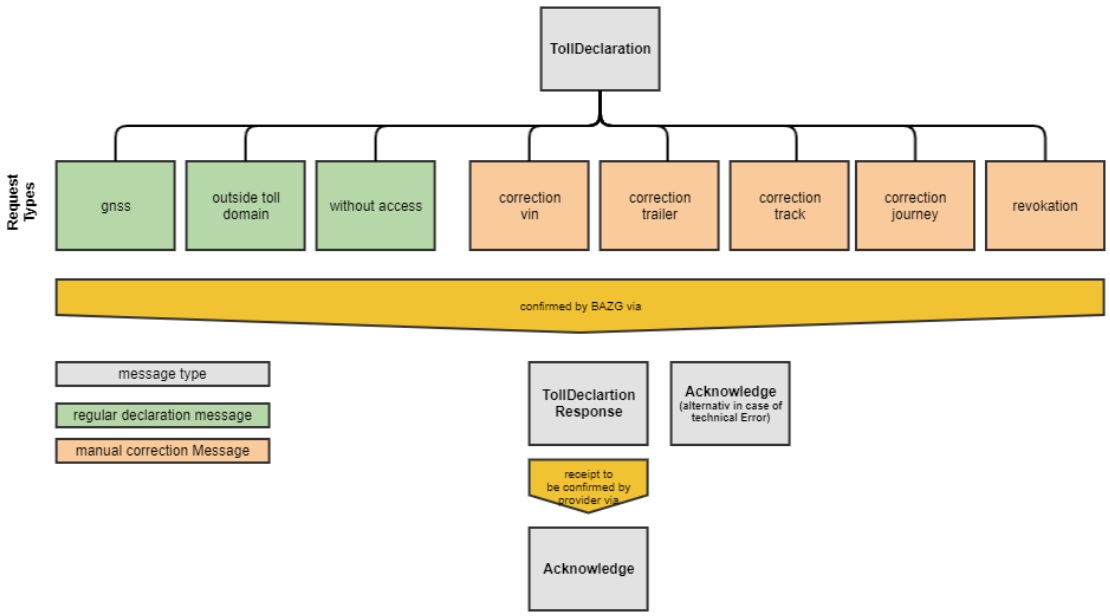
3. NETS Information Model

The information model describes the message structure and message exchange in the context of NETS. Two types of messages and exchange patterns are provided to interact with BAZG / CAMIUNS.

- **TollDeclararation** together with TollDeclarationResponse and Acknoledge for the purpose of daily registration of mileage and the associated levy-determining parameters.
- **Notice** for a more general message exchange used for ex. during registration and deregistration process.
- **Acknowledge** for the confirmation of a message

3.1. TollDeclaration (Message + Interaction Pattern)

3.1.1. TollDeclaration Request



3.1.1.1. TollDeclaration Request Types (Regular Declaration)

The regular declaration message is vehicle-related, and relates to a specific period (=1 day). It is used to deliver driving data in the form of gnss information or when no gnss data is available or the vehicle is outside toll domain a corresponding status information for the vehicle. Therefore, following request message types exist.

Request Type	Description
gnss	Contains the driving data in the form of gnss information including and where available the corresponding trailer information.
outside toll domain	The vehicle was completely outside the toll domain during the relevant period.
without access	The vehicle's driving data could not be retrived for the relevant period.

3.1.1.2. TollDeclaration Request Types (Manual Correction)

A manual correction references to a previously sent regular declaration or parts of it or in some cases to activated manual corrections. For detailed information of the correlation between regular declaration message and manual correction message see: "Correlation and processing activated messages".

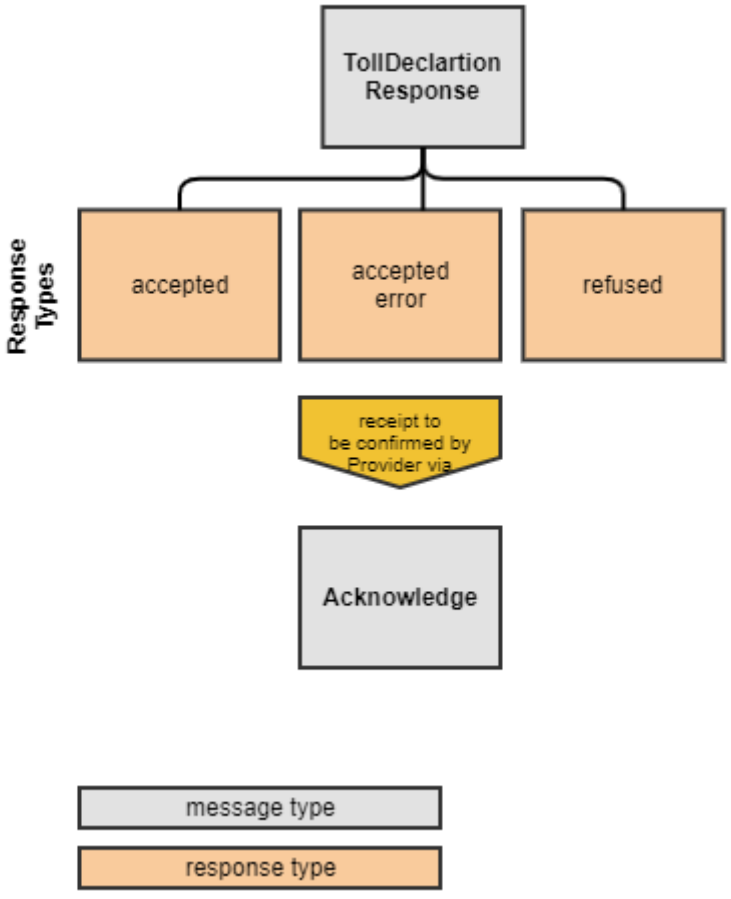
The following types of manual corrections are possible:

Request Type	Description
correction vin	Corrects the vin of regular declaration(s) and if existing manual correction(s).
correction trailer	Corrects the declared trailer information in the provided timeframe not altering declared distance.

Request Type	Description
correction track	Corrects a regular declaration by adding a missing track providing distance and trailer information.
correction journey	Via a regular declaration driving data was registered where there was no journey subject to tax (eg. in cases when vehicle gets transported by a truck, on-board unit has been moved outside the vehicle, ...). Additionaly proactive declaration of days without journey subject to tax can be delcared using correction journey.
revokation	Revokes a submitted manual correction.

3.1.2. TollDeclaration Response

A successfully received tollDeclaration is confirmed by a tollDeclarationResponse. The same response is used for all message types.

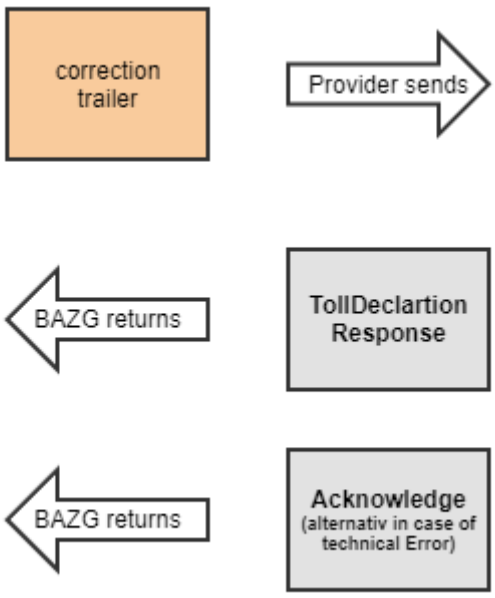


3.1.2.1. TollDeclarationResponse Response Types

Response Types	Description
accepted	Declaration has been accepted by BAZG.
accepted error	Declaration has been accepted by BAZG with a signaled error. Action must be taken by provider.
refused	Declaration has been refused by BAZG. The message will not be processed for given reason(s). Resending after correction is possible.

3.1.2.2. Technical errors

Due to the asynchronous nature of the B2B Hub message exchange, every message (whether from the NETS provider or the BAZG) must be confirmed. For messages of type Notice, an acknowledgement is confirmed in the form of an Acknowledge message. The message exchange for TollDeclarations is much more complex, so a separate message type is available named TollDeclarationResponse. In order to avoid having to implement generic errors such as xml schema violations or problems with the signature twice, no TollDeclarationResponse is reported back in the event of these technical errors, but an Acknowlegde with ackCode NOT_OK is sent back.



This means that once a TollDeclaration has been transmitted, the provider's backend system must be able to handle both messages of type Acknowlegde and messages of type TollDeclarationResponse.

3.1.3. Acknowledge

3.1.3.1. BAZG → Provider

In cases where a sent message cannot be received and processed by CAMIUNS a response on message level shows the reason for rejection. See: 5.8 Acknowledge.

3.1.3.2. Provider → BAZG

Each tollDeclaration message received by the provider needs to be confirmed by a corresponding acknowledge. See: 5.8 Acknowledge.

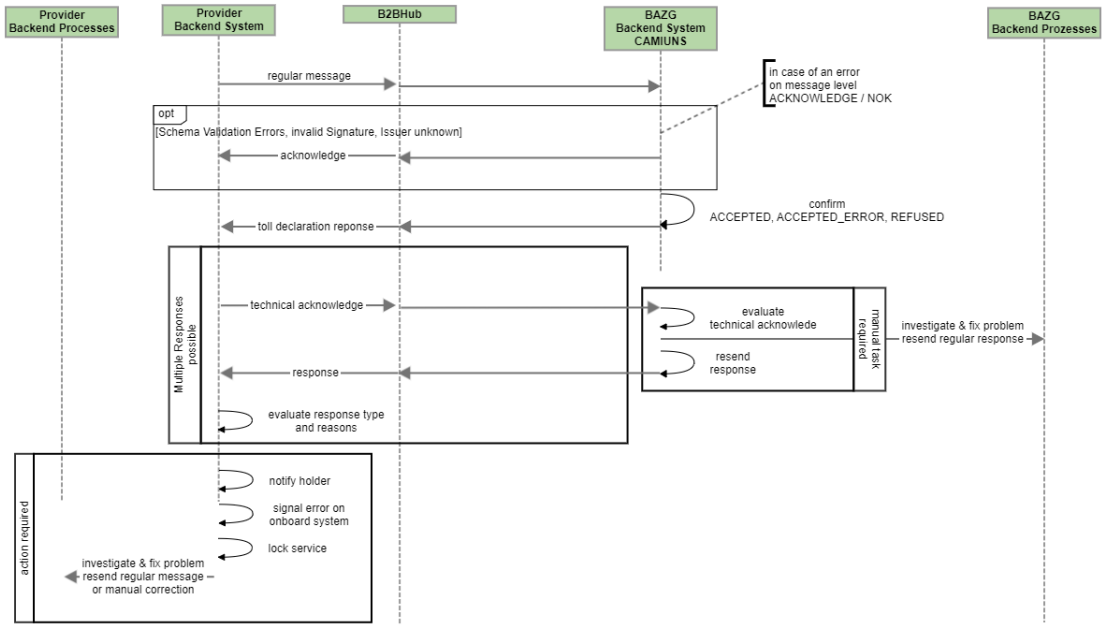
An acknowledge on business level is given usually within a few minutes to an hour. In cases of system downtime it might be delayed up to 24h. If an acknowledge is not proivided within 24h, the message must be resent with its original content after 24h / 48h. If an acknolege is still pending after 48 hours BAZG has to be contacted.

3.1.4. Interaction Pattern

The message exchange for a regular declaration message and a manual correction follow a similar interaction pattern.

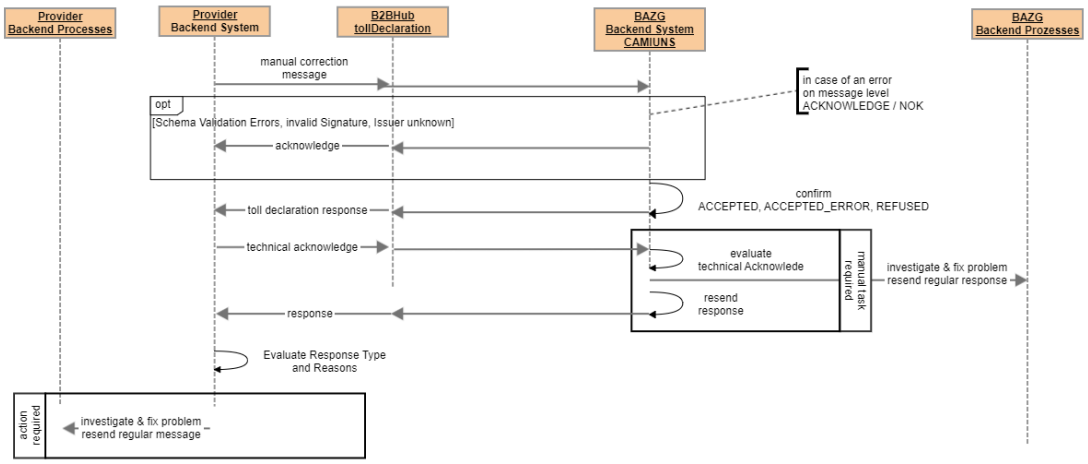
3.1.4.1. Regular Declaration Message

In case of a declaration via regular declaration message the following interaction pattern has to be followed.



3.1.4.2. Manual Correction Message

In case of a declaration via manual correction message the following interaction pattern has to be followed.



3.1.4.3. Message Handling Rules

Beside the interaction pattern described above following rules apply:

3.1.4.3.1. for regular declaration messages:

- Regular declaration messages are identified by vin and declaration period.
- For each vehicle and declaration period at least one regular declaration has to be sent.
- The type can be either:
  - gnss (shall also be used in cases where there was no journey but provider could access the onboard system).
  - outside toll domain
  - without access.
- Declarations have to be done on a daily basis and
- they may be continuous.
- Submitted tracks shall not overlap.
- Submissions of one day by means of multiple registrations will be merged in CAMIUNS.
- Submission is due on the next day following the declaration period but can sent up to 10 days after the declaration period. Any declaration sent after that date will be refused. The date and time of receipt of the notification by the BAZG is decisive.
- Each regular declaration accepted by BAZG will be legally binding after the 10 days.

3.1.4.3.2. for manual correction messages:

- Manual correction messages are identified by vin and declaration period
- Manual correction targets a whole declarationPeriod or part of it
- Manual correction can be sent as soon as data is available
- Manual correction are accepted up to 10 days after the relevant declaration period. The date and time of receipt of the notification by the BAZG is decisive.

- Manual correction shall be delivered at the latest the following day after they have been filed in the system of the provider.
- Each manual correction message accepted by BAZG will be legally binding 10 days after the concernded declaration period.

3.1.5. Correlation and Message Processing

After due dates for both regular declaration messages and manual correction messages have been passed, BAZG starts with correlating and processing of the accepted messages.

3.1.5.1. Correlation (to one or part of) a regular message

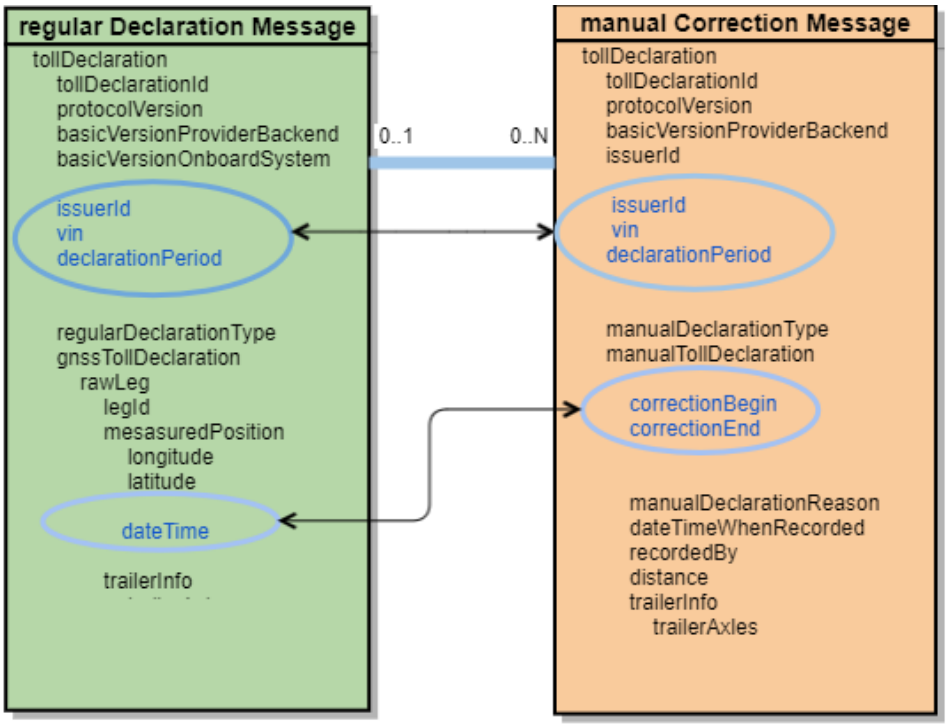
A manual correction message correlates to one (1) or part of a regular declaration message. Correlation is done via attributes:

- issuerId
- vin
- declarationPeriod

If referred to part of the regular declaration message, correlation is further limited by:

- correctionBegin
- correctionEnd

Note: A regular declaration message does not have to be existing (yet) when a corresponding manual correction arrives.



3.1.5.2. Correlation to regular messages and manuel corrections

- A correction of type correction vin is correlated to all affected reglar declaratons and manual corrections.

3.1.5.3. Correlation to exactly one regular message

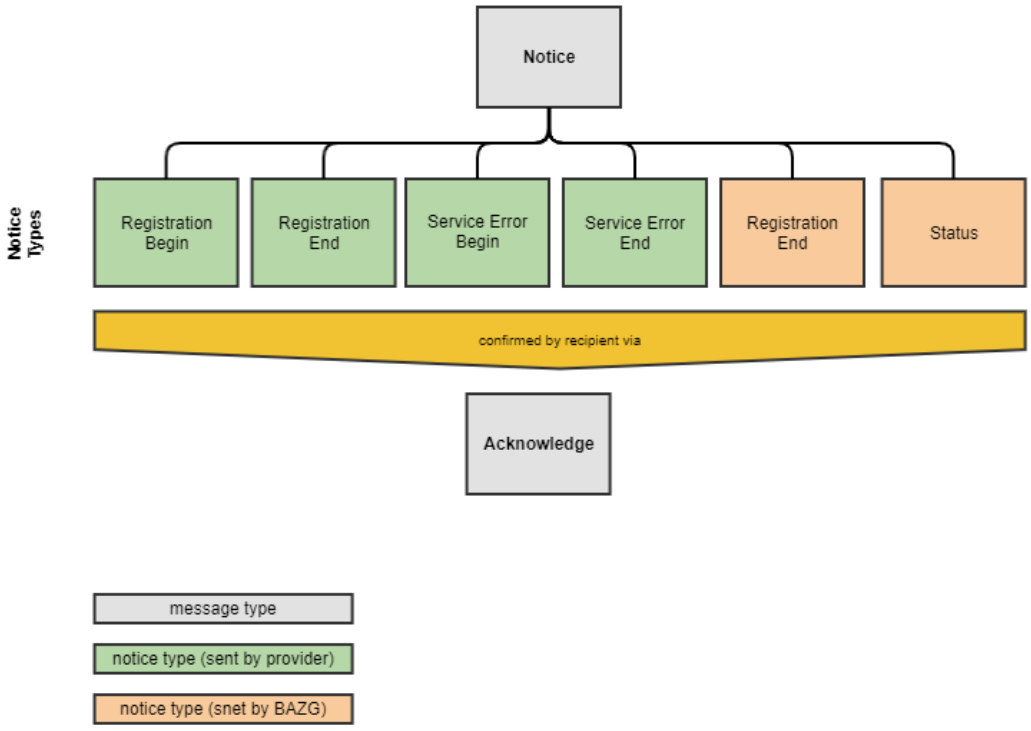
- In some cases, the specification requires correction to a single regular declaration.
- In this case a correlationId is used instead of correctionBegin / correctionEnd to identify the regular declaration to which the correction applies to.
- Manual correction message of type / reason SERVICE_REGISTRATION no correlation may exist.

3.1.5.4. Message Processing

- All received messages which have been sent to CAMIUNS until the 10th day following the relevant period are included in the processing.
- In cases, where multiple manual correction messages for a period have been received and accepted by CAMIUNS, all of them are taken into account.

3.2. Notice (Message + Interaction Pattern)

3.2.1. Notice Request



Notice Request Types

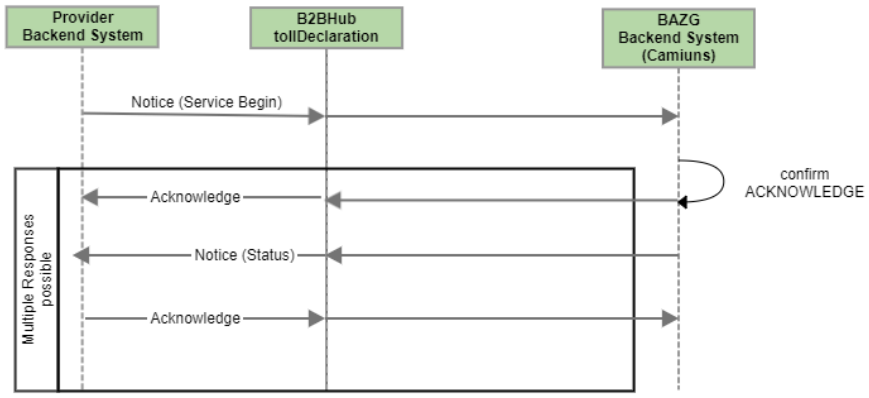
Request Type	Description
Registration Begin	Notice regarding an intended service begin for a vehicle.
Registration End	Notice of a registration end for the given vehicle.
Status	Response to an intended service begin. Provided for foreign vehicles only.

3.2.2. Acknowledge

The recipient of a notice needs to confirm each reception of a response with an ackowledge. The receipt of the response is hereby acknowledged.

Request Type	Description
Acknowledge	Confirmation or rejection of a received notice.

3.2.3. Interaction Pattern



A first response on business level is given usually within a few minutes to a hour. In cases of system downtime it might be delayed up to 24h. If an acknowledge is not proivided within 24h, the message with must be resend with its original content after 24h / 48h. If failure still exists after 48h BAZG has to be contacted.

3.2.4. Correlation

Correlation of status messages can be done via vin.

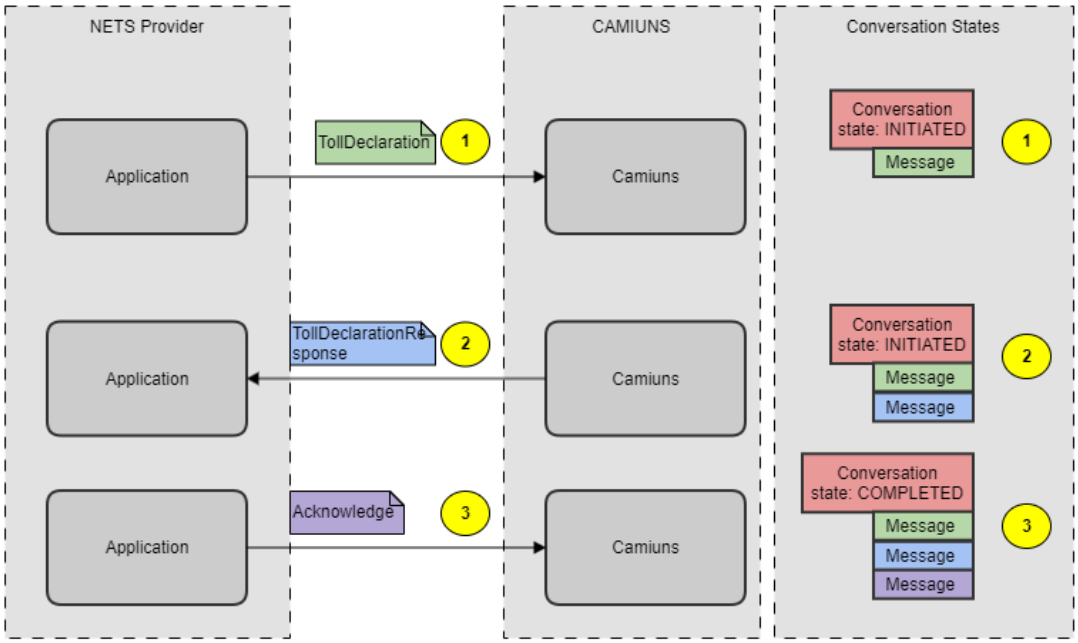
4. NETS Conversation Handling

Every message exchange triggers a new conversation. A conversation is completed when all messages have been adequately confirmed or answered.

If a conversation is open for too long, this is detected and reported via a corresponding channel. This obligation exists both on the provider's side and on the BAZG side.

4.1. TollDeclarations

4.1.1. Provider → BAZG

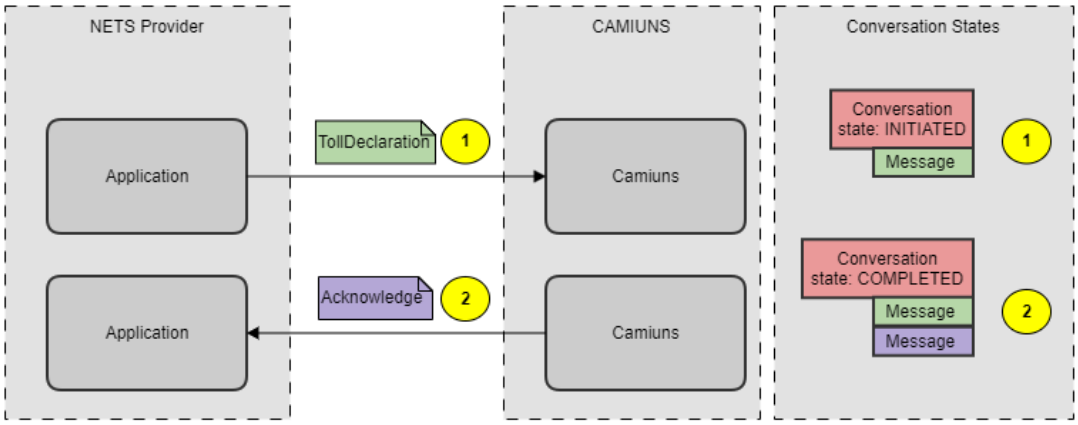


1	The NETS provider sends a TollDeclaration to BAZG. A new conversation is opened with the status INITIATED.
2	The TollDeclaration is checked and processed and is reported back to the NETS provider as accepted with a TollDeclarationResponse. The status of the conversation is not changed.
3	To prove that the provider has received the message, it sends an acknowledge message to the BAZG. The conversation is now COMPLETED.

Remarks:

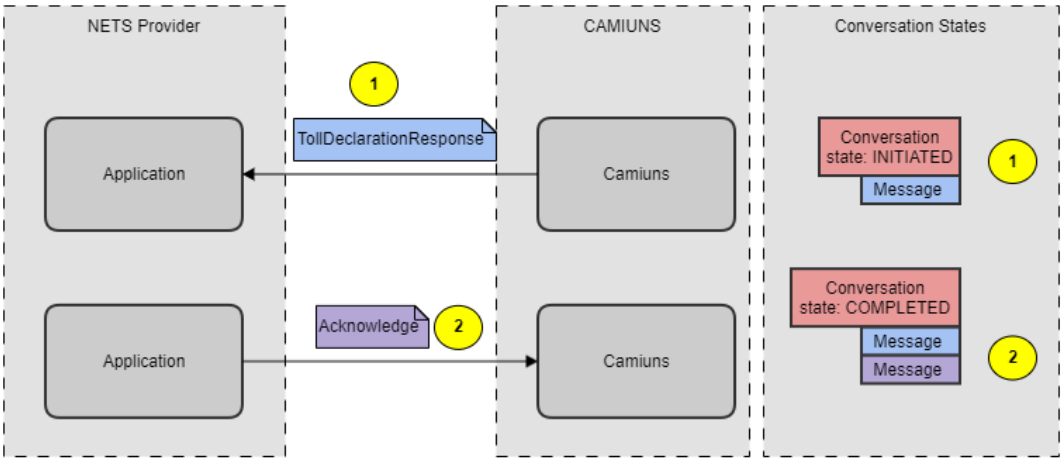
- if a manual toll declaration message is send (see for example TCC01_01 CORRECTION_VIN_SUBMITTED_INCORRECTLY) based on a previous conversation, a new conversation is initiated. Every transmission of TollDeclaration messages leads to a new conversation.
- it doesn't matter if the acknowledge is negative, for the view of a conversation, the conversation is closed.

In case of some specific errors (schema validation, signature errors, the following scenario is also possible:



1	The NETS provider sends a TollDeclaration to BAZG. A new conversation is opened with the status INITIATED.
2	The structure of the xml message was invalid, so a negative acknowlegde is send back. The conversation is now closed.

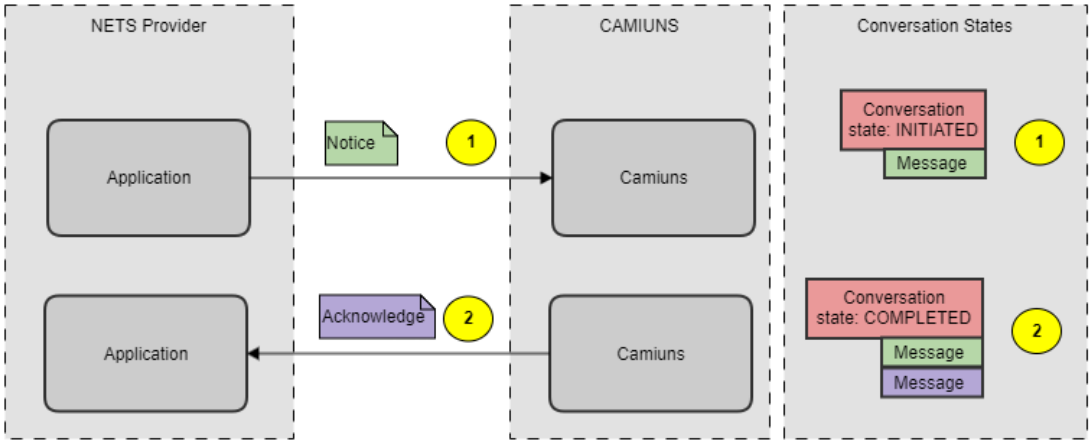
4.1.2. BAZG → Provider



1	A vehicle registration check is carried out approx. 3 days after receipt of a TollDeclaration. If the vehicle cannot be cancelled, for example, this is reported back to the NETS provider. A new conversation with INITIATED status is started.
2	To prove that the provider has received the message, it sends an acknowledge message to the BAZG. The conversation is now COMPLETED.

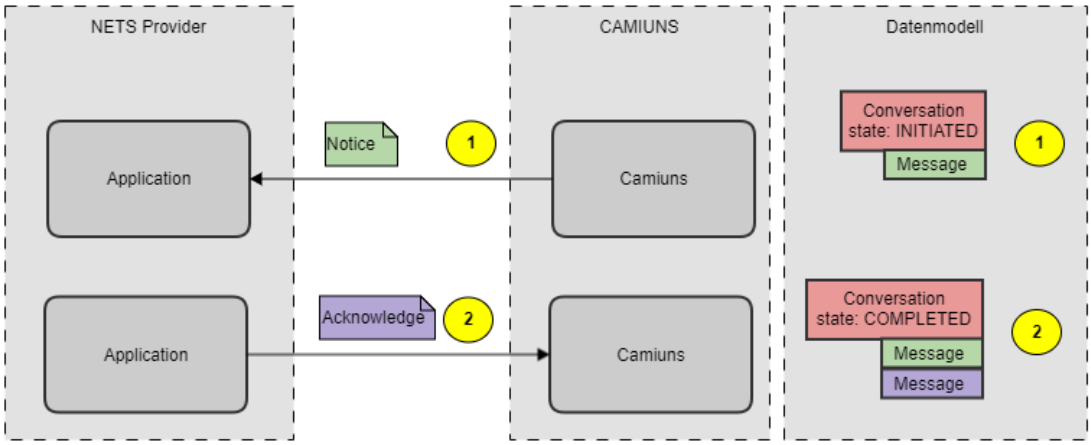
4.2. Notices

4.2.1. Provider → BAZG



1	The NETS provider has signed a contract with a new service user. He registers the service with a Notice to the BAZG. A new conversation is created in state INITIATED.
2	To prove that the BAZG has received the message, it sends an acknowledge message to the NETS provider. The conversation is now COMPLETED.

4.2.2. BAZG → Provider



1	(A provider wants to contract a NETS-A vehicle. To this end, he has sent a notice with REGISTRATION_BEGIN) The BAZG has now checked whether the vehicle is registered in the Vehicle-Scs. The response is sent to the provider. For this reason, a new conversation is opened with the status INITIATED.

2	To prove that the provider has received the message, it sends an acknowledge message to the BAZG. The conversation is now COMPLETED.

5. NETS Messages

5.1. Message

Attribut	Multiplicity	L0	L1	L2	Description
message	1	X			Root-Element for a message
messageContent	1		X		Root-Element for the message content
contentHeader	1			X	Root-Element for the message content header
contentBody	1			X	Root-Element for the content body
ds:Signature	1		X		Root-Element for the envenoped signature

5.2. Message Content Header

Attribut	Multi-plicity	L0	L1	L2	Description
messageId	1	X			Unique identification number given by the issuer
messageDateTime	1	X			Date of the message creation given by the issuer
informationSenderId	1	X			Root-Element for sender information
issuerId	1		X		Businesspartner ID of the sender
informationRecipientId	1	X			Root-Element for recipient information
issuerId	1		X		Businesspartner ID of the recipient

5.3. Message Content Body

Attribut	Multi-plicity	L0	L1	L2	Description
notice tollDeclaration acknowledge	1	X			Identifier for the specific message type: Notice: notice Regular TollDeclaration: tollDeclaration Manual TollDeclaration: tollDeclaration Acknowledge: acknowledge

5.4. Notice

The table below shows a logical model.

Attribut	Multiplicity	L0	L1	L2	Description
notice	1	X			
GENERAL PART					
noticeId	1		X		Unique id issued by the issuer clearly identifying a specific tollDeclarationNotice. 0 to 2^63-1 according to CEN/TS 16986:2016
noticeType	1		X		Type of notice: • REGISTRATION_BEGIN • REGISTRATION_END • REGISTRATION_STATUS
dateTimeWhenRecorded	1		X		Contains the date and time when the notice has been recorded. Timeformat to be used is UTC (e.g. 2024-10-26T19:32:52Z).
recordedBy	1		X		The party who filed the declaration: • HOLDER • PROVIDER • BAZG
REGISTRATION_BEGIN (PROVIDER → BAZG)					
registrationBegin	1		X		
vin	1			X	Vehicle identification number, vin according to the vehicle license.
tollDomain	1			X	If Vehicle is registered in CH or FL use CH else A. • CH • A
effectiveDate	1			X	Date when the service shall be / was activated.
REGISTRATION_END (BAZG → PROVIDER / PROVIDER → BAZG)					
registrationEnd	1		X		
vin	1			X	Vehicle identification number, vin according to the vehicle license.
effectiveDate	1			X	Last day of registration. Timeformat to be used is local date: 2024-01-29
reason	0..1			X	Possible Reasons for termination: • PERMISSION_NO_LONGER_GRANTED

Attribut	Multiplicity	L0	L1	L2	Description
					• DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS • REGISTRATION_CANCELLED • SERVICE_NOT_TIMELY_ACTIVATED (in Diskussion)
REGISTRATION_STATUS (BAZG → PROVIDER)					
registrationStatus			X		
vin	1			X	Id correlating to the corresponding request.
isRegistered	1			X	true, false
effectiveDate	0..1			X	Date of latest registration / deregistration of vehicle.
comment	0..1			X	A comment regarding the Status

5.5. TollDeclaration (Regular Message)

The table below shows a logical model.

Attribut	Multiplicity	L0	L1	L2	L3	L4	L5	L6	L7	Semantics
tollDeclaration	1	X								Common root for the regular and manual interfaces.
tollDeclarationId	1		X							Unique id issued by the issuer clearly identifying a specific tollDeclaration. 0 to 2^63-1 according to CEN/TS 16986:2016
protocolVersion	1		X							Current constant value: 0 Used to maintain different versions of this interface.
basicVersionProviderBackend	1		X							The basicVersionProviderBackend contains the software version of the backend used at the providers side to generate this request.
basicVersionOnboardSystem	0..1		X							The basicVersionOnboardSystem contains the software and hardware version of the onboardsystem used or to generate this request.
vin	1		X							Vehicle identification number, vin according to the vehicle license

Attribut	Multiplicity	L0	L1	L2	L3	L4	L5	L6	L7	Semantics
										or primary source in the vehicle.
declarationPeriod	1		X							Period, a single day to which the submitted data corresponds to. Contains the whole period of the corresponding day (local date) in Switzerland from 00:00:00 to 23:59:59. All the mesured waypoints must fit within this reported period.
regularTollDeclaration	1		X							
regularDeclarationType	1			X						GNSS = GNSS-data as recorded by the onbaord system for the given period. WITHOUT_ACCESS = onboard system could not be accessed for the given period. VEHICLE_OUTSIDE_TOLL_DOMAIN = vehicle outside toll domain for the entire given period.
for tollDeclarations with regularDeclarationType = GNSS										
gnssTollDeclaration	0..1			X						Contains the entire position data as automatically recorded for the given period.
rawLegs	1				X					
rawLeg	1..N					X				Contains the entire position data for a leg. A leg represents a route passed with the same configuration of toll relevant parameters.
legId	1						X			Incremented integer value starting with 1.
measuredPositions	1						X			
measuredPosition	1..N							X		Position data shall be based on the WGS84 coordinate system. For details see: NIMA TR8350.2, Third Edition – Amendment 1, January 2000, Department of Defense – World Geodetic System 1984, Its Definition and Relationships with Local Geodetic Systems, issued by the National Imaging and

Attribut	Multiplicity	L0	L1	L2	L3	L4	L5	L6	L7	Semantics
										Mapping Agency (NIMA), US Department of Defense. Remark: There is no need to order positions in any given way.
longitude	1								X	as defined in ISO 6709, in microdegrees, >0=east, <0=west, absolute value <=180°
latitude	1								X	as defined in ISO 6709, in microdegrees, >0=north, <0=south, absolute value <=90°
timestamp	1								X	date and time, shall be within reported period. Timeformat to be used is UTC.
trailer	0-1						X			Missing element = no trailer
trailerType	1							X		T=trailer, S=semitrailer
trailerWeight	1							X		0 = exempted from HVC Weight in kg with a maximaum resolution of 10 kg. Note: In cases where the onboard system allows the user to enter a lower resolution then 10 kg, the weight shall be rounded downwards.
for tollDeclarations with regularDeclarationType = WITHOUT_ACCESS, no further attributes needed										
for tollDeclarations with regularDeclarationType = VEHICLE_OUTSIDE_TOLL_DOMAIN, no further attributes needed										

5.6. TollDeclaration (Manual Correction Message)

The table below shows a logical model.

Attribut	Multiplicity	L0	L1	L2	L3	L4	Description
tollDeclaration	1	X					
manual interface (partly automatically processed information)							
tollDeclarationId	1		X				Unique id issued by the issuer clearly identifying a specific tollDeclaration. 0 to 2^63-1 according to CEN/TS 16986:2016

Attribut	Multiplicity	L0	L1	L2	L3	L4	Description
protocolVersion	1		X				Current constant value: 0 Used to maintain different versions of this interface.
basicVersionProviderBackend	1		X				The basicVersionProviderBackend contains the software version of the backend used at the providers side to generate this request.
vin	1		X				Vehicle identfication number, vin according to the vehicle license or primary source in the vehicle.
declarationPeriod	1		X				Period, single day to which the submitted data corresponds to.
manualTollDeclaration	1		X				Contains a correction for a related regular declaration or part of it.
manualDeclarationType	1			X			Type of the manual declaration: CORRECTION_TRACK CORRECTION_TRAILER CORRECTION_VIN CORRECTION_JOURNEY REVOCATION Details see chapter: Manual Declaraton Types & Reasons.
dateTimeWhenRecorded	1			X			Contains the date and time when the manual declaration has been filed by the recording party. Timestamp relevant for legal deadlines. Timeformat to be used is UTC (e.g. 2024-10-26T19:32:52Z).
recordedBy	1			X			The party who filed the declaration: • HOLDER • PROVIDER
manualDeclarationReason	1			X			Reason for the manual declaration as stated in chapter: Manual Declaration Reasons.
CORRECTION_TRACK							
correctionTrack	1			X			
correctionPeriod	1				X		
correctionBegin	1					X	dateTime UTC (e.g. 2024-10-26T19:32:52Z) / For further information about the usage of

Attribut	Multiplicity	L0	L1	L2	L3	L4	Description
							correctionBegin see remark at the end of this table.
correctionEnd	1					X	dateTime UTC (e.g. 2024-10-26T20:32:52Z) / For further information about the usage of correctionEnd see remark at the end of this table.
distance	1				X		distance in km.
trailerInfo	0..1				X		Missing element = no trailer
tailerType	1					X	T=trailer, S=semitrailer
trailerWeight	1					X	0 = exempted from HVC Weight in kg with a maximum resolution of 10 kg.
CORRECTION_TRAILER							
correctionTrailer	1			X			
correctionPeriod					X		
correctionBegin	1					X	dateTime UTC (e.g. 2024-10-26T19:32:52Z) / For further information about the usage of correctionBegin see remark at the end of this table.
correctionEnd	1					X	dateTime UTC (e.g. 2024-10-26T20:32:52Z) / For further information about the usage of correctionEnd see remark at the end of this table.
trailerInfo	0..1				X		Missing element = no trailer
trailerType	1					X	T=trailer, S=semitrailer
trailerWeight	1					X	0 = exempted from HVC Weight in kg with a maximum resolution of 10 kg.
CORRECTION_VIN							
correctionVin				X			
replacementVin	1				X		The vin used for correction.
CORRECTION_JOURNEY							
correctionJourney				X			
correctionBegin	1				X		UTC (e.g. 2024-10-26T19:32:52Z) / For further information about the usage of correctionBegin see remark at the end of this table.

Attribut	Multiplicity	L0	L1	L2	L3	L4	Description
correctionEnd	1				X		UTC (e.g. 2024-10-26T19:32:52Z)/ For further information about the usage of correctionEnd see remark at the end of this table.
REVOCATION							
revocation				X			
tollDeclarationId	1				X		tollDeclarationId of manual correction which shall be revocated.

Remark: Usage of correctionBegin and correctionEnd.

Time within the provided declarationPeriod in the format hh:mm.ss. The combination of correctionBegin and correctionEnd can span over:

- a part of a day
- a whole day

The used resolution is:

- exact time where the begin and/or end can be exactly be determined.
- a quarter of an hour where time set is an approximation.

When the correction spans:

- the whole period, the correctionBegin has to be set to: 00:00:00 and for correctionEnd to 23:59:59,
- an approximate time span corretionBegin is including the starting second: e.g. 09:00:00 and exludes the end second: e.g. 09:14:59.

Declarations containing a CorrectionBegin and CorrectionEnd may not overlapp. If overlapping, the interface will respond with: validation_error, reason text: Timespan overlapping. The user shall be guided during data entry.

5.6.1. Manual Declaration Reasons

manualDeclaration Reason	Recorded By	Recommendation for the presentation of supporting information and validations within the failover solution	Notes
CORRECTION_TRACK			
SERVICE_ERROR	HOLDER	The system of the provider marks the timeframe marks confirmed errors by the	The correction is due to a journey during a period where the provider signaled a service error. This error needs

manualDeclaration Reason	Recorded By	Recommendation for the presentation of supporting information and validations within the failover solution	Notes
		begin of the error and the end of the error.	to be signaled whenever: • driving data cannot be recorded by the onboard system. • the error exists over a period longer than 5 minutes on a single day.
SERVICE_REGISTRATION	HOLDER	The system of the provider limits the useage of the reason code: The reason service registration shall only be applied from the time of the initial service registration for a vehicle until the service has been taken into operation by the holder.	The correction is due to receive registration for the service while commissioning has not yet been completed. This reason has not to be used in cases were provisioning is done for replacment reasons. Use service error instead.
JOURNEY_WITHOUT_DEVICE	HOLDER	The system of the provider limits the correction to timeframes where there has been no	The correction is due to a journey without device.

manualDeclaration Reason	Recorded By	Recommendation for the presentation of supporting information and validations within the failover solution	Notes
		journey with the given device.	
CORRECTION_TRAILER			
WRONG_TRAILER	HOLDER	The system of the provider limits the correction to journeys reported by a regular declaration of type gnss.	The correction is due to a journey with wrong trailer settings. Corrections of type WRONG_TRAILER have to be alligned with a reported regular declaration of type gnss.
CORRECTION_VIN			
INCORRECT_VIN	PROVIDER / HOLDER	The reason for the error is due to a discrepancy of the filed registration at the road traffic office and the vin provided. Therefore the error can have multiple sources and reasons.	The correction may be necessary in cases where a declaration has been accepted with error, reason code no_registration_for_vin and a reason text "vin not found". Once the correction has been successfully applied, the corresponding declarations will be confirmed with accepted / ok. The corrections shall be applied to all corresponding incorrect declarations already sent.

manualDeclaration Reason	Recorded By	Recommendation for the presentation of supporting information and validations within the failover solution	Notes
			Provider has to send a declaration for each affected period.
CORRECTION_JOURNEY			
NO_JOURNEY	HOLDER	The correction begin and correction end shall always span the whole day.	The correction has been proactively filed by the holder for a known period with no journey. This for periods where there are no journeys and the holder can not ensure access to the device (e.g. holidays).
MOVED_WITHOUT_JOURNEY_VEHICLE_LOADED	HOLDER	The system of the provider limits the correction to journeys reported by a regular declaration of type gnss.	Example: Vehicle loaded on truck, on train.
MOVED_WITHOUT_JOURNEY_NON_TAXABLE_JOURNEY	HOLDER	The system of the provider limits the correction to journeys reported by a regular declaration of type gnss.	The correction is due to the fact that the onboard system / unit has been moved without taxable journey.
REVOCATION			
REVOKE_MANUAL_CORRECTION	HOLDER	The system of the provider limits the revocation to manual correction	Revoke a previously sent manual correction. If the specified time limit of 10 days

manualDeclaration Reason	Recorded By	Recommendation for the presentation of supporting information and validations within the failover solution	Notes
		where deadline for processing has not yet been reached.	after the relevant declaration period has been exceeded BAZG will answer with refused and deadline missed.

5.7. TollDeclarationResponse

The table below shows a logical model.

Attribut	Multi-plicity	L0	L1	L2	Description
tollDeclarationResponse	1	X			
tollDeclarationId	1		X		TollDeclarationId to which this response correlates to.
responseSequenceNumber	1		X		Sequence number strarting with 1 of the responses for the correlated tollDeclarationId. (Example: An initial response ACCEPTED can be overruled by a later response ACCEPTED_WITH_ERROR in case where a tollDeclaration from a different provider has been received).
responseType	1		X		ACCEPTED - Declaration has been accepted by BAZG. ACCEPTED_ERROR - Declaration has been accepted by BAZG with a signalled error. Action to be taken by issuer. REFUSED - Declaration has been refused by BAZG. Initial Message needs correction and potential resend of the corrected message.
responseReason	1		X		according to details in chapter "ResponseReasons".
responseReasonType	1			X	ResponseRasonID according to chapter "ResponseReasons".
responseReasonText	0..1			X	Further explanations for the ResponseReasonID.
validationErrorCode	0..1			X	Error Code for distinguish validation errors

5.7.1. ResponseReasons

ResponseType	ResponseReasonType	Description
ACCEPTED	OK	No action required. Note: Can be REFUSED due to new facts within 10 days.
ACCEPTED_ERROR / REFUSED	NO_REGISTRATION_FOR_VIN	No registration could be found for the provided vin. Response overrules a previously sent response with ResponseType ACCEPTED in cases where the confirmation of registration at the traffic office does not arrive within the expected time limit. 10 days after declaration: If the vin is not corrected in a timely manner, declarations filed under the incorrect vin will be REFUSED. For foreign vehicles no vehicle with the provided replacementVin has been registered for the NETS service.
REFUSED	DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS	BAZG has received declarations for the same vin from a different provider. Note: BAZG will accept the declarations for

ResponseType	ResponseReasonType	Description
		the day of provider / service change and refuse declarations for the following day for the provider where the vehicle has been under service previousely.
REFUSED	DEADLINE_MISSED	Deadline for the declaration or correction that has been missed.
REFUSED	DECLARATION_ID_NOT_UNIQUE	The combination of IssuerId and tollDeclarationId is not unique. The service provider is ordered to fix the error and potentially resend the message.
REFUSED	REPLACEMENT_VIN_UNKNOWN	No registration could be found for the provided replacement VIN. For domestic vehicles the replacementVin does not correspond to a registration filed by a traffic office or registration has been cancelled. For foreign vehicles no vehicle with the provided replacementVin has been registered for

ResponseType	ResponseReasonType	Description
		the NETS service.
REFUSED	VALIDATION_ERROR	Any genereral validation error. The error needs to be fixed by the provider and the necessary action has to be taken to resend the message. Note: Validation due to there nature cannot be handled on business level are supplied via the acknowledge message.

5.7.2. ResponseReasonsText

The reason text in the message is explanatory in nature. In cases where the exact reason cannot be identified, the BAZG should be contacted.

5.7.3. Validation Error Codes

Validation rule category	Validation Error Code	Response Reason Text	Category description
BUILDING_BLOCK_CONSISTENCY	1000	"Correction begin is not before correction end"	These validation rules apply to all messages types. Every transmitted message must be consistent.
BUILDING_BLOCK_CONSISTENCY	1001	"Correction begin date or correction end date do not match declaration date"	
BUILDING_BLOCK_CONSISTENCY	1002	"GNSS toll declarations must contain at least one waypoint"	

Validation rule category	Validation Error Code	Response Reason Text	Category description
JOURNEY_DECLARATION_CONSISTENCY	2000	Correction period overlaps with correction period of %s other declaration%s of type%s %s	These validation rules also apply to all message types with a reference to a toll declaration. In this category, however, the consistency with the other messages of the same vehicle/day already transmitted is checked. For example, messages of type CORRECTION_TRACK must match the time span of the transmitted GNSS positions.
CORRECTION_JOURNEY	3000	"Begin of period and end of period should cover the whole day"	These validation rules only apply to messages of type CORRECTION_JOURNEY.
CORRECTION_VIN	4000	"Replacement VIN must be different from original VIN"	These validation rules only apply to messages of type CORRECTION_VIN.
CORRECTION_VIN	4001	"There is already a correction VIN registered for this declaration"	
REVOCATION	5000	"Toll declaration with id %s not found or not feasible for revocation"	These validation rules only apply to messages of type REVOCATION.

5.8. Acknowledge

The table below shows a logical model.

Attribut	Multi-plicity	L0	L1	L2	L3	Description
acknowledge	1	X				

Attribut	Multi-plicity	L0	L1	L2	L3	Description
correlationId	1		X			The messageId of the original message this acknowledge correlates to.
ackCode	1		X			The acknowledge code. OK NOT_OK
issues	0..1		X			Root element of possible issues
issue	1..n			X		Root element of a single issue
issueCode	1				X	A possible issue code (has to be defined)

5.8.1. Issue Codes

The table below contains possible issue codes in an acknowledge can occur.

- Issue Codes of type GENERAL can be set on both sides (from the provider or BAZG)

type	IssueCode	Description
GENERAL	1000	There are validation errors there are in contradiction to defined xsd schema.
GENERAL	1001	The signature is not valid.
GENERAL	1002	Issuer ID unknown
NOTICE	2000	The Notice cannot be accepted.
TOLL_DECLARATION	3000	The tollDeclarationId is unknown

5.9. Schemas

The Schemas for the interfaces are provided in the same location as this specification.

5.9.1. General Message Schema Files

The following schema file desribe common and recently used types, the message and signature structure.

Schema File	Description
common.xsd	Common and recently used types like VIN, Issuer ID, ...
message.xsd	The top level structure of a message.
xmldsig-core-schema.xsd	The structure of the xml signature.

5.9.2. Message Content Schema Files

All schemes used in the content body are listed here:

Schema File	Description
tollDeclaration.xsd	The structure of a toll declaration message.
regularTollDeclaration.xsd	The structure of the regular toll declaration subtype.
manualTollDeclaration.xsd	The structure of the manual toll declaration subtype.
tollDeclarationResponse.xsd	The structure of the toll declaration response.
acknowledge.xsd	The structure of an acknolegde message.
notice.xsd	The structure of a notice message.

5.9.3. Versions

Version	Specification Version
1.0.0	1.0
1.0.1	1.1

6. NETS Messages-Exchange Szenarios

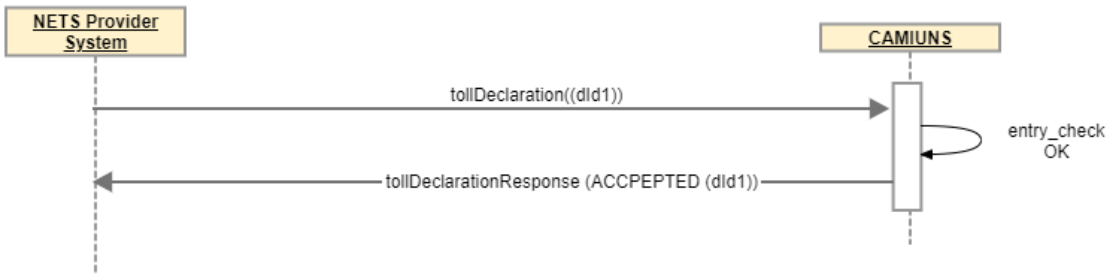
The following illustrations show some message exchange examples.

Note: Conversations are simplified in terms that the exchange of the necessary ackowledges for each message were omitted.

6.1. TollDeclaration (OK)

6.1.1. With result ACCEPTED

- The system of the NETS provider sends a declaration with tollDeclarationId (dId1).
- CAMIUNS entry check evaluates OK and returns ACCEPTED.



6.2. TollDeclaration (CORRECTION_JOURNEY)

There are 3 possible scenarios to correct a regular toll declaration with a manual correction of type correction journey:

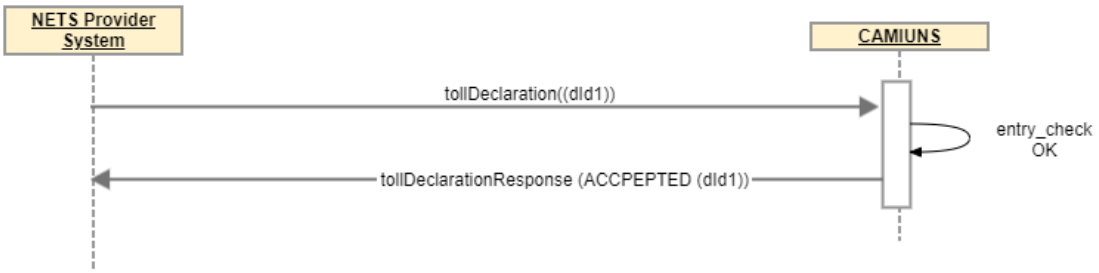
Type	Possible reason	Time period
NO_JOURNEY	The correction has been proactively filed by the holder for a known period with no journey. This for periods where there are no journeys and the holder can not ensure access to the device (e.g. holidays).	Day
MOVED_WITHOUT_JOURNEY_VEHICLE_LOADED	Example: Vehicle loaded on truck, on train.	Timespan
MOVED_WITHOUT_JOURNEY_NON_TAXABLE_JOURNEY	Possible reason: OBU is sent by mail from A to B and records a journey	Timespan

6.2.1. Preconditions

The system of the NETS provider sends a regular declaration as described in TollDeclaration (OK).

6.2.2. Sequence diagram

- The system of the NETS provider sends a manual declaration with tollDeclarationId (dId1).
- CAMIUNS entry check evaluates OK and returns ACCEPTED.



6.3. TollDeclaration (CORRECTION_TRACK)

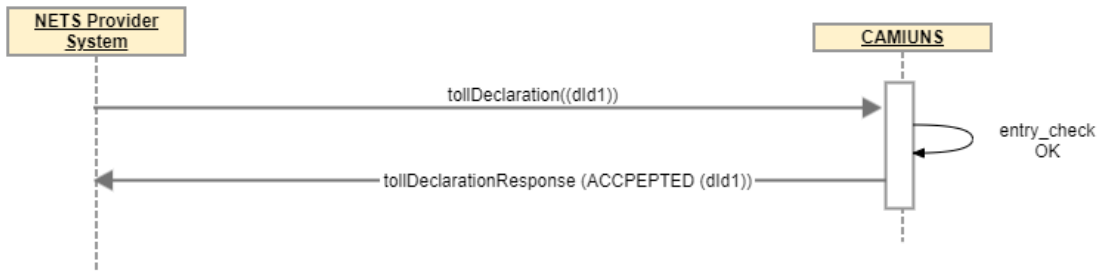
There are 3 possible scenarios to correct a regular toll declaration with a manual correction of type correction journey:

Type	Possible reason	Time Period
SERVICE_ERROR	The correction is due to a journey during a period where the provider signaled a service error. This error needs to be signaled whenever: • driving data cannot be recorded by the onboard system. • the error exists over a period longer than 5 minutes on a single day.	Timespan
SERVICE_REGISTRATION	The correction is due to receive registration for the service while commissioning has not yet been completed. This reason has not to be used in cases were provisioning is done for replacment reasons. Use service error instead.	Timespan
JOURNEY_WITHOUT_DEVICE	The correction is due to a journey without device.	Timespan

6.3.1. Preconditions

The system of the NETS provider sends a regular declaration as described in TollDeclaration (OK).

6.3.2. Sequence diagram



6.4. TollDeclaration (CORRECTION_TRAILER)

The correction is due to a journey with wrong trailer settings. Corrections of type WRONG_TRAILER have to be alligned with a reported regular declaration of type gnss.

6.4.1. Preconditions

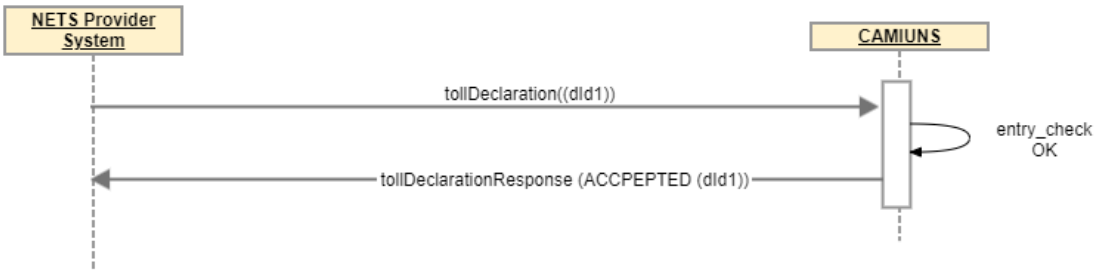
The system of the NETS provider sends a regular declaration of type gnss as described in TollDeclaration (OK).

dateTime of first gnss point	dateTime of last gnss point
2022-02-11T09:48:54.000Z	2022-02-11T12:59:59.999Z

6.4.2. Correction timespan

The timespan a correction can take place is limited by the underlying regular toll declaration (in the example above between 2022-02-11T09:48:54.000Z and 2022-02-11T12:59:59.999Z)

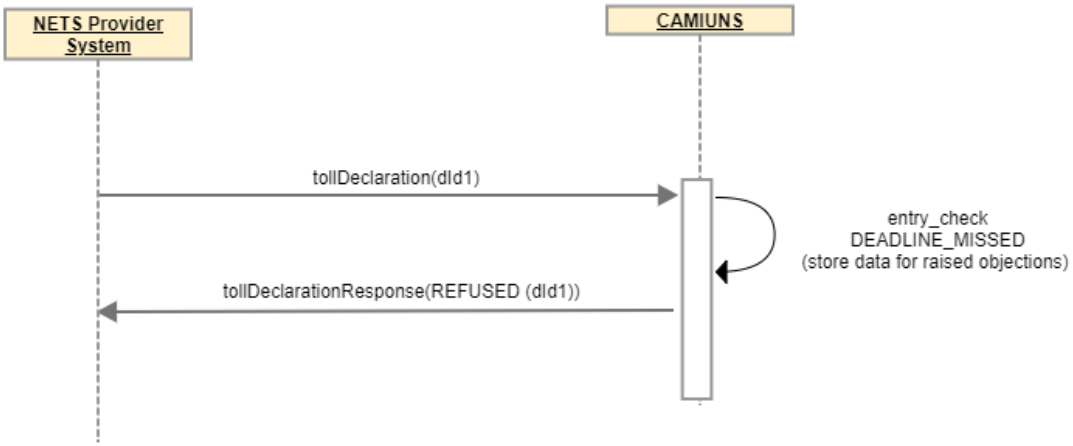
6.4.3. Sequence



6.5. TollDeclaration (DEADLINE_MISSED)

6.5.1. With result REFUSED

- The system of the NETS provider sends a declaration after the deadline given.
- During entry check CAMIUNS recognizes a DEADLINE_MISSED.
- CAMIUNS rejects the message with REFUSED and DEADLINE_MISSED.

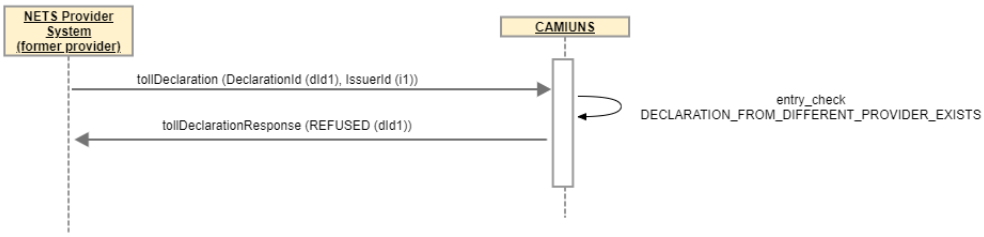


Note: even that regular declarations will be proceeded only for 10 days after declarationPeriod all regular declarations are expected to be sent to BAZG.

6.6. TollDeclaration (DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS)

6.6.1. With result REFUSED

- CAMIUNS has received a declaration for the same vehicle from a different, new provider. Therefore the messages of the existing old / former provider will be refused the day after.



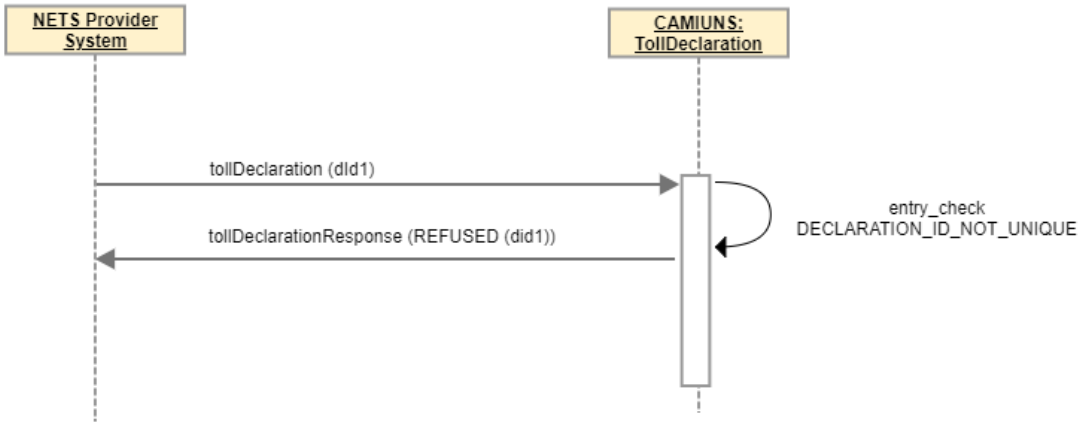
Note:

- Additionally a notice REGISTRATION_END will be sent to the former provider in a different process, signaling that the delivery of further messages for the given vehicle must be stopped and remuneration will be terminated.
- The message of type DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS may be immediate or a previousely accepted message may be refused.

6.7. TollDeclaration (DECLARATION_ID_NOT_UNIQUE)

6.7.1. With result REFUSED

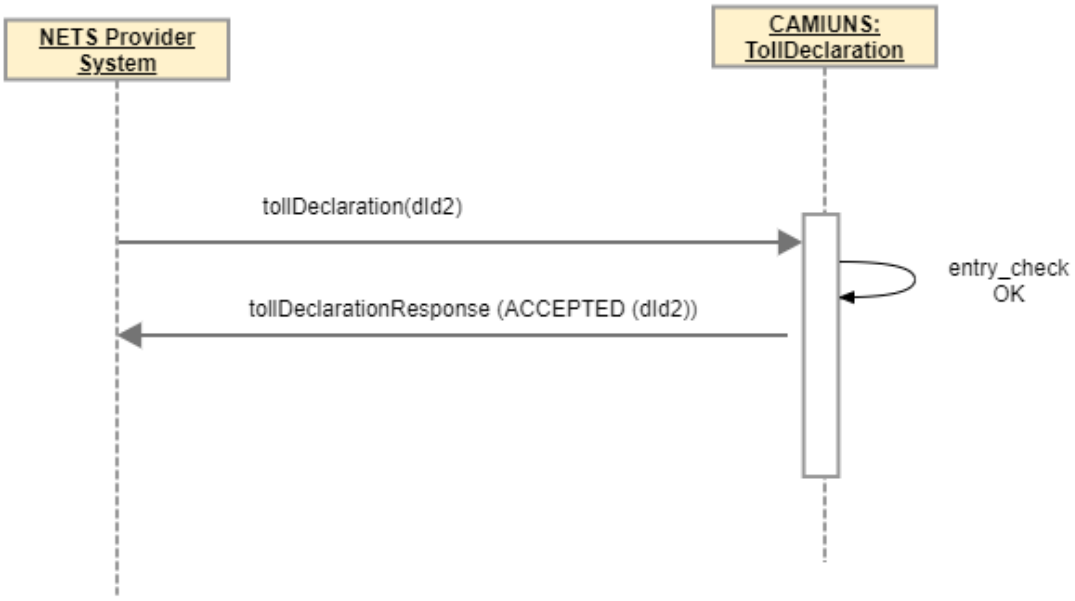
- The system of the NETS provider sends a declaration with declaration id (did1).
- During entry check CAMIUNS recognizes a DECLARATION_ID_NOT_UNIQUE.
- CAMIUNS rejects the message with REFUSED and DECLARATION_ID_NOT_UNIQUE.



Note: Uniqueness is required within the declaration messages sent by one NETS provider.

6.7.1.1. resending with result ACCEPTED

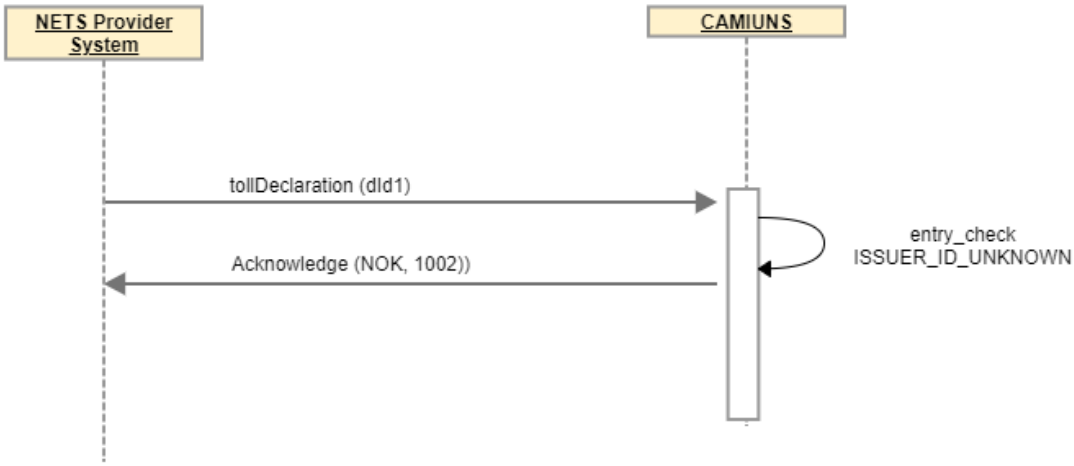
- provider successfully corrects DECLARATION_ID_NOT_UNIQUE



6.8. TollDeclaration, Notice (ISSUER_ID_UNKNOWN)

6.8.1. With result NOK (1002)

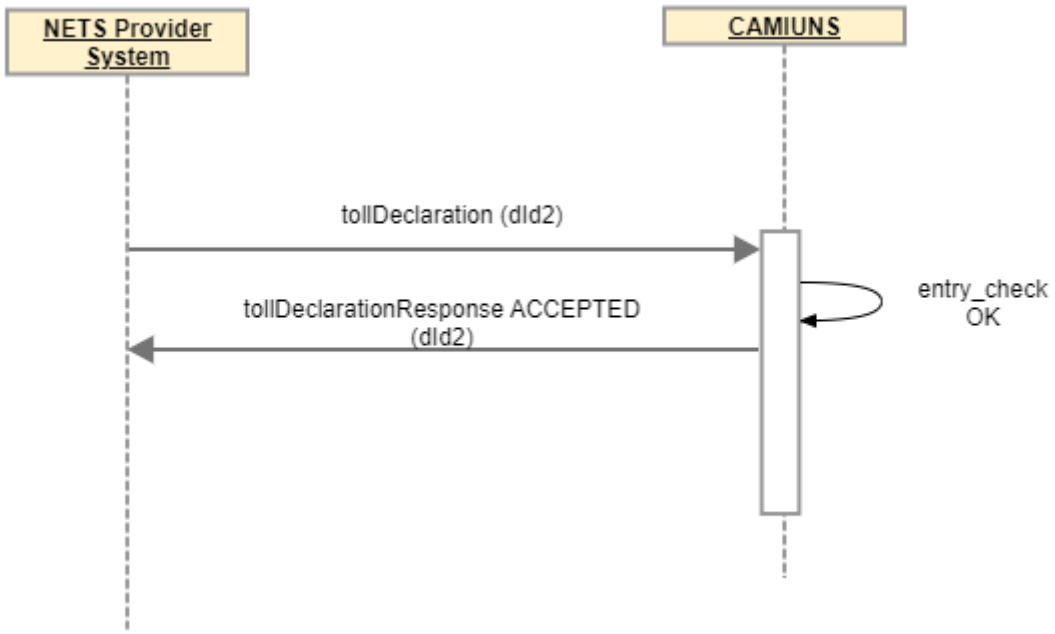
- The system of the NETS provider sends a declaration with declaration id (did1).
- During entry check CAMIUNS recognizes a ISSUER_ID_UNKNOWN.
- CAMIUNS rejects the message with REFUSED and ISSUER_ID_UNKONWN.



The scenario is also valid for messages of the type notice.

6.8.1.1. resending with result ACCEPTED

- The system of the NETS provider successfully correts ISSUER_ID_UNKNOWN.
- Note: the same declarationId (did1) cannot be used to resend the message.



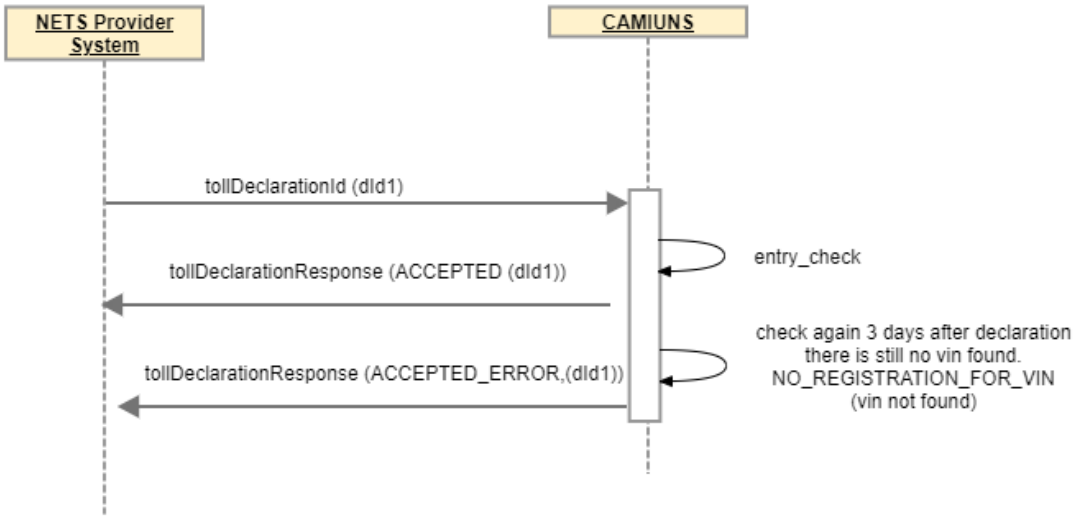
The scenario is also valid for messages of the type notice.

6.9. TollDeclaration (NO_REGISTRATION_FOR_VIN)

6.9.1. TollDeclaration (NO_REGISTRATION_FOR_VIN - vin not found)

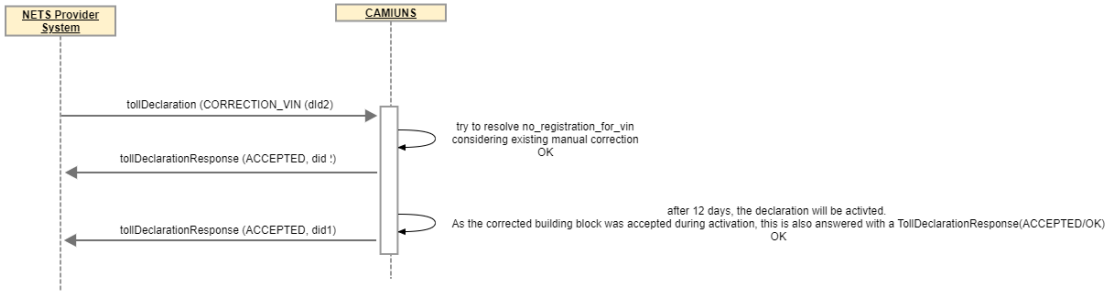
6.9.1.1. With result ACCEPTED ERROR / REFUSED

- During entry check CAMIUNS recognizes an NO_REGISTRATION_FOR_VIN.
- In expectation that this vin will be received successfully during the next coming days CAMIUNS confirms with ACCEPTED referring to (dId1).
- In the following days CAMIUNS tries to resolve the NO_REGISTRATION_FOR_VIN status.
- When not successful during 3-days, BAZG responds with an ACCEPTED_ERROR (NO_REGISTRATION_FOR_VIN) referring to (dId1). The reason text of the message will contain "vin not found"



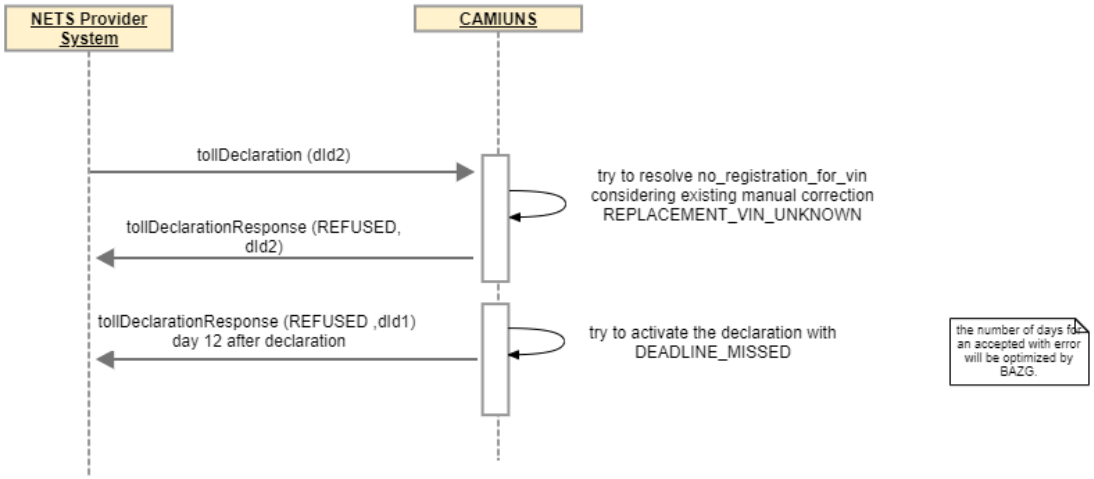
6.9.1.2. ... manual correction with result ACCEPTED

- The system of the NETS provider successfully corrects the NO_REGISTRATION_FOR_VIN.
- Both the manual correction (did2) and the referenced declaration (did1), see above, are confirmed with ACCEPTED.



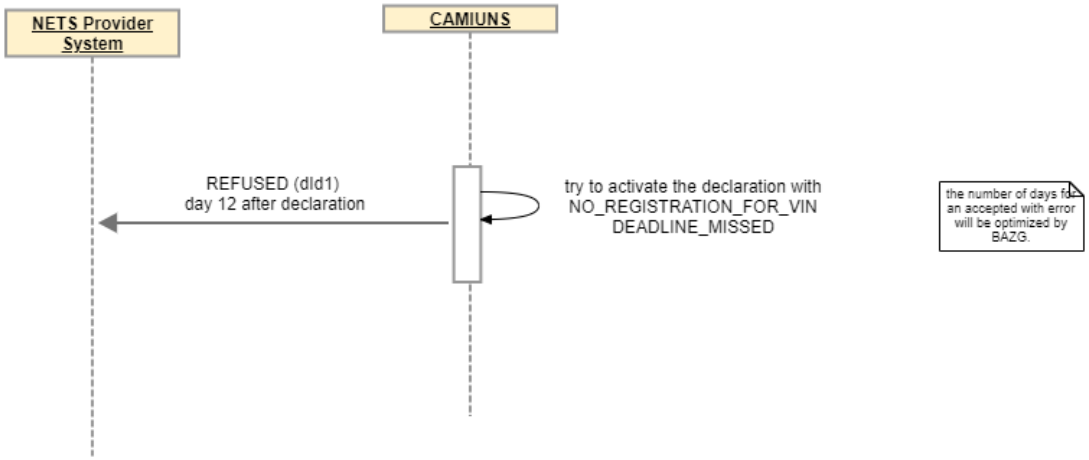
6.9.1.3. ... manual correction with result REFUSED

- The system of the NETS provider unsuccessfully corrects the NO_REGISTRATION_FOR_VIN.
- Both the manual correction (did2) and the referenced declaration (did1), see above, are REFUSED.



6.9.1.4. ... correction not timely received

- the provider / holder or traffic office misses to timely correct the ACCEPTED_ERROR (NO_REGISTRATION_FOR_VIN) status.
- the regular declaration (did) is REFUSED.



6.10. TollDeclaration (CORRECTION_VIN)

This scenario specifies the scenario TollDeclaration (NO_REGISTRATION_FOR_VIN).

This scenario offers several approaches on how to proceed in the event of a VIN correction.

6.10.1. Background

Prior to the scenarios described below, the following communication took place between the NETS provider and CAMIUNS:

- NETS Provider System transmits a new TollDeclaration (Regular Message)

- CAMIUNS responds with TollDeclarationResponse ACCEPTED(OK) after an entry check
- CAMIUNS performs vehicle registration after 3 days and could not find a corresponding vehicle
- CAMIUNS response with a new TollDeclarationResponse ACCEPTED_ERROR(NO_REGISTRATION_FOR_VIN)

6.10.2. Scenarios

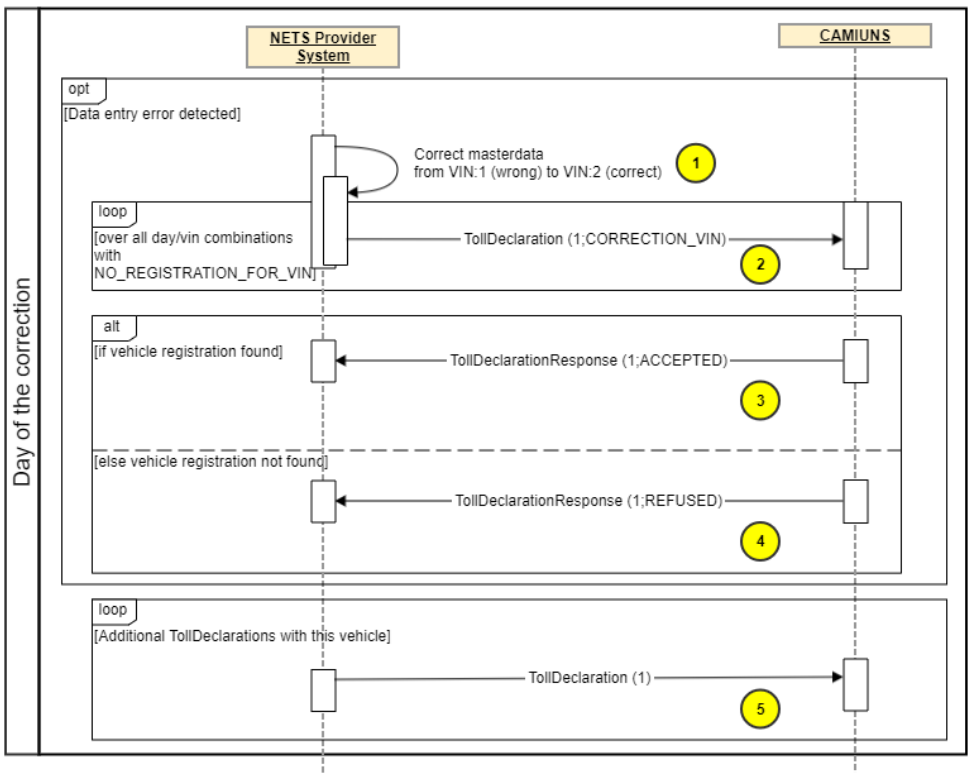
6.10.2.1. Data Entry Error

- The NETS provider detects a data entry error in the VIN.
- The error has to be reported using a manual TollDeclaration of type CORRECTION_VIN (as described in TollDeclaration (NO_REGISTRATION_FOR_VIN))

In the following example, we assume the following VIN's

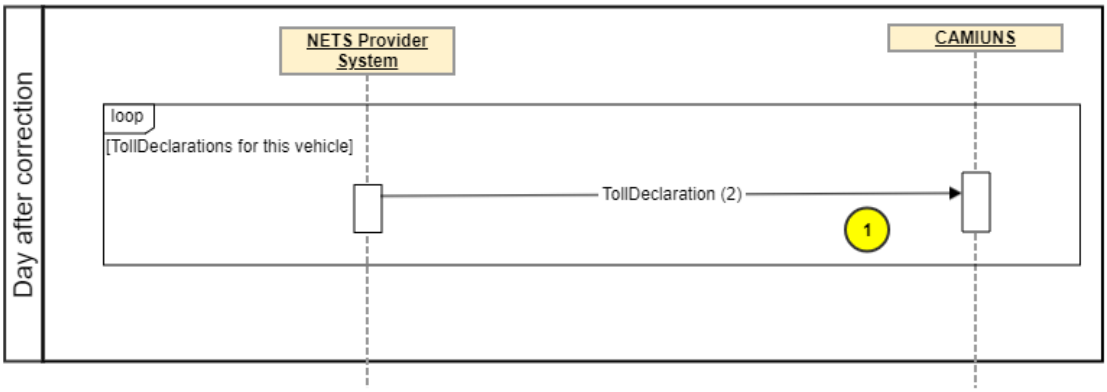
VIN	State
1	Old wrong VIN
2	Corrected VIN

6.10.2.1.1. Behaviour on day 1 (Day of the VIN correction)



1	The error is corrected in the master data.
2	Every VIN/Day combination can have multiple TollDeclarations. Every TollDeclaration can have multiple TollDeclarationResponses. For every VIN/Day combination with (at least one) TollDeclarationResponse of type ACCEPTED_ERROR(NO_REGISTATION_FOR_VIN) a single manual correction of type CORRECTION_VIN must be provided. If there are more than one TollDeclarationResponse of type ACCEPTED_ERROR(NO_REGISTRATION_FOR_VIN), only one manual correction of type CORRECTION_VIN must be provided. Otherwise the second manual correction of type CORRECTION_VIN will be refused, due to multiple correction vins.
3	The TollDeclarationResponse of the CORRECTION_VIN Request is either ACCEPTED(OK) or REFUSED, depending on whether the corrected VIN could be resolved in the vehicle register.
4	In the case of REFUSED, further manual clarifications are necessary. It is possible that the Road Traffic Office has not yet updated the change or that there is another error.
5	Any manual correction (or regular tollDeclaration) for a specific day/VIN combination already corrected with CORRECTION_VIN must still be transmitted with the old, incorrect VIN.

6.10.2.1.2. Behaviour on day 2 (Day after the VIN correction)



1	The master data has been corrected the day before and no TollDeclaration has yet been transmitted for this day/VIN. Now only the new VIN from the master data must be used.

6.10.2.2. Vehicle Registration Errors

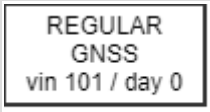
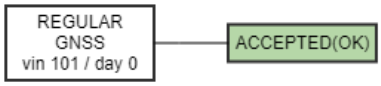
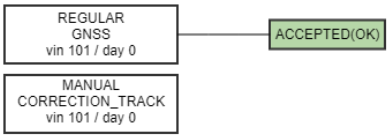
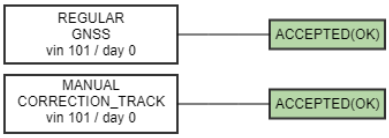
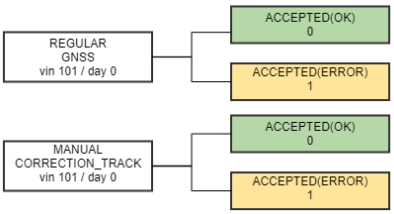
- The NETS provider receives tollDeclarationResponse ACCEPTED_ERROR(NO_REGISTRATION_FOR_VIN)
- There is no data entry error

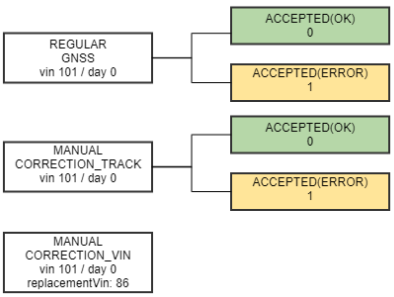
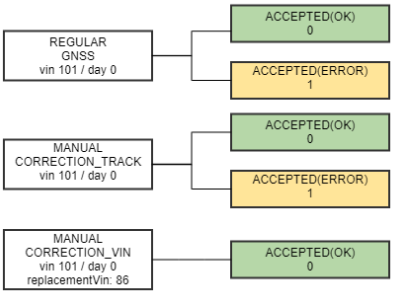
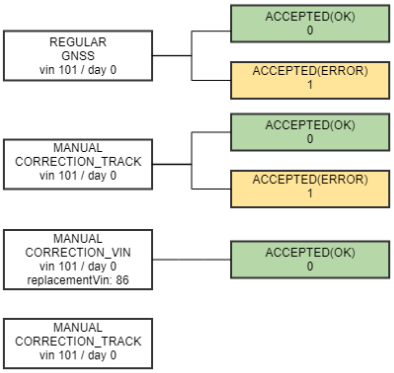
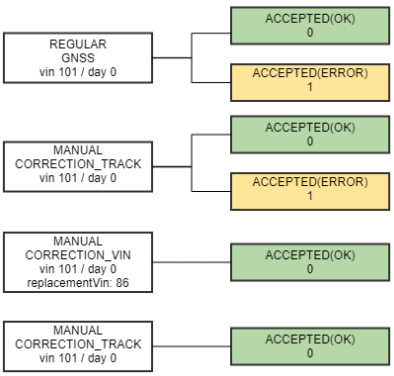
Info
Without an error in the master data, there is no need for the NETS provider to interact.

In the event of a delay at the road traffic office, the vehicle registration is automatically triggered and the corresponding TollDeclarations are answered positively

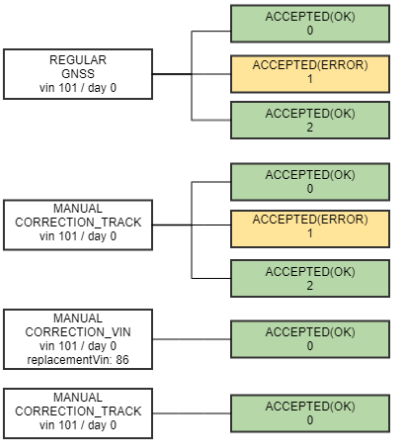
6.10.2.3. Toll Declaration Lifecycle in case of VIN Correction (Example)

This example shows the lifecycle of the Toll Declaration Responses issued to the NETS Provider:

Day	What happend?	Situation on Provider
0	The truck with vin 101 made a journey inside switzerland on day 0. We assume in this example that the VIN is misconfigured, the real vin would be 86.	BAZG responds with TollDeclarationResponse ACCEPTED(OK) for the manual TollDeclaration
1	A TollDeclaration (Regular Message) of type GNSS for the day of journey (day 0) is send to BAZG with a wrong configured VIN	
1	BAZG responds with TollDeclarationResponse ACCEPTED(OK) for the regular TollDeclaration	
2	A TollDeclaration (Manual Correction Message) of type CORRECTION_TRACK for the day of journey (day 0) is send to BAZG with a wrong configured VIN	
2	BAZG responds with TollDeclarationResponse ACCEPTED(OK) for the manual TollDeclaration	
4	BAZG made a vehicle registration check and could not find the VIN 101. Now the previously accepted answers suddenly have a different status.	

Day	What happend?	Situation on Provider
5	NETS Provider corrects the misconfigured VIN in the master data and transmits a TollDeclaration (Manual Correction Message) of type CORRECTION_VIN	
5	BAZG responds with TollDeclarationResponse ACCEPTED(OK) for the manual TollDeclaration	
6	A additional TollDeclaration (Manual Correction Message) of type CORRECTION_TRACK for the day of journey (day 0) is send to BAZG with the wrongly configured VIN. Even if the VIN is corrected, the old one have to be used for the whole day 0.	
6	BAZG responds with TollDeclarationResponse ACCEPTED(OK) for the manual TollDeclaration	

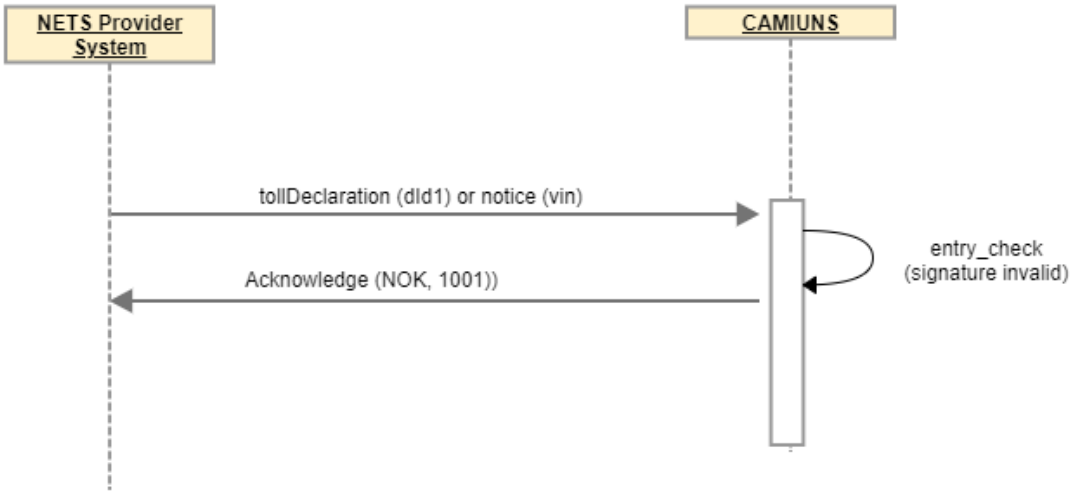
10

Day	What happend?	Situation on Provider
12	BAZG activates the declaration (no more changes possible) and sucessfully processes the corrected vin. So the 2 Responses with ACCEPTED(ERROR) are overwritten with ACCEPTED(OK).	

6.11. TollDeclaration, Notice (SIGNATURE_INVALID)

6.11.1. With result NOK (1001)

- The system of the NETS provider sends a regular declaration with declaration id (did1).
- During entry check CAMIUNS recognizes a ISSUER_ID_UNKNOWN.
- CAMIUNS rejects the message with REFUSED and ISSUER_ID_UNKONWN.

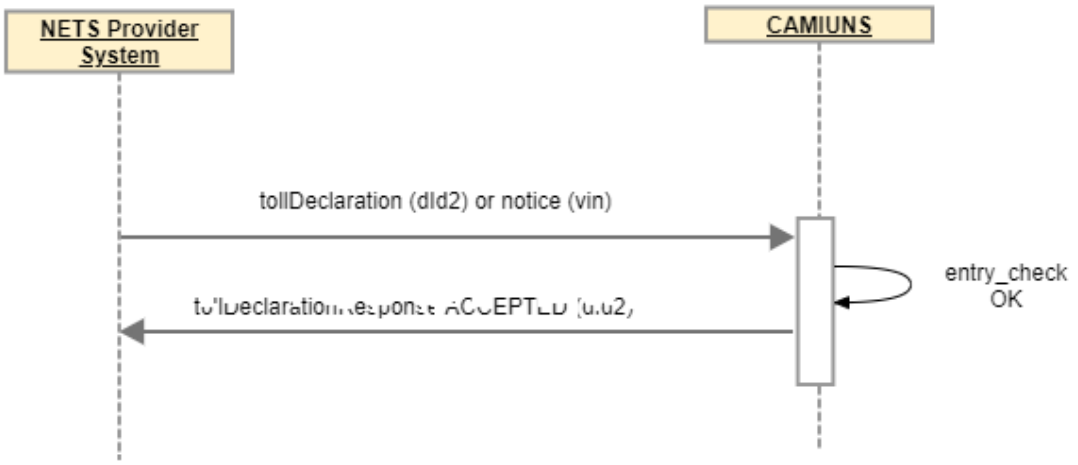


The scenario is also valid for messages of the type notice.

6.11.2. resending with result ACCEPTED

- The system of the NETS provoider successfully correts ISSUER_ID_UNKNOWN.

- Note: the same declarationId (did1) cannot be used to resend the message.



The scenario is also valid for messages of the type notice.

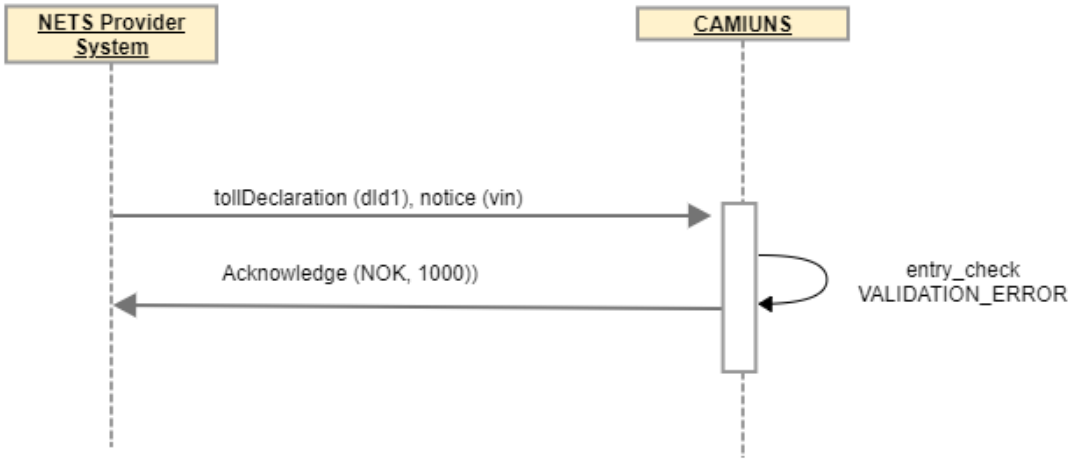
6.12. TollDeclaration, Notice (VALIDATION_ERROR)

Validation error can occur on two levels. The schema level and the application level. All errors which are on schema level, and therefore the corresponding message cannot be answered by BAZG with a correlating TollDeclarationResponse, are handled on schema level.

6.12.1. TollDeclaration, Notice (VALIDATION_ERROR Schema Level)

6.12.1.1. With result NOK (1000)

- The system of the NETS provider sends a regular declaration.
- An error occurs at schema level.
- CAMIUNS rejects the message with ACKNOWLEDGE NOK.

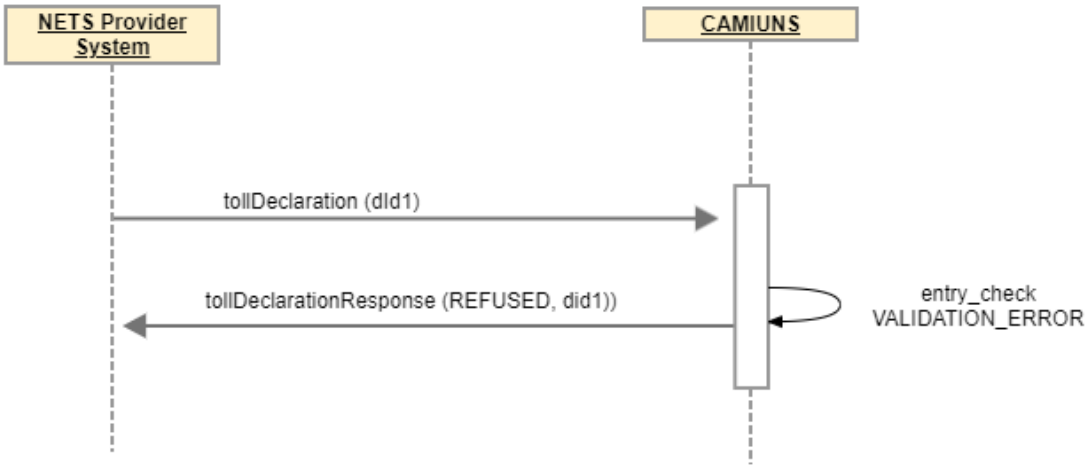


The scenario is also valid for messages of the type notice.

6.12.2. TollDeclaration (VALIDATION_ERROR Application Level)

6.12.2.1. With result REFUSED

- The system of the NETS provider sends a regular declaration.
- During entry check CAMIUNS recognizes a SCHEMA_VALIDATION_ERROR.
- CAMIUNS rejects the message with REFUSED and VALIDATION_ERROR. Validation error is used for different kind of general validation errors.
  The field reason text holds the effective reason for the validation error.



The scenario is also valid for messages of the type notice.

6.13. Notice (REGISTRATION)

6.13.1. Notice (REGISTRATION_BEGIN)

Once a provider wants to take a vehicle under service he issues a notice of type REGISTRATION_BEGIN for the corresponding VIN.



6.13.2. Notice (REGISTRATION_STATUS)

For each notice REGISTRATION_BEGIN, CAMIUNS will specify the status of the requested vehicle.

Note: Vehicles for tollDomain (A) shall only be taken under service by the provider if confirmed by BAZG as registered and only after effective date. Vehicles to be registered in Switzerland (tollDomain (CH)) can always be taken under service. However, the entitlement to remuneration only exists as long as the vehicle is registered with the road traffic office.



6.13.3. Notice (REGISTRATION_END)

6.13.3.1. Notice (REGISTRATION_END - Termitated by BAZG)

In cases where BAZG recognizes a reason to terminate the service for a given vehicle it sends a notice of type registration end and corresponding reason:

Following reasons are possible:

- PERMISSION_NO_LONGER_GRANTED
- DECLARATION_FROM_DIFFERENT_PROVIDER_EXISTS
- REGISTRATION_CANCELLED

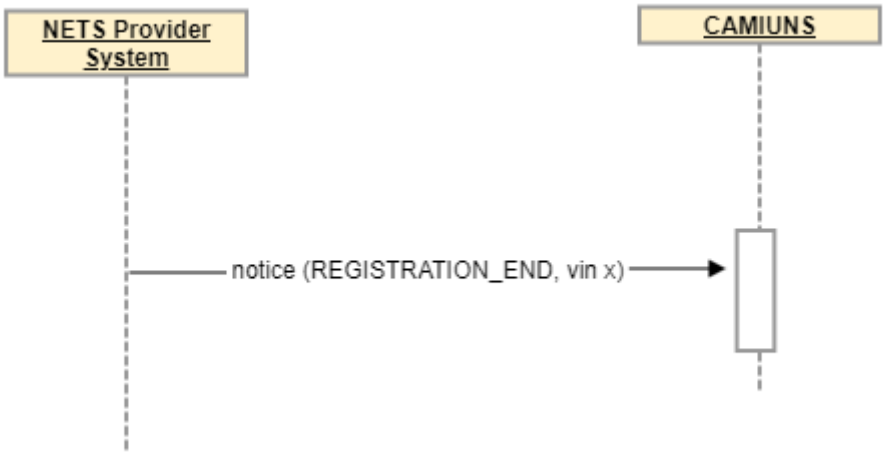


Note:

- Registration end is used by BAZG to signal the end of remuneration for a vehicle.
- A new REGISTRATION_BEGIN has to be sent, once a REGISTRATION for a vehicle has been signaled by REGISTRATION_END.

6.13.3.2. Notice (REGISTRATION_END - Terminated by the holder via NETS provider)

The service has been cancelled by the holder informing the provider. The provider therefore has to singnal BAZG the termination of the serice.



Note:

- Remuneration for a vehicle ends.
- A new REGISTRATION_BEGIN has to be sent if the service for the vehicle shall be continued.