Clearing Platform

REST API Specification

Stand: **03.11.2026**

Version: **1.3**

Notice

Copyright © by the Authors 2023 and Arbeitskreis Schnittstellen und Prozesse . All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works.

License might be changed, and the document might be published under open-source license at a later point in time based on agreements with “Arbeitskreis Schnittstellen und Prozesse”.

Authors

holos supply GmbH	Contact Rene Schäflein Tel.: +49 (0) 1522 278 17 18 rene.schaeflein@holossupply.de
BITConEx GmbH	Contact Halid Zehic Tel.: +49 (0) 160 / 7839658 halid.zehic@bitconex.de

Other applicable documents

The following documents are **mandatory** parts of this specification

[MAIN 1a]	**Clearing_Platform_OpenAPI_Definition_V1.3.png** Clearing Ticket Resource Model diagram of chapter Clearing Ticket Resource Model4.1
[REF-1]	**Clearing_Platform_non_functional_requirements_V1.0.docx** Covers the security, data privacy and availability issues **(unchanged for this version)**
[REF-2]	**Clearing_Platform_OpenAPI_Definition_V1.3.yaml** Formal OpenAPI specification for the REST interfaces in YAML format, and the recommended foundation for implementing the service provider and consumer interfaces as well
[REF-3]	**Clearing_Platform_Metadata_V1.3.xlsx** Specification of clearing scenarios, complete and obligatory list of attributes (meaning, data types), as well as the mapping rules that define the exact usage of attribute(s) per scenario
[REF-4]	**Clearing_Platform_Metadata_V1.3.yaml** The data specified in [**REF-3**] in machine consumable YAML format, intended to enable an automatic validation of the clearing data payload.
[REF-5]	Clearing_Platform_Metadata_Schema_V1.1.yaml Formal JSON schema specification for the platform metadata [REF-4] (unchanged for this version)
[REF-6]	Clearing Platform Specification Test Cases.xlsx Testcases for the validation of a Clearing Platform (unchanged for this version)
[REF-7]	**Clearing_Platform_NCO_handling_V.1.1** Rules for handling NCOs: Non-Clearing Organizations

A central list of abbreviations is used **for all documents and specifications in the "Arbeitskreis Schnittstellen & Prozesse"**, which can be found under the following link:

https://ak-spri.de/arbeitskreis-und-arbeitsgruppen/glossar

Content

Notice 1

Other applicable documents 2

Content 3

1.1 Business Domain 3

1.1.1 ClearingTicket 3

1.1.2 ItuCarrier 4

1.1.3 Attachment 4

1.2 Definition of roles 4

1.2.1 API Specification 4

1.3 HTTP Response Codes 5

1.4 Support of polymorphism and extension patterns 5

2 Sample Use Case 6

3 Lifecycle 8

3.1 Transition rules 10

3.2 States and Transitions 11

4 Clearing Ticket 15

4.1 Clearing Ticket Resource Model 15

4.2 ClearingTicket Operations 30

4.2.1 Retrieve a (single) clearing ticket 30

4.2.2 List clearing tickets 32

4.2.3 Create a clearing ticket 37

4.2.4 Set clearing ticket status 41

4.2.5 Set clearing ticket resolved 44

4.2.6 Set clearing ticket clearing data 47

4.2.7 Set clearing ticket severity 50

4.2.8 Add a note to an existing clearing ticket 52

5 Attachment 55

5.1 Attachment Resource Model 57

5.2 Attachment Operations 59

5.2.1 Create Attachment 59

5.2.2 Retrieve Attachment 59

5.2.3 Delete Attachment 60

6 ITU Carrier 61

6.1 ITU Carrier Resource Model 61
6.2 ITU Carrier Operations 62
6.2.1 List ItuCarrier 62
6.2.2 Retrieve ItuCarrier 63
6.2.3 List of supported scenarios 63
7 Notifications 64
7.1 Notification Resource Models 65
7.2 Notification API 66
7.2.1 Register listener / Unregister listener 66
7.2.2 Notify about ticket creation 66
7.2.3 Notify about ticket data change 68
7.2.4 Notify about ticket status change 70
7.2.5 Notify about ticket resolved 72
7.2.6 Notify about ticket severity change 74
7.2.7 Notify about new ticket note 76
8 Appendix 78
8.1 Special Rules 78
8.1.1 Rules for severity 78
8.1.2 Rules for date and time 79
8.1.3 Rules for requestedResolutionDate 80
8.1.4 ResolvedSuccessfully 81
8.1.5 Rules for prolongationChange 82
8.1.6 Rules for Attachments 84
8.2 Understanding Clearing Metadata 85
9 Change Log 91
9.1 Version 1.0 to version 1.1 91
9.1.1 Lifecycle: Status initial and acknowledge 91
9.1.2 List of attribute changes 91
9.1.3 Adjustments 91
9.2 Version 1.1 to version 1.2 92
9.2.1 Version 1.2 to version 1.2.1 92
9.2.2 Version 1.2.1 to version 1.2.2 93
9.2.3 Version 1.2.2 to version 1.3 93

1 Introduction

The following document is the "*Clearing Ticket for Partner API REST Specification*" of the Clearing Platform. It is part of a document collection for a "Clearing Platform".

1.1 Business Domain

It includes model definitions as well as all available operations for the following resources of the business domain:

- ClearingTicket
- Attachment
- ItuCarrier

1.1.1 ClearingTicket

The API supports the ability to create and retrieve a clearing ticket, changing the status, severity and adding a note. It also supports the ability to create and retrieve an attachment.

Notifications are defined to provide information when a clearing ticket has been created, status changes, severity changes and notes.

A set of states of a clearing ticket has been specified to handle clearing ticket lifecycle management.

Clearing Ticket API performs the following operations on a clearing ticket.

- Retrieval of a clearing ticket or a list of clearing tickets.
- Creation of a clearing ticket
- Clearing Ticket modifications
    - status change / resolved
    - severity change
    - data change
    - new note
- Notification of events on clearing ticket:
    - creation
    - status change / resolved
    - severity change
    - new note

Note: In the following text the names *ClearingTicket* and *TroubleTicket* are not always used concisely. Either way we always mean the same namely the top-level container object.

1.1.2 ItuCarrier

Operations to retrieve a list of “valid” ItuCarrier (those connected to the Clearing Platform).

1.1.3 Attachment

Operations to store and retrieve attachments, for use in a clearing ticket.

1.2 Definition of roles

In every clearing case exactly two subject specific participants are involved:

The ORIGINATOR is the participant having an issue that shall be cleared.

The PROCESSOR is the participant that is solving the issue.

Both originator and processor as well are referred to as PARTNERS subsequently.

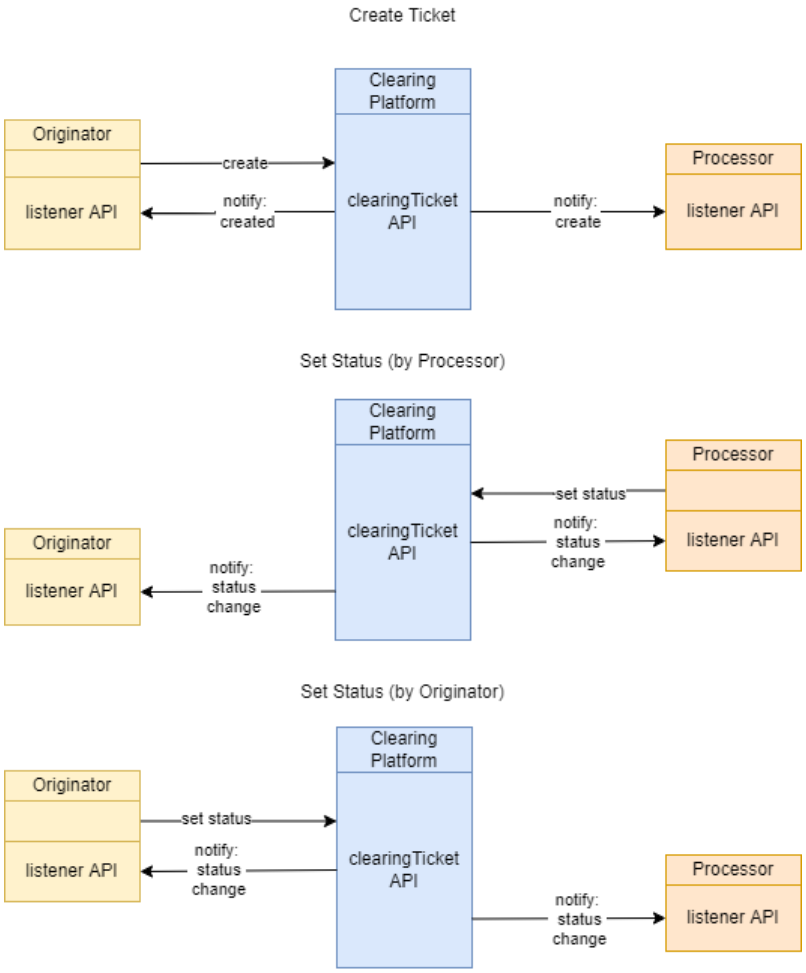
1.2.1 API Specification

The Clearing Platform REST API Specification consists of two different parts:

1. This part is implemented by the clearing PLATFORM as the service provider, and used by the partners as service consumer.
   It contains:
   - Clearing Ticket API
   - Attachment API
   - ITU Carrier API

2. This (optional) part is intended to be implemented by the PARTNER connected to the Clearing Platform via API. The Clearing Platform in turn will use the Notification API to raise the notification events.

The following figure illustrates the interaction of the clearing platform and partner(s)



1.3 HTTP Response Codes

Whenever an API returns a meaningful http response code this is explicitly documented in the API section. Only those codes are then also mentioned in the OpenApi definition: See [REF-2]

Still all other http response codes can occur, but are not listed for the sake of clarity.

Examples for those codes are:

401	`Unauthorized`
405	`Method Not allowed`
415	`Unsupported Content Type`
500	`Server Error`

1.4 Support of polymorphism and extension patterns

For this API we do not make any use of polymorphic (mixed) collections.

However, some types are extensions of base types, e.g., `ClearingTicket` extends the TMF-621 `TroubleTicket` schema.

Here we support the attributes `href`, `@type` and `@baseType`.

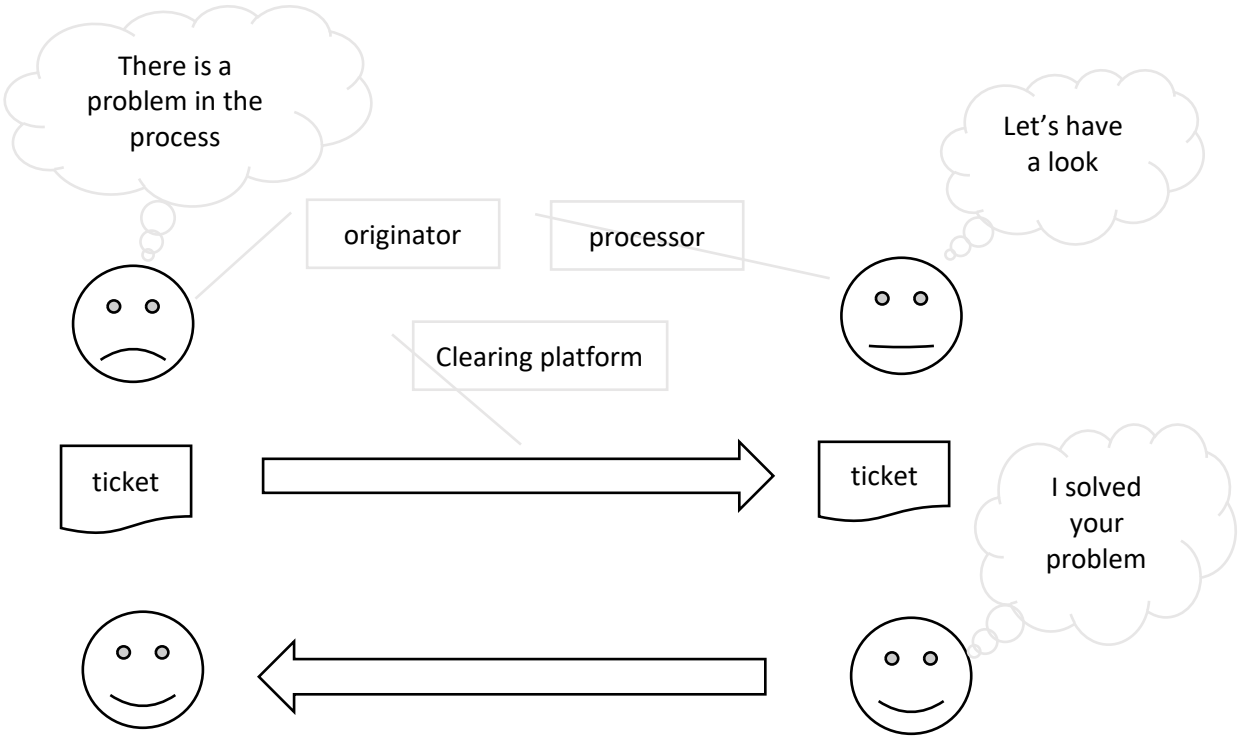
2 Sample Use Case

Today the telecommunication companies in Germany have established multiple APIs and processes to interact. If one of the processes does not work as expected or if there is no API in place, one can raise a clearing case to solve the issue.

The carrier who needs a solution from another party creates a “clearing ticket”. This party is called the (clearing ticket) **originator**.

The clearing ticket is then passed by the clearing platform to any requested party who shall offer a solution. This party is called the (clearing ticket) **processor**.

A simple Use Case

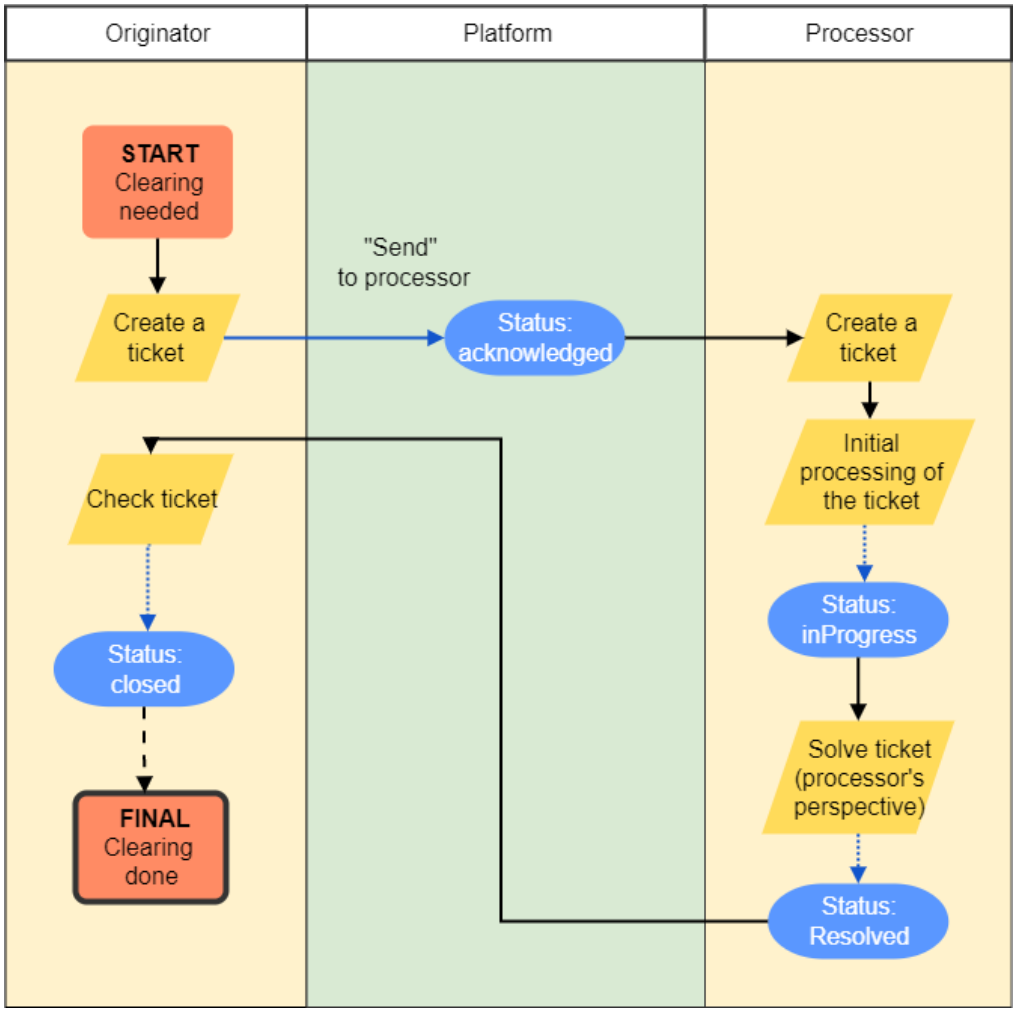


The following sequence will illustrate simplest possible flow of actions to process a clearing ticket.

It assumes that **both parties are connected** to a clearing platform **via the partner API** described in this specification document.

For scenarios where one or both participants are using their respective platforms GUI, the picture is slightly different, because the actions performed by a GUI user take place in the platform itself, but not in another system. However, the actions and state changes are the same, no matter if performed in a platforms GUI or by a partner's system.

Note: Originator and processor may be connected to the **same or different** platforms. For simplicity, the fact that originator and processor MAY be connected to **different** platforms is omitted in this example.



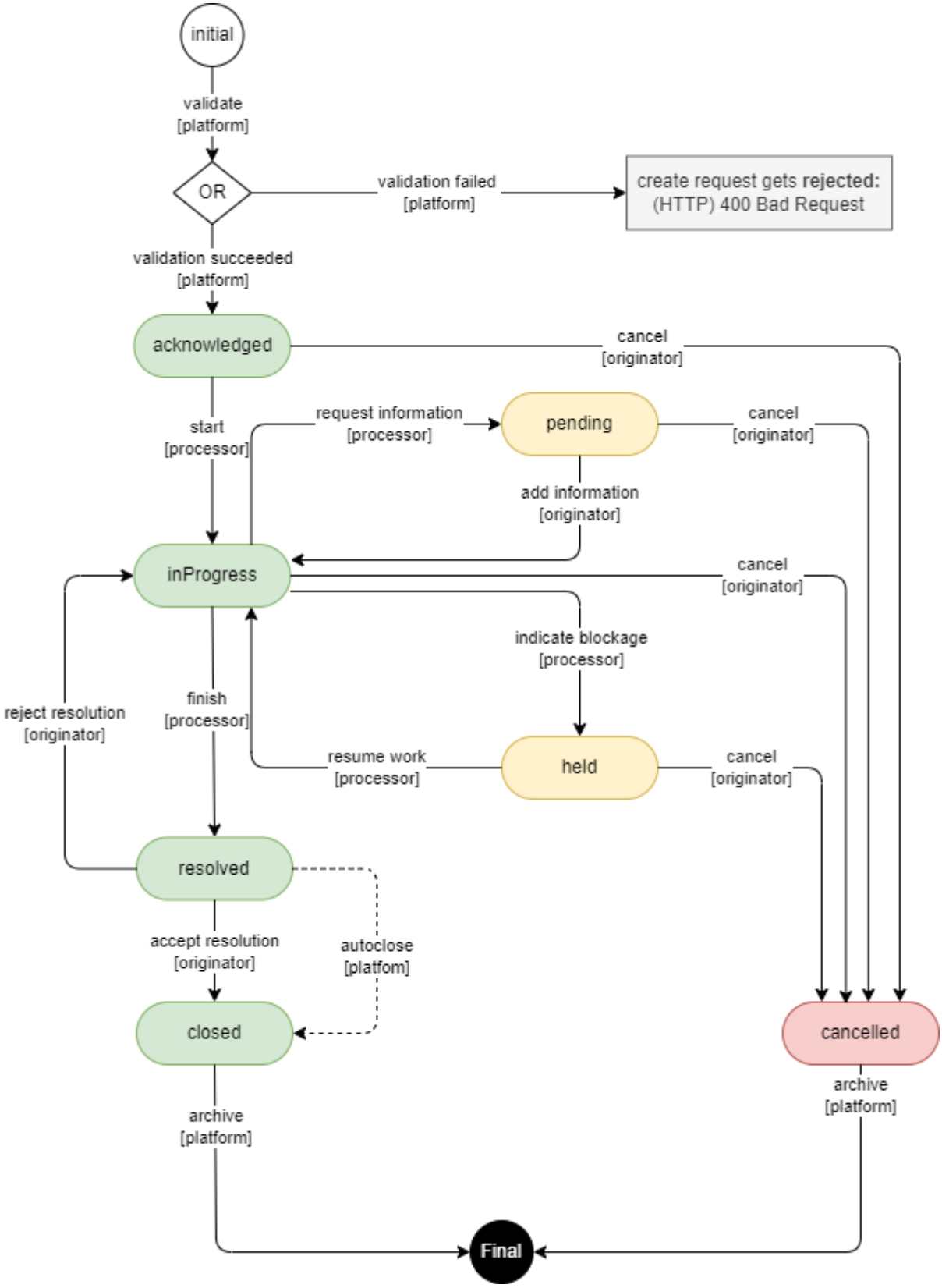
1. the originator creates ticket within its own ticket system.
2. the ticket is passed to the clearing platform via partner API.
3. the clearing platform receives the ticket, creates a unique ID for it and returns the ID to the originator
4. the clearing platform validates the ticket (successfully), moves the ticket to `acknowledged` state, and notifies the processor about the new ticket

5. the processor creates a ticket in its own internal ticket system
6. the processor will start working on the ticket and thus request the platform to set the status to `inProgress`.
7. the processor performs the necessary steps and eventually solves the ticket. To indicate this, the processor sets the tickets state to `resolved`.
8. The platform notifies the originator about the state change
9. The originator checks the solution and sets the ticket to status `closed`

3 Lifecycle

The lifecycle of a clearing ticket is characterized by a sequence of states.

The following UML2 state machine diagram specifies the status values a clearing ticket can take during its lifecycle, as well as the allowed transitions originating from these states. Constraints that limit who may initiate a state transition are specified as UML guard conditions (in square brackets).



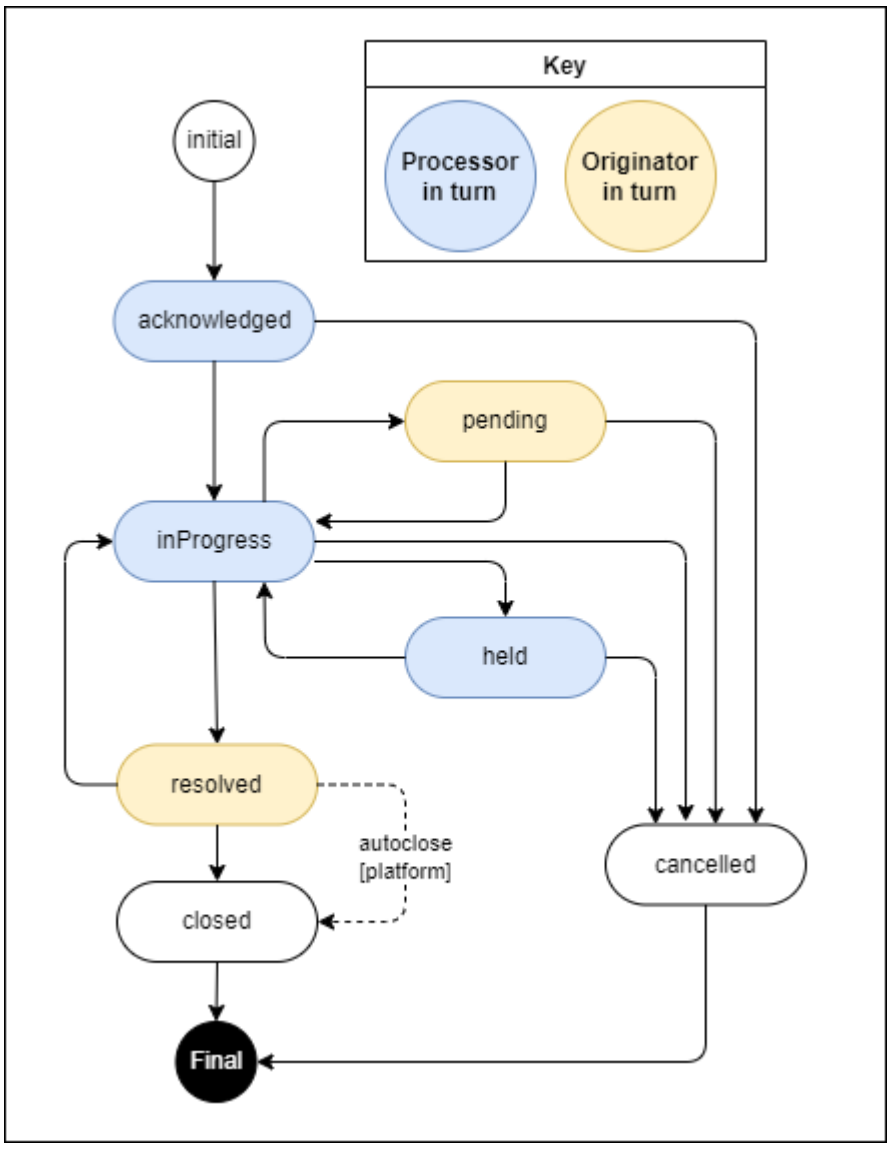
The states are colored. The green states indicate the “happy path”. If there are obstacles leading to delays the states are shown in yellow. If the processing of the ticket is not possible (anymore), the state is colored in red. Internal states are either white or black.

The state of a clearing ticket at any point in time is managed by the field `status` (modelled as a `StatusChange` object). The history of the events (state changing operations) leading to the current state is recorded in `statusChange`. For more details, see chapter 4.1 Clearing Ticket.

3.1 Transition rules

At any point of the lifecycle, **only one** of the participants may request a state change. Besides the internal states `initial` and `final`, either the originator or the processor may trigger a state change. As an **exception** from this rule, the originator can cancel the ticket at any state if it is neither already resolved nor failed (the `acknowledged`, `inProgress`, `pending` and `held` states).

To clarify which participant can perform state change transitions, a simplified version of the state diagram has been added. In the following diagram, **BLUE** indicates states the processor is in turn, while in **YELLOW** states it's the originators turn. No color means NONE of the participants can perform ANY action in this state.



State changes (transitions) are requested by a client (either Originator or Processor) at the respective platform, by one of the following operations:

- `createTicket` → `TroubleTicketCreateEvent`
- `setStatus` → `TroubleTicketStatusChangeEvent`

Each operation is followed by a corresponding notification event issued by the platform AFTER the state change has been performed successfully. This way, a platform can guarantee that clients always have a consistent view on the current state, no matter if they process notification events, poll the status from the platform, or both.

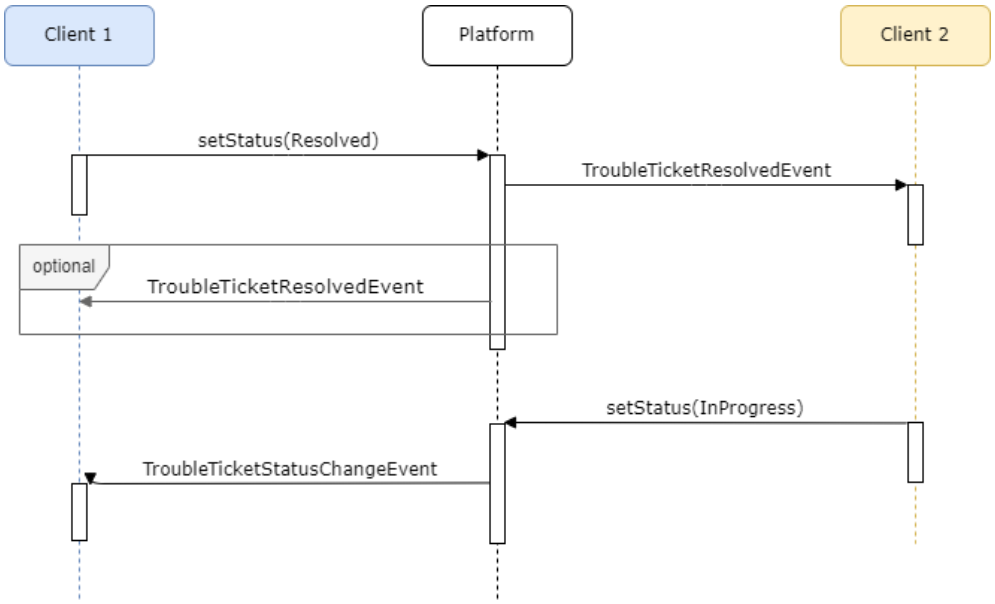
Autoclose

If a ticket is resolved by the processor and the originator does not take any action within 30 calendar days, the ticket will be automatically closed by the platform. A status change notification will be sent to both parties (originator and processor) in any case, even if the Mirror Notifications are deactivated.

Mirroring Notifications

Clients can opt in during onboarding to receive notifications for operations issued by themselves. By default, a platform will not send notifications to the client that triggered the operation.

To illustrate this, consider the following situation: There are two clients **Client1** and **Client2**, where **Client1** has opted in to receive notifications for its own changes, and **Client2** has not. Now, **Client1** requests a state change to `resolved` at the platform, and the platform sends a `TroubleTicketResolvedEvent` to **Client2** as well as **Client1** (because **Client1** opted in for this). Next, **Client2** requests a state change to `inProgress`, and **Client1** gets notified with a `TroubleTicketStatusChangeEvent`, but **Client2** not:



The operations and their corresponding notification events are specified in full detail in the chapters 4.2 ClearingTicket Operations and 7 Notifications.

3.2 States and Transitions

The following table explains in detail the intended semantics for each state, the transitions originating from the respective state and the conditions that apply. The “Actor” column contains the participant who may perform the transition.

To initiate a state, change the operation `setStatus` is used.

State / Transition	Actor	Description
`initial`	Platform	A ticket has been received by the originator’s clearing platform (see `createTicket` operation). This state is only used temporarily within the platform and never be seen outside. The platform must perform a validation against the rules specified in 0. Important: NO events are raised at this point. Especially, the processor will NOT be notified about the ticket before the validation was successful.
→ validation succeeded	Platform	The validation was **successful**. The ticket is now accepted for further processing and moved to `acknowledged` state. A `TroubleTicketCreateEvent` is issued to notify the participants (most important, the processor) about the new ticket. (HTTP) 201 Created is returned with body : ClearingTicket, ticket id is set and status is `acknowleged`.
→ validation failed	Platform	The validation **failed**. **No ticket will be created** and no `TroubleTicketCreateEvent` is raised. `(HTTP) 422 = unprocessable content` is returned with body: `Error object`.
`acknowledged`	Processor	The ticket has been accepted for processing by the platform. The processor has been notified about the new ticket, via a `TroubleTicketCreateEvent` and is expected to start working on the ticket soon. No decision making by the processor is required nor possible at this point. This makes this state suitable for automated processing on the processor side.
→ start	Processor	The processor signals the **reception** of the ticket, as well as the **intention to start working** on the resolution now actively. The ticket is moved to the `inProgress` state. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.

→ cancel	Originator	The originator signals he is no longer interested in the resolution of the ticket. The ticket is moved to the `cancelled` state. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.
`inProgress`	Processor	The ticket is being processed by the processor.
→ finish	Processor	The processor indicates the completion of processing. Information about whether the issue raised by the originator has actually been resolved or not is provided in the `resolvedSuccessfully` field (true means success). In either case, the processor can provide additional information in the `changeReason` field and may also include attachments. If `resolvedSuccessfully` is false, providing changeReason is mandatory The ticket is moved to the `resolved` state. A `TroubleTicketResolvedEvent` is raised to indicate the state change.
→ cancel	Originator	see above
→ request information	Processor	The processor needs the originator to provide extra information to be able to solve the ticket. The ticket is moved to the `pending` state. A `TroubleTicketStatusChangeEvent` is raised to notify the originator.
→ indicate blockage	Processor	The processor signals a temporary obstacle stopping him from working on the ticket at the moment. The ticket is moved to the `held` state. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.
`resolved`	Originator	The ticket has been resolved by the processor (either successfully or not), the originator is expected to either accept the solution and close the ticket or send the ticket back to the processor. Autoclose: If the originator does not take any action within the expected response time (30 days) the ticket will be automatically closed by the platform.
→ accept resolution	Originator	The originator accepts the solution for the ticket. The ticket is moved to the `closed` state.

		A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.
→ reject resolution	Originator	The originator does not accept the solution provided by the processor, and requests further processing. The ticket is moved back to the `inProcess` state. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change. The originator SHOULD provide additional information, which may enable the processor to resolve the ticket.
`pending`	Originator	The processor has stopped working on the ticket and expects the originator to provide updated or additional information.
→ add information	Originator	The originator provides the information requested by the processor, see 4.2.6 Set clearing ticket clearing data Set clearing. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change, see 7.2.3 Notify about ticket data change. Then the ticket must be set to `inProgress` state, see 4.2.4 Set clearing ticket status. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.
→ cancel	Originator	The originator cannot provide the information requested by the processor or cancels the ticket for other reasons. The ticket is moved to the `cancelled` state. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.
`held`	Processor	The processor has temporarily stopped working on the ticket, due to a reason on the processor's side. The processor is expected to resolve that blockage and resume working on the ticket as soon as possible.
→ resume work	Processor	The processor has resolved the blockage and continues working on the ticket. The ticket is moved back to the `inProgress` state. A `TroubleTicketStatusChangeEvent` is raised to indicate the state change.
→ cancel	Originator	see above
`cancelled`		Final state, no further processing is possible. The ticket is still available for enquiry.
→ archive	Platform	**14** calendar days after a ticket has reached the `cancelled` state, the platform will move the ticket to the `final` status.

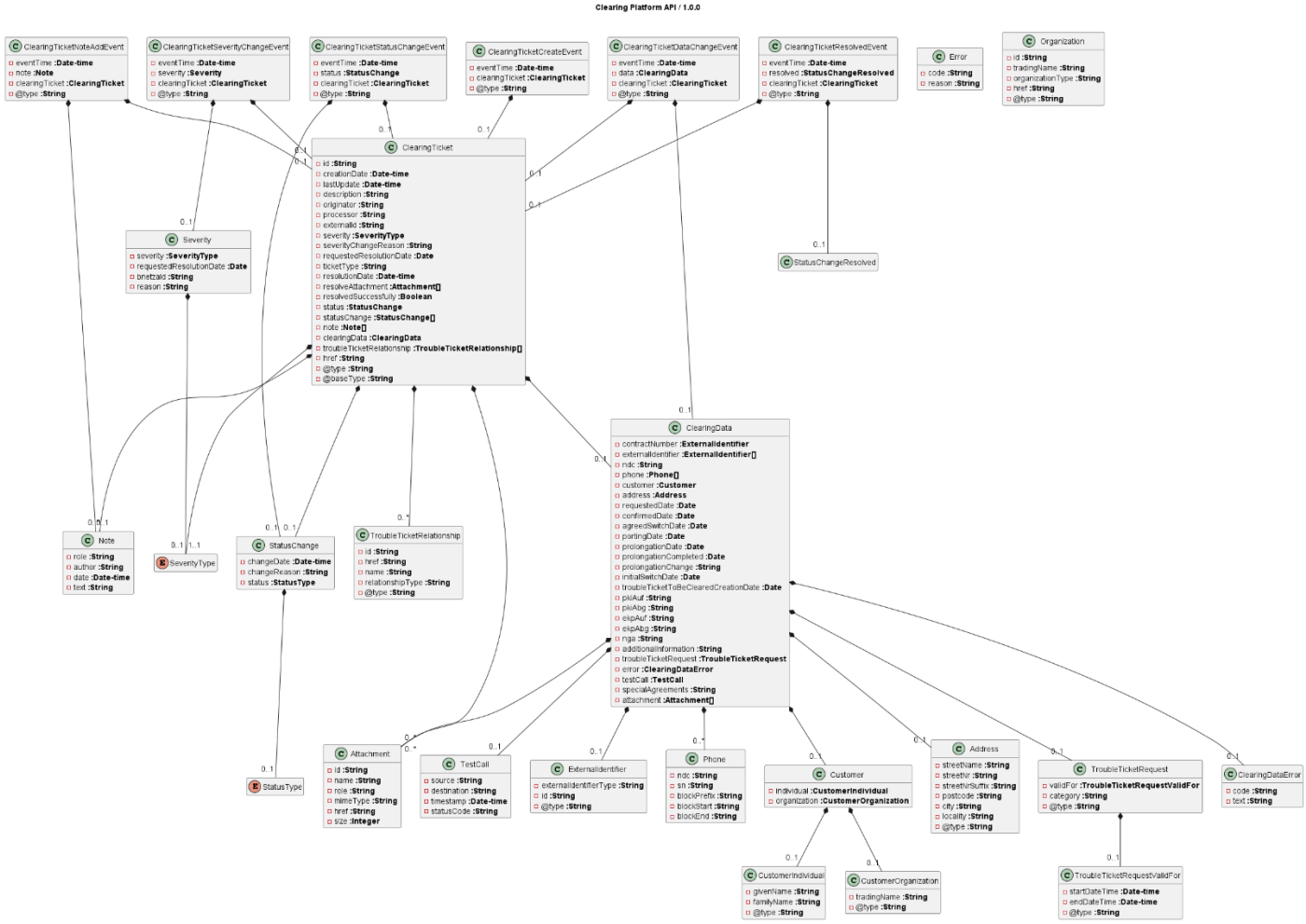
		A `TroubleTicketStatusChangeEvent` is raised to indicate the state change. **Note**: Applies to the `closed` states as well.
`closed`	Platform	Final state, no further processing is possible. The ticket is still available for enquiry.
`final`		The ticket is archived and will be removed from the platform **7** calendar days after being moved into `final` state. After this point, it is no longer available for enquiry. Therefore, **NO EVENT** is raised to indicate this change.

4 Clearing Ticket

4.1 Clearing Ticket Resource Model

A clearing ticket is a record of an issue that is created, tracked, and managed by a clearing ticket management system.

Resource model



Note: this image is available as a separate file ([1a MAIN] Clearing_Platform_OpenAPI_Definition_V1.3.png).

Field descriptions

ClearingTicket Fields

There is a list of mandatory parameters independently of the clearing cases.

These parameters are organized in the ClearingTicket structure. This structure must be present in any Clearing Ticket, whereas all the other structures are optionally present – refer to 8.2 Understanding Clearing Metadata. All fields are filled by the originator if not explicitly mentioned otherwise.

Mandatory fields are rendered **bold**. A more detailed description of mandatory fields is in chapter 4.2.3 Create a clearing ticket.

id	A string. Unique Identifier of the clearing ticket assigned by the Clearing Platform.
creationDate	A date time. The date and time that the clearing ticked was created – set by the platform.
lastUpdate	A date time. The date and time that the trouble ticked was last updated – set by the platform. The value of `lastUpdate` for the originator ticket and the corresponding processor ticket must always be identical.
description	A string. Description of the clearing scenario as defined in the list of clearing cases in AH4Clearing. Should be set to wildcard '*' by the originator and the Clearing Platform will fill the attribute with the proper clearing scenario description.
originator	A string. ITU carrier code of the carrier originating this clearing ticket.
processor	A string. ITU carrier code of the carrier this ticket shall be submitted to.
externalId	A string. Identifier assigned by the originator for this clearing ticket. **Note**: It is recommended to keep this identifier unique, but this will **not** be checked.
severity	A `SeverityType`. The severity of the ticket as defined by the AH4Clearing. The allowed values are: `regular`, `critical`, `escalated`. There are some complex rules for the severity described in *8.1.1* Rules for severity.
severityChangeReason	The reason for a severity change. Required as soon as severity is no longer regular.

requestedResolutionDate	A Date. The resolution date requested by the originator. See also *8.1.3* Rules for requestedResolutionDate.
ticketType	A string. **The id of the clearing scenario** as defined in the list of clearing cases in AH4Clearing. This is a crucial attribute as it defines the presence of all other clearing scenario specific parameters! See 8.2 Understanding Clearing Metadata
resolutionDate	A date time (DateTime). The date and time the trouble ticket was resolved. Set by the platform whenever the state changes to resolved. Note: This might happen several times. The attribute always keeps the latest change. **This field might NOT be filled by the originator but is rather taken by the platform from a resolve action.**
resolveAttachment[]	A list of `Attachment`s. File(s) attached to the clearing ticket by the processor when the status of the ticket is changed to resolved. Note: The attribute always keeps the value of latest resolve action. **This field might NOT be filled by the originator but is rather taken by the platform from a resolve action.**
resolvedSuccessfully	A boolean. Indicating if the clearing ticket was successfully solved as requested: (`true`) or else: `false`. See also *8.1.4* ResolvedSuccessfully. It should be used by the originator to decide if any further action is needed. Note: The attribute always keeps the value of latest resolve action. **This field might NOT be filled by the originator but is rather taken by the platform from a resolve action.**
status	A `StatusChange`. The status contains the current status of the clearing ticket.
statusChange	A list of `StatusChange`s. The status change history that are associated to the ticket. Populated by the server.
note	A list of `Note`s. The note(s) that are associated with the ticket.
clearingData	A set of `ClearingData`. Provided by the originator to identify the transaction that shall be handled by this clearing ticket.
troubleTicketRelationship[]	A list of `TroubleTicketRelationShip`s.
prolongationChange	A string. Reason for the change of prolongation, selected from a list of pre-assigned values. See also *8.1.5* Rules for prolongationChange

href	A string. Hyperlink, a reference to the trouble ticket entity.
@type	A string. The type of this entity. Must be: 'ClearingTicket'.
@baseType	A string. The base type of this entity. Must be: 'TroubleTicket'

StatusChange Fields

A Status change can only be processed depending on the current state of a clearing ticket. See the state model for details.

The changeReason is optional, however it must be filled with helpful information in case of these state changes:

- inProgress → held
- resolved → inProgress
- inProgress → pending
- pending → inProgress
- acknowledged, inProgress, pending, held → cancelled

In the case of a change of status from inProgress → pending, it must be clearly stated which information is not comprehensible or which information is still needed

(Mandatory fields are rendered **bold**.)

changeDate	A date time (DateTime). The date and time that the status was changed - set by the platform.
changeReason	A string. The reason for changing the status.
status	A `StatusType`. The current status of the clearing ticket. For allowed values see chapter 3 Lifecycle.

StatusChangeResolved Fields

changeDate	A date time. The date and time that the status was changed - set by the platform.
changeReason	A string. The reason for changing the status.
status	A `StatusType`. The current status of the clearing ticket. For allowed values see chapter 3 Lifecycle.

resolvedSuccessfully	A boolean. Indicating if the clearing ticket was successfully solved as requested (true) or false if any additional action is required by the originator.
resolveAttachment[]	A list of `Attachment`s. File(s) attached to the clearing ticket by the processor.

Note Fields

role	A string. The role of this note • empty for regular notes • 'system message' for notes sent by the platform
author	A string. Author of the note = ItuCarrier Id - or platform id for 'system message'. If there is a need to mention the real author (person) if necessary, this shall be mentioned in the text of the note.
date	A date time. Date and time when the note was added to the ticket.
text	A string. Text of the note - human readable.

Severity Fields

severity	A `SeverityType`. The severity of the issue as defined by the AH4Clearing. The allowed types are: `regular, critical, escalated`
requestedResolutionDate	A Date. The resolution date requested by the user.
bnetzaId	A string. Case id of 'Bundesnetzagentur'.
reason	A string. Reason for the severity change. Will be stored as `severityChangeReason`.

TroubleTicketRelationship Fields

Set by the originator to indicate a relationship between the new clearing ticket and the referenced clearing ticket. Please note that any partner may only view tickets that he owns as originator or processor. All other tickets are not accessible even if referenced here.
All parameters are filled by the originator. Mandatory fields are rendered **bold**.

id	A string. Unique Identifier of the referenced clearing ticket.
name	A **string.** Name of the Trouble Ticket.
relationshipType	A string. Type of the Trouble Ticket relationship. A text like “related ticket“, “Weiterversorgung start“, …

href	A string. Hyperlink, a reference to the referenced trouble ticket entity.
@type	A string: 'TroubleTicketRelationShip'.

ClearingData Fields

These fields all provide information to identify the order to be cleared or repeat data of the order. This data is provided by the originator of a clearing ticket and must not be changed by the processor.

Refer to 8.2 Understanding Clearing Metadata to see which attributes of the ClearingData have to be present depends on the clearing scenario.

contractNumber	An `ExternalIdentifier`. A contract number for the order to be cleared like WITA contract number, S/PRI contract number etc. A complete list of allowed `externalIdentifierType`(s) is provided in 0.
externalIdentifier[]	A list of `ExternalIdentifier`(s). Any other external identifier required in the clearing case to identify the order to be cleared. A complete list of allowed `externalIdentifierType`(s) is provided in 0.
ndc	A string. The National Destination Code (ONKZ /Ortsnetzkennzahl – without leading zero), e.g., 89, 5241, 30. For details refer to ITU-T E.101. Intended use: When clearing the ndc for an address.
phone[]	A list of `Phone`. Phone numbers associated with the clearing case.
customer	A `Customer`. An individual or organization used with the order to be cleared.
address	An `Address`. The address used with the ordered to be cleared.
requestedDate	A date. This is the original requested date for the order to be cleared, not the requested date the clearing should be completed.
confirmedDate	A date. Delivery date as confirmed by the supplier (VLT = Verbindlicher Liefertermin). In WITA or S/PRI environment this date is sent in the latest ABM message.
agreedSwitchDate	A date. Date as agreed upon between the EKPabg and the EKPauf in pre-negotiation (WBCI).

portingDate	A date. Date when the phone numbers are ported as used by PDA. (PDA = Portierungs-Daten-Austausch. Intended for prolongation = “Weiterversorgung”)
prolongationDate	A date. Date until the prolongation shall be active. Intended for “Weiterversorgung”.
prolongationCompleted	A date. Date the prolongation has been stopped. Intended for Weiterversorgung)
initialSwitchDate	A date. Latest agreed switching date before the prolongation (Ursprünglicher Wechseltermin). Intended for “Weiterversorgung”.
troubleTicketToBeCleared CreationDate	A date. Date the trouble ticket was created (not the clearing ticket).
pkiAuf	A string. Id of the new voice carrier (Portierungskennungs-Inhaber aufnehmend) e.g., D123 Note: *Portierungskennungen* are assigned by *Bundenetzagentur* and are not identical to the *ITU-Carrier Code*.
pkiAbg	A string. Id of the old voice carrier (Portierungskennungs-Inhaber abgebend)
ekpAuf	A string. End-customer contract holder new (Endkundenvertragspartner aufnehmend). ITU-Carrier Id
ekpAbg	A string. End-customer contract holder old (Endkundenvertragspartner aufnehmend). Itu-Carrier Id
nga	A string. Next Generation Access “Betreiber”. ITU-Carrier Id
additionalInformation	A string. Additional information required for solving this issue. This information shall be human readable.
troubleTicketRequest	A `TroubleTicketRequest`. New date and time for an appointment at the end customer’s site.
error	A `ClearingDataError`. The error code and text as transmitted in the original response that shall be cleared.
testCall	A `TestCall`. Data of a test call.
specialAgreements	A string. Any special agreements between the two partners specific for this clearing. Intended to identify a new clearing scenario.
attachment []	A list of `Attachment`s. Any pdf document or picture or list of phone numbers required to solve the issue.

Phone fields

Used to carry an E.164 based phone number.

Either the `sn` (subscriber number) must be present for a single number - or the complete block data exclusively.

Either the `sn` (for a single number) or the entire block must be present.

This structure is provided by the originator and will not be changed by the processor.

Mandatory fields are rendered **bold**.

ndc	A string. The National Destination Code (ONKZ /Ortsnetzkennzahl – without leading zero), e.g., 89, 5241, 30. For details refer to ITU-T E.101
sn	A string. The Subscriber number part of this phone number. (Hauptwahl). Either this `sn` or (exclusive) a `block` has to be present. For details refer to ITU-T E.101
blockPrefix	A string. Prefix of the number block. e.g., 181
blockStart	A string. First number in the number block. e.g., 000
blockEnd	A string. First number in the number block e.g., 499

Customer fields

Structure to identify a customer in the clearing context, all elements are provided by the originator and are not changed by the processor.

The customer may **either** be an Individual (exclusive) or an Organization. It is not intended to provide both elements as the APIs to be cleared (WITA, WBCI, S/PRI) also allow only one of these elements

individual	A `CustomerIndividual`. The individual customer as used by the original order.
organization	A `CustomerOrganization`. The organization customer as used by the original order.

CustomerIndividual fields

All parameters are filled by the originator. Mandatory fields are rendered **bold**.

givenName	A string. Given name of the customer as used in the original order.
familyName	A string. Family name of the customer as used in the original order.

@type	A string. 'Individual’

CustomerOrganization fields

All parameters are filled by the originator. Mandatory fields are rendered **bold**.

tradingName	A string. Trading name of the customer as used in the original order.
@type	A string. 'Organization’

Address fields

Whenever an address is required to identify the order to be cleared, this structure shall be used. This structure is not intended to communicate with the end-customer but to identify the original process for clearing purposes. The address shall be provided exactly as in the original “order”.

If not specified elsewhere, this address is always the line address.

All parameters are filled by the originator. The values are derived from the currently used APIs like WIT, S/PRI and WBCI. Mandatory fields are rendered **bold**.

streetName	A string. Name of the street
streetNr	A string. Number of the house in the street – only the numerical part.
streetNrSuffix	A string. Suffix of the house number, e.g. a, -129, III.
postcode	A string. Postal code of the address. Always 5 digits with leading “0”.
city	final
locality	A string. Name of the locality = suburb or district.
@type	A string. ‘GeographicAddress’

TroubleTicketRequest fields

This structure is only used to carry a new end-customer appointment from the originator to the processor in cases the new appointment may not be transferred using the regular API (ESS).

All Parameters are filled by the originator. Mandatory fields are rendered **bold**.

validFor	A `TroubleTicketRequestValidFor`. Start and end date and time for the appointment.
category	A string. 'On-Site'.

@type	A string. ‘Appointment’

TroubleTicketRequestValidFor fields

The startDateTime and endDateTime have to be aligned with the time slots as defined by S/PRI, ESS or WITA.
All Parameters are filled by the originator. Mandatory fields are rendered **bold**.

startDateTime	A date time. Start date and time of the end-customer appointment.
endDateTime	A date time. End date and time of the end-customer appointment.

TestCall fields

In any clearing scenario where a Testcall is required, e.g. routing issues, this structure shall be filled completely. All parameters are filled by the originator.
Mandatory fields are rendered **bold**.

source	A string. The source phone number of the test call - fully qualified e.g. 49711233434. Used to find log entries
destination	A string. The destination phone number of the test call - fully qualified e.g. 49711233434. Used to find log entries
timestamp	A date time. Timestamp when placing the testcall. Including time zone.
statusCode	A string. The status code provided by the switch

ExternalIdentifier fields

The parameter external Identifier is used to transport an identifier like a reference to an order (NEU, KUE-AG, LAE, …) to a prenegotiation or anything else along with the clearing ticket. It is used to identify the action that should be cleared. All parameters are filled by the originator.

For each clearing case, specific external Identifiers are defined as mandatory or optional. Refer to 8.2 Understanding Clearing Metadata to see a list of expected and optional external Identifiers.

Mandatory fields are rendered **bold**.

externalIdentifierType	A string. The type of the external identifier. Allowed values: • `externalOrderId`: *externalOrderId* used for the order to be cleared. • `lineId`: *lineId* of the order to be cleared. • `homeId`: *homeId* used to identify a fiber socket.

	• `prenegotiationId`: *prenegotiationId* of the order to be cleared (Vorabstimmungsld). • `prenegotiationChangeId`: In case the prenegotiation was changed the id used for the change (prenegotiationId für the cancelation or rescheduling). • `externalOrderIdServiceApi:` External order number PreOrder-ID • `externalOrderIdTroubleTicketApi:` External ID used to send the order to ESS. • `technicalLineIdentifier:` Line designation (LSZ). Technical line identification assigned by the network owner to identify the physical line. • `bnetzaId:` Case number (BNetzA) • `ONT:` Optical Network Termination (ONT); installation identifier.
id	A string. The external Identifier.
@type	A string. 'ExternalIdentifier'.

ClearingDataError fields

Is used to carry the original error message in question e.g. the originator has sent an order via S/PRI to the processor and this was rejected with the error code "*text 1101 : Produkt an diesem Anschlusstyp nicht möglich*". Then he would place these error code & text into this fields in the Clearing Ticket.

All parameters are filled by the originator. Mandatory fields are rendered **bold**.

code	A string. Message code provided by the API to be cleared caused this clearing (Meldungscode).
text	A string Message text provided by the API to be cleared that caused this clearing (Meldungscode).

Json representation sample

We provide below the json representation of an example of a 'ClearingTicket' resource object.

Note: This is just a complete example showing all attributes with some value. The combination of the values in this sample might NOT comply to the rules for scenarios.


```
{
    "id": "0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "creationDate": "2022-05-19T12:07:35.882Z",
    "lastUpdate": "2022-05-19T12:07:35.882Z",
    "description": "2.1.03 Ausbleibende RUEM-VA im Anbieterwechsel",
    "originator": "DEU.CAR1",
```

```
"processor": "DEU.CAR2",
"externalId": "DEU.CAR1.4711",
"severity": "regular",
"severityChangeReason": "reason for severity change",
"requestedResolutionDate": "2022-05-22",
"ticketType": "1.03",
"resolveAttachment": [
    {
        "id": "11c9b5a6-a1a3-46f8-b1c1-eb87ce621f16",
        "name": "document.pdf",
        "role": "PROOF",
        "mimeType": "application/pdf",
        "href": "/attachment/11c9b5a6-a1a3-46f8-b1c1-eb87ce621f16",
        "size": 1234567
    }
],
"resolvedSuccessfully": true,
"status": {
    "changeDate": "2022-05-19T12:07:35.882Z",
    "changeReason": "change reason text",
    "status": "resolved"
},
"statusChange": [
    {
        "changeDate": "2022-05-19T12:07:35.882Z",
        "changeReason": "change reason text",
        "status": "status",
        "resolvedSuccessfully": false
    }
],
"note": [
    {
        "author": "DEU.CAR1",
        "date": "2022-05-19T12:07:35.882Z",
        "text": "This is a note"
    }
],
"clearingData": {
    "contractNumber": {
        "externalIdentifierType": "witaContractNumber",
        "id": "3054621927",
        "@type": "ExternalIdentifier"
    },
    "externalIdentifiers": [
        {
            "externalIdentifierType": "externalOrderId",
            "id": "qwerftghjui87654",
            "@type": "ExternalIdentifier"
        },
        {
            "externalIdentifierType": "lineID",
            "id": "DEU.CARRIER.123ABC7H",
            "@type": "ExternalIdentifier"
        },
        {
```

```
            "externalIdentifierType": "prenegotiationId",
            "id": "DEU.ITUC.V123456789",
            "@type": "ExternalIdentifier"
        },
        {
            "externalIdentifierType": "prenegotiationChangeId",
            "id": "DEU.ITUC.T123456790",
            "@type": "ExternalIdentifier"
        },
        {
            "externalIdentifierType": "externeOrderIdServiceApi",
            "id": "a83d03e4-7c20-4a47-b2e3-453f2c33dfaa",
            "@type": "ExternalIdentifier"
        },
        {
            "externalIdentifierType": "externalOrderIdTroubleTicketApi",
            "id": "0815abc2711",
            "@type": "ExternalIdentifier"
        },
        {
            "externalIdentifierType": "bnetzaId",
            "id": "47110815",
            "@type": "ExternalIdentifier"
        }
    ],
    "ndc": "228",
    "phone": [
        {
            "ndc": "228",
            "sn": "9752000",
            "blockPrefix": "181",
            "blockStart": "00000",
            "blockEnd": "99999"
        }
    ],
    "customer": {
        "individual": {
            "givenName": "Peter",
            "familyName": "Müller",
            "@type": "Individual"
        },
        "organization": {
            "tradingName": "Vereinigte Stahlwerke Hintertupfingen GmbH&Co KG",
            "@type": "Organization"
        }
    },
    "address": {
        "streetName": "Hauptstrasse",
        "streetNr": "47",
        "streetNrSuffix": "a",
        "postcode": "59423",
        "city": "Irgendwo",
        "locality": "Gartenvorstadt",
        "@type": "GeographicalAddress"
    },
```

```
        "requestedDate": "2022-05-19",
        "confirmedDate": "2022-05-19",
        "agreedSwitchDate": "2022-05-19",
        "portingDate": "2022-05-19",
        "prolongationDate": "2022-05-19",
        "prolongationCompleted": "2022-06-01",
        "initialSwitchDate": "2022-05-19",
        "troubleTicketToBeClearedCreationDate": "2022-05-19",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.FIRMA",
        "ekpAbg": "DEU.CARRIE",
        "nga": "string",
        "additionalInformation": "Please call for more information: 030 1122333",
        "troubleTicketRequest": {
            "validFor": {
                "startDateTime": "2022-05-19T08:00:00.000Z",
                "EndDateTime": "2022-05-19T12:00:00.000Z"
            },
            "category": "On-Site",
            "@type": "Appointment"
        },
        "error": {
            "code": "2007",
            "text": "WAL Anteil liegt vor."
        },
        "testCall": {
            "source": "05241234567",
            "destination": "03011223344",
            "timestamp": "2022-05-19T12:07:35.882Z",
            "statusCode": "301"
        },
        "specialAgreements": "#Migration Sylt#",
        "attachment": [
            {
                "id": "898a5801-e928-4587-b8f2-ae6320e8698b",
                "name": "document.pdf",
                "role": "PROOF",
                "mimeType": "application/pdf",
                "href": "/attachment/898a5801-e928-4587-b8f2-ae6320e8698b",
                "size": 1234567
            },
            {
                "id": "f84549d3-1f2f-4f0d-b22c-9a473d2cec3f",
                "name": "document2.pdf",
                "role": "OTHER",
                "mimeType": "application/pdf",
                "href": "/attachment/f84549d3-1f2f-4f0d-b22c-9a473d2cec3f",
                "size": 1234567
            }
        ]
    },
    "href": "/troubleTicket/0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
```

```
}
```

4.2 ClearingTicket Operations

4.2.1 Retrieve a (single) clearing ticket

```
GET /troubleTicket/{id}
```

Description

This operation retrieves a clearing ticket entity.

HTTP status codes:

200: OK	The ticket with id has been found
404: Not Found	Ticket with id has not been found.

Usage Samples

Here's an example of a request for retrieving a single clearing ticket.

```
GET /partner-api/v1/troubleTicket/0d33ee3e-7061-4a62-9ca6-57557d801c93
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "id": "0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "creationDate": "2022-05-19T12:07:35.882Z",
    "lastUpdate": "2022-05-19T12:07:35.882Z",
    "description": "2.1.03 Ausbleibende RUEM-VA im Anbieterwechsel",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.03",
    "status": {
        "changeDate": "2022-05-19T12:07:35.882Z",
        "status": "acknowledged"
    },
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
```

```
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```


4.2.2 List clearing tickets

`GET /troubleTicket?{filters}&{pagination}`

Description

This operation lists clearing ticket entities.

Filters

The following `filters` are available:

- `ticketType` (= scenario id): filter tickets with this `ticketType.` Supports a pattern, e.g., "1.*" selects all tickets with `ticketType` "1.a — 1.z".
- `creationDateFrom`, `creationDateTo`: only date, no time, filters tickets with `creationDate` creationDateFrom <= creationDate <= creationDateTo
- `lastUpdateFrom`, `lastUpdateTo`: date and time, filters tickets with `lastUpdate` lastUpdateFrom <= lastUpdate <= lastUpdateTo

- `originator`: filters tickets with this `originator`
- `processor`: filters tickets with this `processor`
- `externalId`: filters tickets with this `externalId`
- `requestedResolutionDateFrom, requestedResolutionDateTo`: filters tickets with `requestedResolutionDateFrom <= requestedResolutionDate <= requestedResolutionDateTo`
- `status`: filters tickets with this `status (StatusType)`
- `severity`: filters tickets with this `severity`

Pagination

The following optional parameters support pagination:

- `limit`: return only `limit` tickets even if more tickets would match. The default is “no limit”
- `offset`: skip the first `offset` tickets of the set of matching tickets. The default is “0”

HTTP status codes:

`200: OK`	All clearing tickets matching the criteria have been returned.
`206: Partial content`	Only a part of these tickets could be returned.

Note on partial content

In this case 2 header fields describe the situation:

`X-Total-Count` (integer): Total number of clearing tickets matching the criteria.

`X-Result-Count` (integer): Actual number of clearing tickets returned in the response body.

Note: Even without specifying a limit there might be a partial content result as the clearing platform may decide to set a limit itself, the default limit is 1000.

Usage Samples

Here's an example of a simple request for retrieving a list of clearing ticket(s).

```
GET /partner-api/v1/troubleTicket?ticketType=1.*&originator= DEU.CAR1
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[
    {
        "id": "0d33ee3e-7061-4a62-9ca6-57557d801c93",
        "creationDate": "2022-05-19T12:07:35.882Z",
        "lastUpdate": "2022-05-19T12:07:35.882Z",
        "description": "2.1.03 Ausbleibende RUEM-VA im Anbieterwechsel",
        "originator": "DEU.CAR1",
        "processor": "DEU.CAR2",
        "externalId": "DEU.CAR1.4711",
        "severity": "regular",
        "requestedResolutionDate": "2022-05-23",
        "ticketType": "1.03",
        "status": {
            "changeDate": "2022-05-19T12:07:35.882Z",
            "status": "acknowledged"
        },
        "clearingData": {
            "externalIdentifiers": [
                {
                    "externalIdentifierType": "prenegotiationId",
                    "id": "DEU.ITUC.V123456789",
                    "@type": "ExternalIdentifier"
                }
            ],
            "phone": [
                {
                    "ndc": "228",
```

```
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
},
{
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-02T08:17:21.564Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
      Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    "status": {
        "changeDate": "2022-05-02T08:17:21.564Z",
        "changeReason": "ticket in progress",
        "status": "inProgress"
    },
    "statusChange": [
        {
            "changeDate": "2022-05-01T12:07:35.882Z",
            "status": "acknowledged"
        }
    ],
    "clearingData": {
        "externalIdentifiers": [
            {
```

```
                    "externalIdentifierType": "prenegotiationId",
                    "id": "DEU.ITUC.V123456789",
                    "@type": "ExternalIdentifier"
                }
            ],
            "phone": [
                {
                    "ndc": "228",
                    "sn": "9752000"
                }
            ],
            "customer": {
                "individual": {
                    "givenName": "Peter",
                    "familyName": "Müller",
                    "@type": "Individual"
                }
            },
            "address": {
                "streetName": "Hauptstrasse",
                "streetNr": "47",
                "streetNrSuffix": "a",
                "postcode": "59423",
                "city": "Irgendwo",
                "locality": "Gartenvorstadt",
                "@type": "GeographicalAddress"
            },
            "requestedDate": "2022-05-16",
            "pkiAuf": "D123",
            "pkiAbg": "D001",
            "ekpAuf": "DEU.CAR2",
            "ekpAbg": "DEU.CAR1"
        },
        "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
        "@type": "ClearingTicket",
        "@baseType": "TroubleTicket"
    }
]
```


This more complex example illustrated the usage of queries and limit for retrieving a list of clearing ticket(s).:

Request

```
GET /partner-api/v1/troubleTicket?ticketType=1.*&originator= DEU.CAR1&limit=1
Accept: application/json
```

Response

```
HTTP/1.1 206 Partial Content
Content-Type: application/json
X-Total-Count: 2
X-Result-Count: 1

[
```

```
{
    "id": "0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "creationDate": "2022-05-19T12:07:35.882Z",
    "lastUpdate": "2022-05-19T12:07:35.882Z",
    "description": "2.1.03 Ausbleibende RUEM-VA im Anbieterwechsel",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.03",
    "status": {
        "changeDate": "2022-05-19T12:07:35.882Z",
        "status": "acknowledged"
    },
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/0d33ee3e-7061-4a62-9ca6-57557d801c93",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
},
```

```
    {
        "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
        "creationDate": "2022-05-01T12:07:35.882Z",
        "lastUpdate": "2022-05-02T08:17:21.564Z",
        "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
        "originator": "DEU.CAR1",
        "processor": "DEU.CAR2",
        "externalId": "DEU.CAR1.4711",
        "severity": "regular",
        "requestedResolutionDate": "2022-05-23",
        "ticketType": "1.04",
        "status": {
            "changeDate": "2022-05-02T08:17:21.564Z",
            "changeReason": "ticket in progress",
            "status": "inProgress"
        }
]
```


4.2.3 Create a clearing ticket

`POST /troubleTicket`

Description

This operation creates a clearing ticket entity.

HTTP status codes:

`201: Created`	Valid ticket create request.
`422: Unprocessable Content`	**validation error.** The `Error` object will contain the reason(s). **Example**: mandatory input missing.
`400: Bad Request`	**format (syntax) error.** There will be **no** `Error` object. **Example**: Bad json format.

For each successfully created clearing ticket, there will be a notification `TroubleTicketCreateEvent` sent.

Note:

Only valid clearing tickets are allowed. In case of any error **no** ticket will be created.

The validation check includes both compliance with all rules from the metadata and the specific rules in Chapter 8.1 Special Rules, as well as verification of the originator, processor and their limitations regarding the supported scenarios as reported via 6.2.3 List of supported scenarios.

In addition, the platform checks if a created ticket is a duplication of an existing ticket. If a ticket with all of the following parameters already exist, the created ticket is considered a duplication:
Does a ticket already exist, that
- is not in final state
- has same originator
- has same processor

- has same clearing scenario
- has same "mandatory external identifiers".

When a ticket is considered a duplicate, a ticket note has to be added to the newly created ticket with the content:
"Duplicate
(existing ticket id) abcdefgh-1234-5678-90ab-abcdefghijkl"

Mandatory fields

The following list contains the mandatory fields when creating a clearing ticket:

- `description`: Description of the clearing scenario as defined in the list of clearing cases in AH4Clearing.
  Recommendation: Value SHOULD be set to asterisk *, then the clearing platform will fill this attribute with the clearing scenario description according to the given `ticketType` (= clearing scenario id). Otherwise, the clearing platform will not touch this attribute.
- `severity`
- `ticketType`
- `originator` The platform will check, if the sender is entitled to act as this **originator** (Itu-Code)
- `processor`
- `externalId`

There might be more mandatory fields which are related to the chosen clearing scenario. This information is provided in 0.
Important Note: Missing fields attributes will cause a **validation error**.

Protected Attributes

Some attributes must not be specified when creating a clearing ticket. The following table provides a list of these protected attributes.
Note: If one of these attributes is accidentally set during ticket creation there will be no error but the attribute will be silently reset or overwritten.

Important Note: There might be more protected attributes which are related to the chosen clearing scenario. This information is provided in 0. Setting one of these protected attributes will cause a **validation error**.

protected Attributes	rule
Id	The ticket id will be generated by the clearing platform
Status	The status will be set to `acknowledged` by the clearing platform
statusChange[]	The statusChange history will be filled by the clearing platform with each “ticket status change” or “ticket resolved” request
Note	Notes are not allowed during creation

resolutionDate	This will be set by the clearing platform along with every state change `inProgress → resolved` and must stay empty during creation
resolvedSuccessfully	This will be set along with every status change `inProgress → resolved` and must stay empty during creation
resolvedAttachment[]	This list might be filled along with every status change `inProgress → resolved` and must stay empty during creation
creationDate	The ticket creationDate will be set by the clearing platform
lastUpdate	The ticket lastUpdate will be updated by the clearing platform with each change request

Usage Sample – Create Request

```
POST /partner-api/v1/troubleTicket
Content-Type: application/json

{
    "description": "*",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.03",
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
```

```
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```

Response

```
HTTP/1.1 201 CREATED
Content-Type: application/json

{
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-02T08:17:21.564Z",
    "description": "2.1.03 Ausbleibende RUEM-VA im Anbieterwechsel",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.03",
    "status": {
        "changeDate": "2022-05-19T12:07:35.882Z",
        "status": "acknowledged"
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
```

```
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```


4.2.4 Set clearing ticket status

`PATCH/troubleTicket/{id}/status`

Description

This operation sets a new status for a clearing ticket entity. It expects a body of type `StatusChange`.

Note: This operation will only allow valid status transitions. See description in 3 Lifecycle.

HTTP status codes:

`200: OK`	Valid status transition request.
`422: Unprocessable Content`	**validation error.** The `Error` object will contain the reason(s). **Example**: transition not allowed.
`400: Bad Request`	**format (syntax) error.** There will be **no** `Error` object. **Example**: Bad json format.
`404: Not Found`	Ticket with id has not been found.

There will be a notification `TroubleTicketStatusChangeEvent` sent for each successful status change.

Important: For the status transition `inProgress → resolved` the special operation `PATCH troubleTicket/resolved` must be used.

Mandatory fields

The following list contains the mandatory fields when setting a ticket status:

- status: The status to be set

Protected Attributes

Some attributes of `StatusChange` must not be specified when changing a status. The following table provides a list of these protected attributes.

Note: If one of these attributes is accidentally set there will be no error but the attribute will be silently reset or overwritten.

protected Attributes	rule
changeDate	The changeDate will be set by the clearing platform

Request

```
PATCH /partner-api/v1/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36/status
Content-Type: application/json

{
    "changeReason": "ticket in progress",
    "status": "inProgress"
}
```

Response

```
 HTTP/1.1 200 OK

Content-Type: application/json

{
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-02T08:17:21.564Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    "status": {
        "changeDate": "2022-05-02T08:17:21.564Z",
        "changeReason": "ticket in progress",
        "status": "inProgress"
    },
```

```
    "statusChange": [
        {
            "changeDate": "2022-05-01T12:07:35.882Z",
            "status": "acknowledged"
        }
    ],
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```


4.2.5 Set clearing ticket resolved

```
PATCH /troubleTicket/{id}/resolved
```

Description

This operation sets the resolved status for a clearing ticket entity. It expects a body of type `StatusChangeResolved`.

Note: This operation will only allow valid status transitions (here only allowed from status `inProgress`). See description of the clearing ticket in chapter 3 Lifecycle.

HTTP status codes:

`200: OK`	Valid resolve request.
`422: Unprocessable Content`	**validation error.** The `Error` object will contain the reason(s). **Example**: transition not allowed.
`400: Bad Request`	**format (syntax) error.** There will be **no** `Error` object. **Example**: Bad json format.
`404: Not Found`	Ticket with id has not been found.

There will be a notification `TroubleTicketResolvedEvent` sent for this status change.

Important Note: The payload of this operation is of type `StatusChangeResolved`.This structure consists of all attributes of `StatusChange` plus `resolvedSuccessfully` and `resolveAttachment[].` Both additional attributes are not stored within `StatusChange` but within the `ClearingTicket` resource itself. This means that in the rare case of multiple `resolved` operations there is no history for these attributes, but only the latest state is available. This is consistent with the handling of `ClearingData` which also has no history, but also always reflects the latest change.

Mandatory fields

The following list contains the mandatory fields when setting a ticket status to resolved:

- **status**: The status to be set (here: resolved)
- **resolvedSuccessfully**: The indicator if the clearing ticket was successfully solved as requested (true) or false if any additional action is required by the originator.
- **ResolveAttachment**: **Some** clearing cases expect an attachment as a mandatory part of the resolution. See 0 for more details.

Protected Attributes

Some attributes of `StatusChangeResolved` must not be specified when changing a status. The following table provides a list of these protected attributes.
Note: If one of these attributes is accidentally set there will be no error but the attribute will be silently reset or overwritten.

protected attributes	rule
changeDate	The changeDate will be set by the clearing platform

Request

```
PATCH /partner-api/v1/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36/resolved
Content-Type: application/json

{
    "changeReason": "ticket resolved",
    "status": "resolved",
    "resolvedSuccessfully": true,
    "resolveAttachment": [
        {
            "id": "b5678ddd-b29f-4520-bca6-f32f65306444",
            "name": "document.pdf",
            "role": "PROOF",
            "mimeType": "application/pdf",
            "href": "/attachment/b5678ddd-b29f-4520-bca6-f32f65306444",
            "size": 1234567
        }
    ]
}
```

Response

```
 HTTP/1.1 200 OK

Content-Type: application/json

{
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-03T10:01:22.465Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    "resolutionDate": "2022-05-19T12:07:35.882Z",
    "resolvedSuccessfully": true,
    "resolveAttachment": [
        {
            "id": "b5678ddd-b29f-4520-bca6-f32f65306444",
            "name": "document.pdf",
            "role": "PROOF",
            "mimeType": "applcation/pdf",
            "href": "/attachment/b5678ddd-b29f-4520-bca6-f32f65306444",
            "size": 1234567
        }
    ],
    "status": {
        "changeDate": "2022-05-03T10:01:22.465Z",
        "changeReason": "ticket resolved",
```

```
        "status": "resolved"
    },
    "statusChange": [
        {
            "changeDate": "2022-05-02T08:17:21.564Z",
            "changeReason": "ticket in progress",
            "status": "inProgress"
        },
        {
            "changeDate": "2022-05-19T12:07:35.882Z",
            "status": "acknowledged"
        }
    ],
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```

4.2.6 Set clearing ticket clearing data

`PATCH /troubleTicket/{id}/clearingData`

Description

This operation updates the clearing data for a clearing ticket entity. It expects a body of type `ClearingData`. In principle, all data of a ticket can be changed, but it is not possible to change the processor or the clearing scenario.

Note: This operation is only allowed in status `pending` = `information needed` and might only be initiated by the originator of a clearing ticket.

HTTP status codes:

`200: OK`	Valid data change request.
`422: Unprocessable Content`	**validation error.** The `Error` object will contain the reason(s). **Example**: mandatory input missing.
`400: Bad Request`	**format (syntax) error.** There will be **no** `Error` object. **Example**: Bad json format.
`404: Not Found`	Ticket with id has not been found.

A data change notification is sent for this data change.

After the originator has updated the clearing data the state of the ticket has to be set to `inProgress` by the originator to allow the processor working on the clearing ticket again.

Mandatory and protected fields

Only fields within structure `clearingData` might be changed. So, in principle, all data of a ticket can be changed, but it is not possible e.g. to change the processor or the clearing scenario.

Other than that, the same rules apply as for **creating** a clearing ticket.

Again, validation errors will end up in a http return code
`400: Bad Request` or `422: Unprocessable Content`
and for the latter the `Error` object will contain the reason for the rejection.

Request

```
PATCH partner-api/v1/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36/clearingData
Content-Type: application/json

{
    "externalIdentifiers": [
        {
            "externalIdentifierType": "prenegotiationId",
            "id": "DEU.ITUC.V123456789",
            "@type": "ExternalIdentifier"
        }
    ],
    "phone": [
        {
```

```
            "ndc": "228",
            "sn": "9752000"
        }
    ],
    "customer": {
        "individual": {
            "givenName": "Petra",
            "familyName": "Maier",
            "@type": "Individual"
        }
    },
    "address": {
        "streetName": "Nebenstrasse",
        "streetNr": "56",
        "postcode": "59423",
        "city": "Irgendwo",
        "locality": "Gartenvorstadt",
        "@type": "GeographicalAddress"
    },
    "requestedDate": "2022-05-16",
    "pkiAuf": "D123",
    "pkiAbg": "D001",
    "ekpAuf": "DEU.CAR2",
    "ekpAbg": "DEU.CAR1"
}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-04T13:45:51.665Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    "status": {
        "changeDate": "2022-05-03T10:01:22.465Z",
        "changeReason": "information needed",
        "status": "pending"
    },
    "statusChange": [
        {
            "changeDate": "2022-05-02T08:17:21.564Z",
            "changeReason": "ticket in progress",
            "status": "inProgress"
        },
        {
```

```
            "changeDate": "2022-05-19T12:07:35.882Z",
            "status": "acknowledged"
        }
    ],
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Petra",
                "familyName": "Maier",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Nebenstrasse",
            "streetNr": "56",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```


4.2.7 Set clearing ticket severity

`PATCH/troubleTicket/{id}/severity`

Description

This operation sets a new severity for a clearing ticket entity. It expects a body of type `Severity`.

Note: This operation may only be initiated by the originator of a clearing ticket.

HTTP status codes:

`200: OK`	Valid severity change request.
`422: Unprocessable Content`	**validation error.** The `Error` object will contain the reason(s). **Example**: transition not allowed.
`400: Bad Request`	**format (syntax) error.** There will be **no** `Error` object. **Example**: Bad json format.
`404: Not Found`	Ticket with id has not been found.

There will be a notification `TroubleTicketSeverityChangeEvent` sent for each severity change.

Mandatory fields

The following list contains the mandatory fields when setting a new severity:

severity: The new severity to be set

See also: 8.1.1 Rules for severity

Request

```
PATCH/partner-api/v1/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36/severity
Content-Type: application/json

{
  "severity": "critical",
  "reason": "the reason for the severity change",

}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-02T08:17:21.564Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "critical",
    "severityChangeReason": "the reason for the severity change",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    "status": {
        "changeDate": "2022-05-02T08:17:21.564Z",
        "changeReason": "ticket in progress",
        "status": "inProgress"
    },
    "statusChange": [
        {
```

```
            "changeDate": "2022-05-01T12:07:35.882Z",
            "status": "acknowledged"
        }
    ],
    "clearingData": {
        "externalIdentifiers": [
            {
                "externalIdentifierType": "prenegotiationId",
                "id": "DEU.ITUC.V123456789",
                "@type": "ExternalIdentifier"
            }
        ],
        "phone": [
            {
                "ndc": "228",
                "sn": "9752000"
            }
        ],
        "customer": {
            "individual": {
                "givenName": "Peter",
                "familyName": "Müller",
                "@type": "Individual"
            }
        },
        "address": {
            "streetName": "Hauptstrasse",
            "streetNr": "47",
            "streetNrSuffix": "a",
            "postcode": "59423",
            "city": "Irgendwo",
            "locality": "Gartenvorstadt",
            "@type": "GeographicalAddress"
        },
        "requestedDate": "2022-05-16",
        "pkiAuf": "D123",
        "pkiAbg": "D001",
        "ekpAuf": "DEU.CAR2",
        "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
}
```

4.2.8 Add a note to an existing clearing ticket

`POST /troubleTicket/{id}/note`

Description

This operation adds a new note to a clearing ticket entity. It expects a body of type `Note`.

Note: This operation may be processed by either originator or processor of a clearing ticket while the clearing ticket is in an active state.

HTTP status codes:

`201: Created`	Valid add note request.
`422: Unprocessable Content`	**validation error.** The `Error` object will contain the reason(s). **Example**: transition not allowed.
`400: Bad Request`	**format (syntax) error.** There will be **no** `Error` object. **Example**: Bad json format.
`404: Not Found`	Ticket with id has not been found.

There will be a notification `TroubleTicketAddNoteEvent` sent for each added note.

Mandatory fields

The following list contains the mandatory fields when adding a new note:

- text: The message to be sent.

Protected Attributes

Some attributes of `Note` must not be specified when creating a note. The following table provides a list of these protected attributes.
Note: If one of these attributes is accidentally set there will be no error but the attribute will be silently reset or overwritten.

protected Attributes	rule
author	The ticket `author` will be set by the clearing platform to the ITU carrier code of the sender or the id of the clearing platform (for system messages). If the sending party is willing to reveal information about the **real author** (name, department, email address, ...) this should go into the **text** attribute.
date	The `date` will be set by the clearing platform

Request

```
POST /partner-api/v1/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36/note
Content-Type: application/json

{
    "text": "a note"
}
```

Response

```
HTTP/1.1 201 CREATED
Content-Type: application/json

{
        "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
        "creationDate": "2022-05-01T12:07:35.882Z",
        "lastUpdate": "2022-05-02T08:17:21.564Z",
        "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
        "originator": "DEU.CAR1",
        "processor": "DEU.CAR2",
        "externalId": "DEU.CAR1.4711",
        "severity": "regular",
        "requestedResolutionDate": "2022-05-23",
        "ticketType": "1.04",
        "status": {
            "changeDate": "2022-05-02T08:17:21.564Z",
            "changeReason": "ticket in progress",
            "status": "inProgress"
        },
        "statusChange": [
            {
                "changeDate": "2022-05-01T12:07:35.882Z",
                "status": "acknowledged"
            }
        ],
        "note": [
            {
                "author": "DEU.CAR1",
                "date": "2022-05-02T08:17:21.564Z",
                "text": "a note"
            }
        ],
        "clearingData": {
            "externalIdentifiers": [
                {
                    "externalIdentifierType": "prenegotiationId",
                    "id": "DEU.ITUC.V123456789",
                    "@type": "ExternalIdentifier"
                }
            ],
            "phone": [
                {
                    "ndc": "228",
                    "sn": "9752000"
```

```
                }
            ],
            "customer": {
                "individual": {
                    "givenName": "Peter",
                    "familyName": "Müller",
                    "@type": "Individual"
                }
            },
            "address": {
                "streetName": "Hauptstrasse",
                "streetNr": "47",
                "streetNrSuffix": "a",
                "postcode": "59423",
                "city": "Irgendwo",
                "locality": "Gartenvorstadt",
                "@type": "GeographicalAddress"
            },
            "requestedDate": "2022-05-16",
            "pkiAuf": "D123",
            "pkiAbg": "D001",
            "ekpAuf": "DEU.CAR2",
            "ekpAbg": "DEU.CAR1"
        },
        "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
        "@type": "ClearingTicket",
        "@baseType": "TroubleTicket"
    }
}
```

5 Attachment

Attachments are files that need to be attached to a ticket.

See also 8.1.6 Rules for Attachments.

The API supports attachments

within `ClearingTicket.ClearingData.attachment[]` when **creating** a new ticket or **changing** a ticket in status `pending`

within `StatusChangeResolved.resolvedAttachment[]` when resolving a ticket (status `resolved`)

How to add an attachment

The process of adding attachments to a ticket is like follows:

1. Post the attachment to the clearing platform and add the resulting `Attachment` resource to the list of attachments
2. When done: Post your *ticket create* or your *status resolved*

Pseudocode:

Add attachment to new ticket

```
ticket = new Ticket

/* add all ticket data */
ticket.addData

/* prepare attachment */
file = path + myAttachment.pdf

attachment = client.post("/attachment?filename=myAttachment.pdf")
          .contentType("application/pdf")
          .body(file)

/* add attachment to ticket*/
ticket.clearingDate.attachment.add(attachment)

/* create ticket */
attachment = client.post("/troubleTicket ")
          .contentType("application/json")
          .body(ticket)
```

Add attachment to status resolved

```
/* prepare status resolved */
resolveStatus = new StatusChangeResolved
resolveStatus.status = StatusType.RESOLVED
resolveStatus.changeReason = “OK”
resolveStatus.resolvedSuccessfully = true
```

```
/* prepare attachment for status resolved */
file = path + myAttachment.pdf
attachment = client.post("/attachment?filename=myAttachment.pdf")
          .contentType("application/pdf")
          .body(file)

/* add attachment to resolveStatus */
resolveStatus.resolvedAttachment.add(attachment)

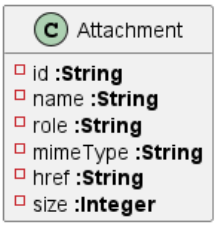
/* send resolve status */
client.patch("/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36/resolved")
          .contentType("application/json")
          .body(resolveStatus)
```

5.1 Attachment Resource Model

An attachment, content not embedded but referenced by URL.

Note: See chapter 5.2 Attachment Operations to learn how to create an attachment.

Attachment (Clearing Platform API/ 1.0.0)



Field descriptions

An Attachment can be created by any partner any time.

It may be attached to a ticket by the *originator* only at creation or if the ticket is in the pending state.

The *processor* may add attachments when resolving the ticket.
Mandator fields are rendered **bold**.

id	A string. The unique id of the attachment.
name	A string. The (file)name of the attachment. The extension of the file describes its type. Example: `mydocument.txt` has the file type `txt`. While most of the file types are supported, some file types are forbidden.

	See: _8.1.6.2 Unsupported file types: Attachments Blacklist
role	A string. The type (or role) of the attachment. Here the list of supported types: • ADDRESS_CHANGE: change of address German Adressänderung • EXCERPT_PROVE: proof of excerpt German: Auszugs-Nachweis • PROOF: proof German: Nachweis • PHONE_NUMBER_LIST: list of phone numbers German: Rufnummernliste • HALF_YEARLY_REPORT: half year phone number list (HJM) German: HJM-Liste • BNETZA_DOCUMENT: Document of the BNetzA (or Federal Network Agency) German: Dokument der Bundesnetzagentur • BUILDER_SERVICE_INVOICE: Invoice of the builder service German: Rechnung des Bauherren-Service • OTHER: any other German: Andere
mimeType	A string. The mime type of the attachment such as application/pdf.
href	A string. Hyperlink, a reference to the attachment as relative URL.
size	A number. The size of the attachment in bytes.

Json representation sample

```
{
    "id": "30db2965-b676-47e2-9e25-b1342889e51e",
    "name": "document.pdf",
    "role": "PROOF",
    "mimeType": "application/pdf",
    "href": "/attachment/30db2965-b676-47e2-9e25-b1342889e51e",
    "size": 123456
}
```

5.2 Attachment Operations

5.2.1 Create Attachment

`POST /attachment/?filename={filename}`

Description

This operation supports creating a new attachment.

The attachment content (file) is provided as a binary stream (body).

The `Content-type` header must specify the real nature of the file (mime type).

The parameter `filename` can be used to pass the (url-encoded) file name.

HTTP status codes:

`201: Created`	The http return code of this operation for valid create attachment request.

The result of this operation is an object of type `Attachment`. It contains the following information: `id`, `href`, `mimeType`, `size` and (if given) `name`. In order to use this Attachment object within a clearing ticket, the missing attribute `role` should be added.

Usage Samples

Here's an example of a request for creating a new Attachment resource.

Request

```
POST /partner-api/v1/attachment?filename=document.pdf
Content-Type: application/octet-stream
{ --- binary data for pdf document --- }
```

Response

```
Content-Type: application/json

HTTP/1.1 201 CREATED
Content-Type: application/json

{
    "id": "30db2965-b676-47e2-9e25-b1342889e51e",
    "name": "document.pdf",
    "mimeType": "application/pdf",
    "href": "/attachment/30db2965-b676-47e2-9e25-b1342889e51e",
    "size": 123456
}
```

5.2.2 Retrieve Attachment

`GET/attachment/{id}`

Description

This operation retrieves an attachment content (file) as a binary stream.

HTTP status codes:

`200: OK`	Attachment content (file) with id has been found
`404: Not Found`	Attachment has not been not found.

Note: While the `create` operation returns an Attachment object for convenience, this `get` operation just returns the plain file. All other information is stored in the Attachment object within the clearing ticket and must be taken from there.

Usage Samples

Here's a sample of a request for retrieving an Attachment resource based on its id.

Request

```
GET /partner-api/v1/attachment/30db2965-b676-47e2-9e25-b1342889e51e
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream

{ --- binary data for document --- }
```

5.2.3 Delete Attachment

```
DELETE/attachment/{id}
```

Description

This operation deletes an object of type `Attachment` as well as the actual binary file. An attachment may only be deleted if it has not yet been linked to a ticket.

HTTP status codes:

`204: No Content`	Attachment and binary file has been deleted
`404: Not Found`	Attachment has not been not found.
`422: Unprocessable Content`	The attachment is already linked to a ticket and can not be deleted.

Note: Chapter 8.1.6.1 describes attachments required for distinct scenarios. These rules must not be circumvented by this operations.

Usage Samples

Here's a sample of a request for deleting an Attachment resource based on its id.

Request

```
DELETE /partner-api/v1/attachment/30db2965-b676-47e2-9e25-b1342889e51e
```

Response

```
HTTP/1.1 204 No Content
```

6 ITU Carrier

6.1 ITU Carrier Resource Model

Organization (Clearing Platform API/ 1.0.0)



Field descriptions

id	A string. The unique fully qualified identifier of the ItuCarrier given by the International Telecommunication Union (DEU.XXXXXX).
tradingName	A string. The name that the organization (unit) trades under.
organizationType	A string. The type of this organization: 'ItuCarrier'.
unsupportedScenarios	List of unsupported scenarios (ids). To get a list of supported scenarios, use method `GET /organization/unsupportedScenarios`
href	Hyperlink, a reference to the ItuCarrier Organization entity as relative URL.
@type	The type of this entity: 'Organization'.

Json representation sample

```
{
    "id": "DEU.XXXXXX",
    "tradingName": "a fancy name for DEU.XXXXXX",
    "organizationType": "ItuCarrier",
    "href": "/organization/DEU.XXXXXX ",
    "@type": "Organization"
}
```

6.2 ITU Carrier Operations

6.2.1 List ItuCarrier

`GET/organization`

Description

This operation lists organizations = ItuCarrier entities.

HTTP status codes:

200: OK	List of organizations has been returned.

Hint: The content of this list of carriers connected to the clearing platform(s) will not expire over the course of a working day. You should consider caching this information rather than retrieving it over and over again.

Usage Samples

Here's an example of a request for retrieving organization resources.

Request

```
GET /partner-api/v1/organization

Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[
    {
        "id": "DEU.CAR1",
        "tradingName": "a fancy name for DEU.CAR1",
        "organizationType": "ItuCarrier",
        "href": "/organization/DEU.CAR1",
        "@type": "Organization"
    },
    {
        "id": "DEU.CAR2",
        "tradingName": "a fancy name for DEU.CAR2",
        "organizationType": "ItuCarrier",
        "unsupportedScenarios": [],
        "href": "/organization/DEU.CAR2",
        "@type": "Organization"
    },
    ...
]
```

6.2.2 Retrieve ItuCarrier

`GET/organization/{id}`

Description

This operation retrieves an organization = ItuCarrier entity.

HTTP status codes:

`200: OK`	The organization with id has been found
`404: Not Found`	The organization with id has **not** been found

Usage Samples

Here's a sample of a request for retrieving an organization resource based on its id.

Request

```
GET /serverRoot/partner-api/v1/organization/DEU.CAR1
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "id": "DEU.CAR1",
    "tradingName": "a fancy name for DEU.CAR1",
    "organizationType": "ItuCarrier",
    "unsupportedScenarios": [],
    "href": "/organization/DEU.CAR1",
    "@type": "Organization"
}
```

6.2.3 List of supported scenarios

`GET /organization/supportedScenarios/{id}`

Description

This operation retrieves the organization's list of supported scenarios.

Important Note:
The WebGUI of the Clearing platform will not allow creation of tickets with scenarios not supported by the processor. Also, any attempt to create such a ticket via REST Api will be rejected.
See 4.2.3 Create a clearing ticket.

HTTP status codes:

`200: OK`	The organization with id has been found
`404: Not Found`	The organization with id has **not** been found

Usage Samples

Here's a sample of a request for retrieving an organization's list of supported scenarios based on its id.

In this example only scenarios 1.01, 1.02, 1.03 are supported.

Request

```
GET /serverRoot/partner-api/v1/ organization/supportedScenarios/DEU.CAR1
Accept: application/json
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json

[
    "1.01", "1.02", "1.03"
]
```

7 Notifications

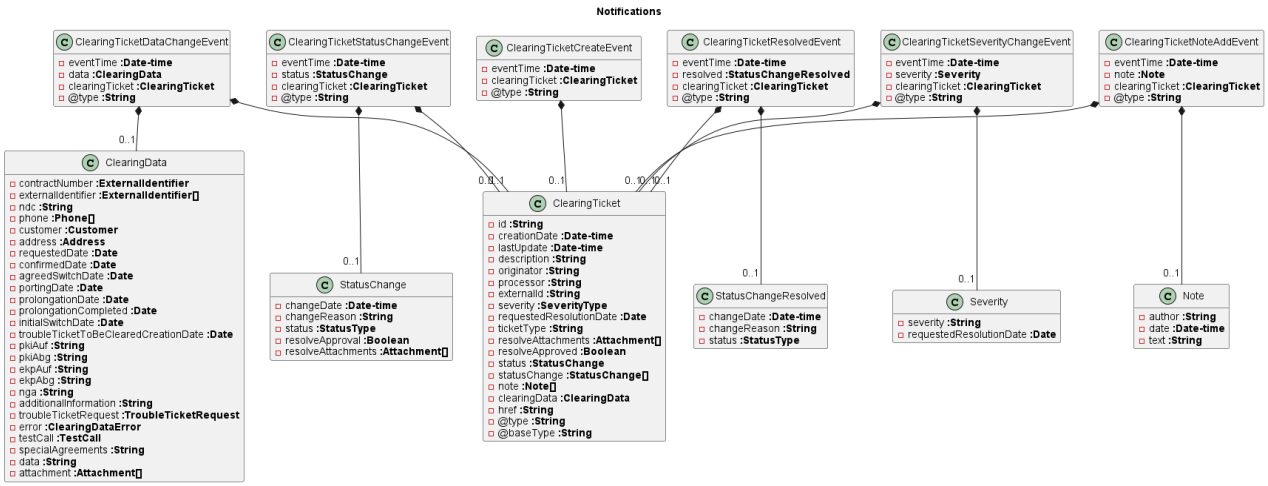
The Clearing Platform will notify their clients about new clearing tickets and changes of a clearing ticket (status, severity and notes).

A client can listen to the following events

- `TroubleTicketCreateEvent`
- `TroubleTicketDataChangeEvent`
- `TroubleTicketStatusChangeEvent`
- `TroubleTicketResolvedEvent`
- `TroubleTicketSeverityChangeEvent`
- `TroubleTicketNoteAddEvent`

7.1 Notification Resource Models

Following resources are defined for this API.



Field descriptions

eventTime	The date and time (DateTime) the event occurred.
clearingTicket	The clearing ticket (`ClearingTicket`) that has been created or modified.
data	The clearing data (`ClearingData`) that has been modified.
status	For `ClearingTicketStatusChangeEvent`: The change of status (`StatusChange`) that has been processed.
resolved	For `ClearingTicketResolvedEvent`: The ticket has been resolved. The structure `StatusChangeResolved` contains all necessary information.
severity	For `ClearingTicketSeverityChangeEvent`: The change of severity (`Severity`) that has been processed

note	For `ClearingTicketNoteAddEvent`: The note (`Note`) that has been added
@type	The type of this entity: '`ClearingTicket*Event`' = type of payload per event.

For field descriptions of `ClearingTicket`, `ClearingData`, `StatusChange`, `StatusChangeResolved`, `Severity` and `Note` see chapter 4.1 Clearing Ticket Resource Model.

7.2 Notification API

The notification API is part of the Clearing Platform OpenAPI Definition (OpenApi schema) and is tagged with `NotificationListener.`

Important Note: The client (recipient of the following notifications) **must** always respond with http return code `200: OK,` whenever the notification has been **successfully received**.

All **other** return codes will lead to a new attempt to deliver this current notification and **no further notifications will get delivered** as long as the current notification gets bounced with a http return code other that `200: OK`.

Thus, any possible errors detected on the client side must be handled **asynchronously**, e.g. by reporting the problem to the platform provider.

7.2.1 Register listener / Unregister listener

The API methods `POST /hub` and `DELETE /hub/{id}` as specified in TMF-621 are **not supported**.

There is no need to register a listener at runtime.

The client registers the call-back URL for the listener during the **Clearing Platform onboarding process**.

In the following examples the call-back URL of the client is shown as `http://in.listener.com`.

7.2.2 Notify about ticket creation

`POST /listener/troubleTicketCreateEvent`

Description

This notification happens when a new trouble ticket with id is created.

The type of the payload is `ClearingTicketCreateEvent`.

The expected http return code of this operation is `200: OK`.

Usage Sample

Request

```
POST http://in.listener.com/listener/listener/troubleTicketCreateEvent
Content-Type: application/json
```


```
{
    "eventTime": "2022-05-19T12:07:35.912Z",
    "clearingTicket": {
        "id": "0d33ee3e-7061-4a62-9ca6-57557d801c93",
        "creationDate": "2022-05-19T12:07:35.882Z",
        "lastUpdate": "2022-05-19T12:07:35.882Z",
        "description": "2.1.03 Ausbleibende RUEM-VA im Anbieterwechsel",
        "originator": "DEU.CAR1",
        "processor": "DEU.CAR2",
        "externalId": "DEU.CAR1.4711",
        "severity": "regular",
        "requestedResolutionDate": "2022-05-23",
        "ticketType": "1.03",
        "status": {
            "changeDate": "2022-05-19T12:07:35.882Z",
            "status": "acknowledged"
        },
        "clearingData": {
            "externalIdentifiers": [
                {
                    "externalIdentifierType": "prenegotiationId",
                    "id": "DEU.ITUC.V123456789",
                    "@type": "ExternalIdentifier"
                }
            ],
            "phone": [
                {
                    "ndc": "228",
                    "sn": "9752000"
                }
            ],
            "customer": {
                "individual": {
                    "givenName": "Peter",
                    "familyName": "Müller",
                    "@type": "Individual"
                }
            },
            "address": {
                "streetName": "Hauptstrasse",
                "streetNr": "47",
                "streetNrSuffix": "a",
                "postcode": "59423",
                "city": "Irgendwo",
                "locality": "Gartenvorstadt",
                "@type": "GeographicalAddress"
            },
            "requestedDate": "2022-05-16",
```

```
            "pkiAuf": "D123",
            "pkiAbg": "D001",
            "ekpAuf": "DEU.CAR2",
            "ekpAbg": "DEU.CAR1"
        },
        "href": "/troubleTicket/0d33ee3e-7061-4a62-9ca6-57557d801c93",
        "@type": "ClearingTicket",
        "@baseType": "TroubleTicket"
    },
  "@type": "ClearingTicketCreateEvent"
}
```

Response

```
HTTP/1.1 200 OK
```

7.2.3 Notify about ticket data change

```
POST /listener/troubleTicketDataChangeEvent/{id}
```

Description

This notification happens whenever the `clearingData` of a trouble ticket with id has been changed. This might only happen in status `pending` and can happen more than once before the status is finally changed to `inProgress` (or any other valid status).

The type of the payload is `ClearingTicketDataChangeEvent`.

The expected http return code of this operation is `200: OK`.

Usage Sample

Request

```
POST http://in.listener.com/listener/troubleTicketDataChangeEvent
Content-Type: application/json

{
    "eventTime": "2022-05-02T08:17:21.774Z",
    "clearingData": {
        "externalIdentifiers": [
           {
              "externalIdentifierType": "prenegotiationId",
              "id": "DEU.ITUC.V123456789",
              "@type": "ExternalIdentifier"
           }
        ],
        "phone": [
           {
              "ndc": "228",
              "sn": "9752000"
           }
        ],
```

```
      "customer": {
         "individual": {
            "givenName": "Peter",
            "familyName": "Müller",
            "@type": "Individual"
       }
       },
       "address": {
          "streetName": "Hauptstrasse",
          "streetNr": "47",
          "streetNrSuffix": "a",
          "postcode": "59423",
          "city": "Irgendwo",
          "locality": "Gartenvorstadt",
          "@type": "GeographicalAddress"
       },
       "requestedDate": "2022-05-16",
       "pkiAuf": "D123",
       "pkiAbg": "D001",
      "ekpAuf": "DEU.CAR2",
      "ekpAbg": "DEU.CAR1"
  },
  "clearingTicket": {
      "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
      "creationDate": "2022-05-01T12:07:35.882Z",
      "lastUpdate": "2022-05-02T08:17:21.564Z",
      "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
        Anbieterwechsel, durch Rolle EKPauf",
      "originator": "DEU.CAR1",
      "processor": "DEU.CAR2",
      "externalId": "DEU.CAR1.4711",
      "severity": "regular",
      "requestedResolutionDate": "2022-05-23",
      "ticketType": "1.04",
      "status": {
          "changeDate": "2022-05-02T08:17:21.564Z",
          "changeReason": "ticket in progress",
          "status": "inProgress"
      },
      "statusChange": [
          {
              "changeDate": "2022-05-01T12:07:35.882Z",
              "status": "acknowledged"
          }
      ],
      "clearingData": {
          "externalIdentifiers": [
              {
                  "externalIdentifierType": "prenegotiationId",
                  "id": "DEU.ITUC.V123456789",
                  "@type": "ExternalIdentifier"
              }
          ],
          "phone": [
              {
```

```
                    "ndc": "228",
                    "sn": "9752000"
                }
            ],
            "customer": {
                "individual": {
                    "givenName": "Peter",
                    "familyName": "Müller",
                    "@type": "Individual"
                }
            },
            "address": {
                "streetName": "Hauptstrasse",
                "streetNr": "47",
                "streetNrSuffix": "a",
                "postcode": "59423",
                "city": "Irgendwo",
                "locality": "Gartenvorstadt",
                "@type": "GeographicalAddress"
            },
            "requestedDate": "2022-05-16",
            "pkiAuf": "D123",
            "pkiAbg": "D001",
            "ekpAuf": "DEU.CAR2",
            "ekpAbg": "DEU.CAR1"
        },
        "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
        "@type": "ClearingTicket",
        "@baseType": "TroubleTicket"
    },
  "@type": "ClearingTicketDataChangeEvent"
}
```


Response

```
HTTP/1.1 200 OK
```

7.2.4 Notify about ticket status change

```
POST /listener/troubleTicketStatusChangeEvent
```

Description

This notification happens whenever the status of a trouble ticket with id has changed.

The type of the payload is `ClearingTicketStatusChangeEvent`.

The expected http return code of this operation is `200: OK`.

Usage Sample

Request

```
POST http://in.listener.com/listener/troubleTicketStatusChangeEvent
Content-Type: application/json

{
    "eventTime": "2022-05-02T08:17:21.774Z",
    "statusChange": {
        "changeDate": "2022-05-19T12:07:35.882Z",
        "changeReason": "ticket in progress",
        "status": "inProgress"
    },
    "clearingTicket": {
        "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
        "creationDate": "2022-05-01T12:07:35.882Z",
        "lastUpdate": "2022-05-02T08:17:21.564Z",
        "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
        "originator": "DEU.CAR1",
        "processor": "DEU.CAR2",
        "externalId": "DEU.CAR1.4711",
        "severity": "regular",
        "requestedResolutionDate": "2022-05-23",
        "ticketType": "1.04",
        "status": {
            "changeDate": "2022-05-02T08:17:21.564Z",
            "changeReason": "ticket in progress",
            "status": "inProgress"
        },
        "statusChange": [
            {
                "changeDate": "2022-05-01T12:07:35.882Z",
                "status": "acknowledged"
            }
        ],
        "clearingData": {
            "externalIdentifiers": [
                {
                    "externalIdentifierType": "prenegotiationId",
                    "id": "DEU.ITUC.V123456789",
                    "@type": "ExternalIdentifier"
                }
            ],
            "phone": [
                {
                    "ndc": "228",
                    "sn": "9752000"
                }
            ],
            "customer": {
                "individual": {
                    "givenName": "Peter",
                    "familyName": "Müller",
                    "@type": "Individual"
                }
            },
            "address": {
```

```
                "streetName": "Hauptstrasse",
                "streetNr": "47",
                "streetNrSuffix": "a",
                "postcode": "59423",
                "city": "Irgendwo",
                "locality": "Gartenvorstadt",
                "@type": "GeographicalAddress"
            },
            "requestedDate": "2022-05-16",
            "pkiAuf": "D123",
            "pkiAbg": "D001",
            "ekpAuf": "DEU.CAR2",
            "ekpAbg": "DEU.CAR1"
        },
        "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
        "@type": "ClearingTicket",
        "@baseType": "TroubleTicket"
    },
  "@type": "ClearingTicketStatusChangeEvent"
}
```


Response

```
HTTP/1.1 200 OK
```

7.2.5 Notify about ticket resolved

```
POST /listener/troubleTicketResolvedEvent
```

Description

This notification happens whenever a trouble ticket with id gets resolved.

The type of the payload is `ClearingTicketResolvedEvent`.

The expected http return code of this operation is `200: OK`.

Usage Sample

Request

```
POST http://in.listener.com/troubleTicketResolvedEvent
Content-Type: application/json
```


```
{
  "eventTime": "2022-05-03T10:01:22.775Z",
  "statusChange": {
    "changeDate": "2022-05-19T12:07:35.882Z",
    "changeReason": "ticket resolved",
    "status": "resolved",
    "resolvedSuccessfully ": true,
    "resolveAttachment": [
      {
        "id": "b5678ddd-b29f-4520-bca6-f32f65306444",
        "name": "document.pdf",
```

```
        "role": "PROOF",
        "mimeType": "applcation/pdf",
        "href": "/attachment/b5678ddd-b29f-4520-bca6-f32f65306444",
        "size": 1234567
      }
    ]
  },
  "clearingTicket": {
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-03T10:01:22.465Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "regular",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    “resolutionDate": "2022-05-03T10:01:22.465Z"
    "resolvedSuccessfully ": true,
    "resolveAttachment": [
      {
        "id": "b5678ddd-b29f-4520-bca6-f32f65306444",
        "name": "document.pdf",
        "role": "PROOF",
        "mimeType": "applcation/pdf",
        "href": "/attachment/b5678ddd-b29f-4520-bca6-f32f65306444",
        "size": 1234567
      }
    ],
    "status": {
      "changeDate": "2022-05-03T10:01:22.465Z",
      "changeReason": "ticket resolved",
      "status": "resolved"
    },
    "statusChange": [
      {
        "changeDate": "2022-05-02T08:17:21.564Z",
        "changeReason": "ticket in progress",
        "status": "inProgress"
      },
      {
        "changeDate": "2022-05-19T12:07:35.882Z",
        "status": "acknowledged"
      }
    ],
    "clearingData": {
      "externalIdentifiers": [
        {
          "externalIdentifierType": "prenegotiationId",
          "id": "DEU.ITUC.V123456789",
          "@type": "ExternalIdentifier"
        }
      ],
```

```
      "phone": [
        {
          "ndc": "228",
          "sn": "9752000"
        }
      ],
      "customer": {
        "individual": {
          "givenName": "Peter",
          "familyName": "Müller",
          "@type": "Individual"
        }
      },
      "address": {
        "streetName": "Hauptstrasse",
        "streetNr": "47",
        "streetNrSuffix": "a",
        "postcode": "59423",
        "city": "Irgendwo",
        "locality": "Gartenvorstadt",
        "@type": "GeographicalAddress"
      },
      "requestedDate": "2022-05-16",
      "pkiAuf": "D123",
      "pkiAbg": "D001",
      "ekpAuf": "DEU.CAR2",
      "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
  },
  "@type": "ClearingTicketResolvedEvent"
}
```


Response

```
HTTP/1.1 200 OK
```

7.2.6 Notify about ticket severity change

```
POST /listener/troubleTicketSeverityChangeEvent
```

Description

This notification happens whenever the severity of a trouble ticket with id has changed.

The type of the payload is `ClearingTicketSeverityChangeEvent`.

The expected http return code of this operation is `200: OK`.

Usage Sample

Request

```
POST http://in.listener.com/troubleTicketSeverityChangeEvent
Content-Type: application/json

{
  "eventTime": "2022-05-02T08:17:21.774Z",
  "statusChange": {
    "severity": "date"
  },
  "clearingTicket": {
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-02T08:17:21.564Z",
    "description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
          Anbieterwechsel, durch Rolle EKPauf",
    "originator": "DEU.CAR1",
    "processor": "DEU.CAR2",
    "externalId": "DEU.CAR1.4711",
    "severity": "critical",
    "requestedResolutionDate": "2022-05-23",
    "ticketType": "1.04",
    "status": {
      "changeDate": "2022-05-02T08:17:21.564Z",
      "changeReason": "ticket in progress",
      "status": "inProgress"
    },
    "statusChange": [
      {
        "changeDate": "2022-05-01T12:07:35.882Z",
        "status": "acknowledged"
      }
    ],
    "clearingData": {
      "externalIdentifiers": [
        {
          "externalIdentifierType": "prenegotiationId",
          "id": "DEU.ITUC.V123456789",
          "@type": "ExternalIdentifier"
        }
      ],
      "phone": [
        {
          "ndc": "228",
          "sn": "9752000"
        }
      ],
      "customer": {
        "individual": {
          "givenName": "Peter",
          "familyName": "Müller",
          "@type": "Individual"
        }
      },
      "address": {
        "streetName": "Hauptstrasse",
        "streetNr": "47",
```

```
        "streetNrSuffix": "a",
        "postcode": "59423",
        "city": "Irgendwo",
        "locality": "Gartenvorstadt",
        "@type": "GeographicalAddress"
      },
      "requestedDate": "2022-05-16",
      "pkiAuf": "D123",
      "pkiAbg": "D001",
      "ekpAuf": "DEU.CAR2",
      "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
  },
  "@type": "ClearingTicketSeverityChangeEvent"
}
```


Response

```
HTTP/1.1 200 OK
```

7.2.7 Notify about new ticket note

```
POST /listener/troubleTickeNoteAddEvent
```

Description

This notification happens whenever a note has been added to a trouble ticket with id.

The type of the payload is `ClearingTicketNoteAddEvent`.

The expected http return code of this operation is `200: OK`.

Usage Sample

Request

```
POST http://in.listener.com/listener/troubleTickeNoteAddEvent
Content-Type: application/json
```


```
{
  "eventTime": "2022-05-02T08:17:21.774Z",
  "note": {
    "author": "DEU.CAR1",
    "date": "2022-05-02T08:17:21.564Z",
    "text": "a note"
  },
  "clearingTicket": {
    "id": "14225687-2a2f-44f3-a18f-becf06f2ac36",
    "creationDate": "2022-05-01T12:07:35.882Z",
    "lastUpdate": "2022-05-02T08:17:21.564Z",
```

```
"description": "2.1.04 Ausbleibende Antwort auf TVS-VA/STR-AEN/STR-AUF im
      Anbieterwechsel, durch Rolle EKPauf",
"originator": "DEU.CAR1",
"processor": "DEU.CAR2",
"externalId": "DEU.CAR1.4711",
"severity": "regular",
"requestedResolutionDate": "2022-05-23",
"ticketType": "1.04",
"status": {
  "changeDate": "2022-05-02T08:17:21.564Z",
  "changeReason": "ticket in progress",
  "status": "inProgress"
},
"statusChange": [
  {
    "changeDate": "2022-05-01T12:07:35.882Z",
    "status": "acknowledged"
  }
],
"note": [
  {
    "author": "DEU.CAR1",
    "date": "2022-05-02T08:17:21.564Z",
    "text": "a note"
  }
],
"clearingData": {
  "externalIdentifiers": [
    {
      "externalIdentifierType": "prenegotiationId",
      "id": "DEU.ITUC.V123456789",
      "@type": "ExternalIdentifier"
    }
  ],
  "phone": [
    {
      "ndc": "228",
      "sn": "9752000"
    }
  ],
  "customer": {
    "individual": {
      "givenName": "Peter",
      "familyName": "Müller",
      "@type": "Individual"
    }
  },
  "address": {
    "streetName": "Hauptstrasse",
    "streetNr": "47",
    "streetNrSuffix": "a",
    "postcode": "59423",
    "city": "Irgendwo",
    "locality": "Gartenvorstadt",
    "@type": "GeographicalAddress"
```

```
      },
      "requestedDate": "2022-05-16",
      "pkiAuf": "D123",
      "pkiAbg": "D001",
      "ekpAuf": "DEU.CAR2",
      "ekpAbg": "DEU.CAR1"
    },
    "href": "/troubleTicket/14225687-2a2f-44f3-a18f-becf06f2ac36",
    "@type": "ClearingTicket",
    "@baseType": "TroubleTicket"
  },
  "@type": "ClearingTicketNoteAddEvent"
}
```


Response

```
HTTP/1.1 200 OK
```

8 Appendix

8.1 Special Rules

8.1.1 Rules for severity

The severity of a clearing ticket depends on the specific case.

For each clearing ticket the originator has to set a severity – mandatory field.

The severity can be one of these values:

Severity	meaning
`regular`	Regular severity as defined in the Arbeitshandbuch4Clearing for this specific clearing case. This is the default severity.
`critical`	The service for the customer is impacted. The clearing ticket shall be solved as soon as possible at least within one working day. If set to `critical` a specific reason must be provided. If the ticket is critical due to a Bundesnetzagentur escalation then the `bnetzaid` shall be provided.
`escalated`	The severity may not be escalated before the time to handle the ticket is used up or the ticket severity has already been set to `critical`.

	It is **not** intended to automatically escalate all overdue tickets to reduce the risk of overloading the processor. It is **not** possible to start a ticket with an escalated severity. If set to `escalated` a specific reason shall be provided. If changed from `critical` to `escalated` the already given reason might stay untouched.

The severity can only be set by the originator and only when creating a ticket or while a ticket is active: status `acknowledged`, `inProgress`, or `held`. This means that a severity cannot be changed when the active party is the originator (e.g. `pending`) or the ticket is no longer active (e.g. `cancelled`).

See also: *8.1.3* Rules for requestedResolutionDate

Note: While typically the severity is changed from `regular` to something more severe one can also reduce the severity back to e.g. `regular` if there is no longer a reason for a higher severity.

It is also possible to keep a `severity` while changing the `requestedResolutionDate`. This might be useful if one would gain more time for processing for good reasons e.g. ticket has been in `pending` state for a while. It is possible to change the `requestedResolutionDate` into the past, but it is not possible to break the rules for the 8.1.3 Rules for requestedResolutionDate.

8.1.2 Rules for date and time

If date and time is transmitted the regular format must be used:

```
"changeDate": "2022-05-19T12:07:35.882Z",
```

which includes a time zone.

If no time zone is present, UTC is assumed.

8.1.3 Rules for requestedResolutionDate

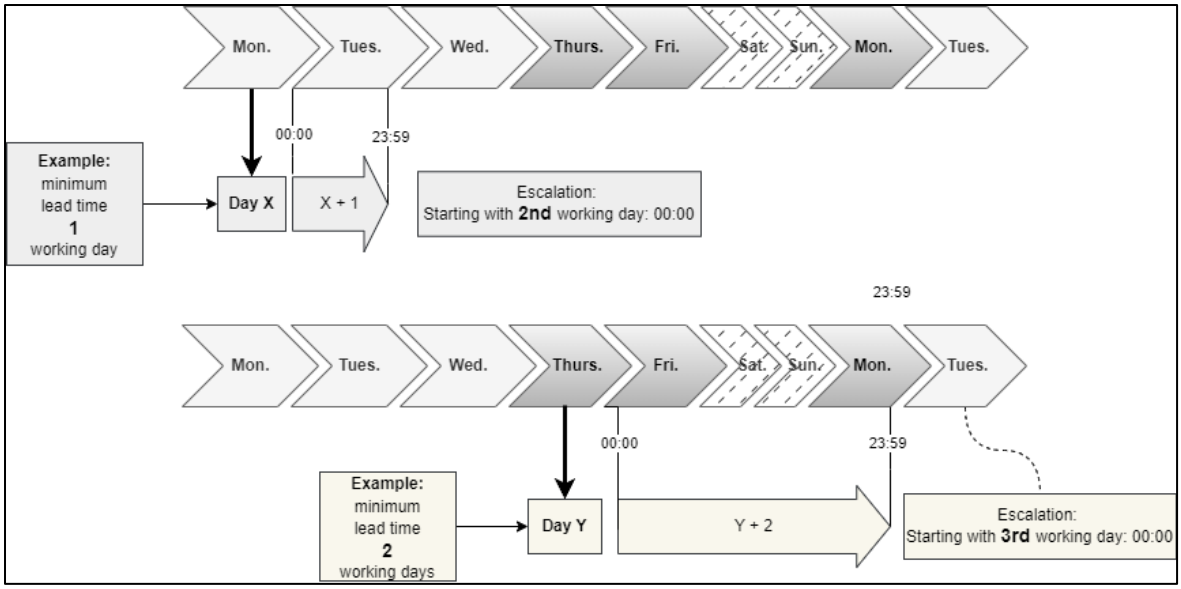
Date until the processor shall handle this clearing case.

This date might be after but never before specified in the AH4Clearing.

There, the regular time (number of days) for processing a request is defined for most clearing cases. This time is considered as working days:

- working days do not include weekends (Saturday and Sunday) and public holidays.
- calculation starts with the ticket creation time.

- the processor has time to resolve the issue until end of the day specified as this `requestedResolutionDate`.



If the `requestedResolutionDate` is not given by the originator, or if it is not at least as long as the minimum lead time as defined for this clearing scenario, it will be set to this minimum lead time by the platform.

Exception for severity critical:

- For a newly created ticket with severity critical, the requestedResolutionDate may be set by the originator to one working day after ticket creation, even if the scenario specific minimum lead time is longer.
- If the severity of an active ticket is changed from regular to critical, the requestedResolutionDate may remain unchanged. If it is changed by the originator, the earliest allowed requestedResolutionDate is one working day after the severity change, even if the scenario specific minimum lead time is longer. A later requestedResolutionDate remains allowed.

See also: *8.1.1* Rules for severity.

8.1.4 ResolvedSuccessfully

To reduce the workload of all participating partners, there is a new mandatory parameter that must be set by the processor when resolving the clearing ticket.

The `resolvedSuccessfully` parameter is intended to help the originator to close the ticket automatically without further analysis.

If `resolvedSuccessfully` is set to `true`, the processor has solved the issue in the requested way. If set to `false` the issue may also be solved, but the originator shall take another look to verify if the solution requires some more manual invention.

In either case, the processor **can** (if `false`: **MUST**) provide additional information in the `changeReason` field. If necessary, also with resolve attachments.

8.1.5 Rules for prolongationChange

Whenever a prolongation (German: Weiterversorgung) is modified or cancelled the background for the change must be provided so the processor can change his systems accordingly.

This is done by selecting one of the following values per scenario:

2.3.08 Change of Prolongation (German: Änderung der Weiterversorgung)

Value	Reason	Background
`RESCHEDULED`	New date German: Neuer Termin	The switch **will be** processed on new date provided as: `clearingData.prolongationDate` German: Vorziehung oder Verschiebung: Datumseingabe über Attribut: `clearingData.prolongationDate`
`PKI_CHANGE`	change of PKI German: Neuer PKI	The voice service provider has changed. The new PKI is provided as: `additional Information.` German: Bitte Angabe des neuen PKI in Attribut `additional Information.`
`OUTDATED`	Resource no longer needed German: Änderung Ressourcenmodus	The physical resource shall no longer be transferred. The old provider shall turn off the resource at the end of the prolongation. More information might be given as: `additional Information` German: Bitte Angabe, ob die Änderung mit oder ohne neue Vorabstimmung erfolgt, in Attribut: `additional Information`
`OTHER`	Any other German: Sonstiges	Please explain in: `additional Information` German: Freitext: Weitere Angaben in Attribut `additional Information`

2.3.09 Cancellation of Prolongation (German: Aufhebung der Verlängerung)

Value	Cancellation reason	Background
`CUSTOMER_CANCELLATION`	Cancelled by end customer German: Widerruf des Kunden	The end customer cancelled the new order and stays with the previous provider. More information might be given as: `additional Information` German: Abbruch des Anbieterwechsels aufgrund Kundenwiderruf. KEINE neue Vorabstimmung. Weitere Angaben in Attribut `additional Information`
`ORDER_FAILED`	Order failed German: Abbruch mit erneuter Vorabstimmung durch EKPauf	There are technical issues. So, the new provider will create a new order. More information might be given as: `additional Information` German: Anbieterwechsel-Auftrag fehlgeschlagen aufgrund technischer Probleme: Weitere Angaben in Attribut `additional Information`
`CARRIER_CANCELLATION`	Technically not possible German: Abbruch des Anbieterwechsels durch EKPauf	There are technical issues. So, the new provider cannot supply service to the customer. No new order will be created. More information might be given as: `additional Information` German: Abbruch des Anbieterwechsels durch EKPauf. Abbruch des Anbieterwechsels aufgrund technischer Probleme von EKPauf. KEINE neue Vorabstimmung.: Weitere Angaben in Attribut `additional Information`
`OTHER`	Any other German: Sonstiges	Please explain in: `additional Information`

2.3.11 Continued supply before the switch day on provider change
(German: Weiterversorgung vor Wechseltag im Anbieterwechsel)

Value	Reason	Background
`RESCHEDULED`	New date German: Neuer Termin	The switch **will be** processed on new date provided as: `clearingData.prolongationDate`

`CUSTOMER_CANCELLATION`	Cancelled by end customer German: Widerruf des Kunden	The end customer cancelled the new order and stays with the previous provider. More information might be given as: `additional Information`
`ORDER_FAILED`	Order failed German: Abbruch mit erneuter Vorabstimmung durch EKPauf	There are technical issues. So, the new provider will create a new order. More information might be given as: `additional Information`
`CARRIER_CANCELLATION`	Technically not possible German: Abbruch des Anbieterwechsels durch EKPauf	There are technical issues. So, the new provider cannot supply service to the customer. No new order will be created. More information might be given as: `additional Information`
`OTHER`	Any other German: Sonstiges	Please explain in: `additional Information`

8.1.6 Rules for Attachments

- maximum size per attachment: 5 MB.
- maximum number of attachments per ticket: no limit
- Virus scan: Although taking reasonable precautions to ensure no viruses or malicious software are present in the stored attachments, the clearing platform cannot accept responsibility for any loss or damage arising from the use of attachments.

8.1.6.1 Attachment roles per scenario

Some scenarios expect at least one attachment with a special attachment role:

Scenario	Role
`5.4.01.list`	`HALF_YEARLY_REPORT`
All other scenarios ending with `-list` (except `5.4.01.list`)	`PHONE_NUMBER_LIST`
All **BnetzA** scenarios = main scenario `6.0`	`BNETZA_DOCUMENT`

8.1.6.2 Supported MIME types

The following MIME types are supported as attachments. (The common file types for the MIME types are displayed in parentheses)

It is highly recommended to validate “Magic Bytes” and scan for viruses in attachments, since only checking the MIME type or file type cannot ensure the expected file content!

- application/xml (.xml)
- application/pdf (.pdf)
- application/vnd.oasis.opendocument.spreadsheet (.ods)
- application/vnd.oasis.opendocument.text (.odt)
- application/vnd.openxmlformats-officedocument.spreadsheetml.sheet (.xslx)
- application/vnd.openxmlformats-officedocument.wordprocessingml.document (.docx)
- image/jpeg (.jpg)
- image/jpeg (.jpeg)
- image/png (.png)
- image/tiff (.tiff)
- message/rfc822 (.eml)
- text/csv (.csv)
- text/plain (.txt)

8.2 Understanding Clearing Metadata

The clearing metadata describe, which attributes have to be present depends on the clearing scenario. A list of all clearing scenarios and the respective presence rules can be found as separate documents in the following formats:

- **Clearing_Platform_Metadata.yaml:** a machine-readable document in yaml format
- **Clearing_Platform_Metadata.xlsx:** the same information as **Clearing_Platform_Metadata.yaml** rendered as Excel file for improved human readability

There is also a file **Clearing_Platform_Metadata_Schema.yaml** which describes the schema of Clearing_Platform_Metadata.yaml to validate the correctness of this file. We will use this file later to describe the content and the structure of **Clearing_Platform_Metadata.yaml.**

8.2.1.1 Clearing_Platform_Metadata.yaml.xlsx

When opening Clearing_Platform_Metadata.yaml.xlsx there are 5 different sheets.

1. Attributes: list of all attributes of a clearing ticket along with technical information like type, length etc.
2. Scenarios: list of all clearing scenarios with reference to their main scenario and additional attributes like default response deadline.
3. Main Scenarios: list of the main scenarios which are needed to group clearing scenarios e.g., in a GUI.
4. Configured Presence: list of all attributes and their presence (occurrence) per clearing scenario.

Important note: This list does not contain all possible attributes. For the sake of comprehensiveness only containers (OBJECTS) are listed. The **inner** attributes (members) of such containers are treated along with the container's presence. For detail presence rules see table **Presence Definition** down below.

5. **Effective Presence**: the same information as **Configured Presence** but flipped to see the attribute presence per scenario. Some parameters must be present in all clearing cases for the API to work e.g. originator and processor. The Effective Presence also includes these parameters into account and lists all required parameters while Configured Presence shows empty spots wherever there is a default value used.
   The default occurrence is part of the **Attributes** list.

8.2.1.2 Clearing_Platform_Metadata.yaml

This file is the source for runtime validation of a clearing ticket.

As mentioned before, **[REF-5] Clearing_Platform_Metadata_Schema.yaml** is the (meta) schema for this file.

It contains the following sections:

section	Description and sample

attributes:
list of attributes
	Attributes are the "fields" along with their properties.
	See: table **attribute properties**.
	
	Sample:
	`- name: id`
	`  label: Ticket ID`
	`  description: "Die Ticket-ID identifiziert ein Clearing Ticket eindeutig. Hierfür wird eine sogenannte GUID verwendet (global unique identifier) spezifiziert durch RFC 4122. Diese ID wird vom Server vergeben, an den die Anfragen versendet werden."`
	`  datatype: STRING`
	`  example: 550e8400-e29b-11d4-a716-446655440000`
	`  occurrence: 0..1`

mainScenarios: list of main scenarios	**Main scenarios** are logical container to group them by category The property mainScenarios is an object containing all main scenarios as named elements. Sample: `"1.0": 2.1 Vorabstimmung`
scenarioDef: list of scenarios	A **Scenario definition** • provides a specific list of attributes with their respective presence • global settings for this scenario

	The property scenarioDef is an object containing all scenarios as named elements.
	
	See: table **scenario properties**.
	
	Sample:
	`"1.01":`
	`  name: 2.1.01 Verhinderung Eigenkündigung im Anbieterwechsel`
	`  mainKey: 1.0`
	`  attributes:`
	`  clearingData.additionalInformation: 0..1`
	`  clearingData.address: 1`
	`  clearingData.attachment: 0..n`
	`  clearingData.customer: 1`
	`  clearingData.data: 0..1`
	`  clearingData.ekpAbg: 1`
	`  clearingData.ekpAuf: 1`
	`  clearingData.externalIdentifier: 0..2`
	`  clearingData.externalIdentifier[bnetzaId]: 0..1`
	`  clearingData.externalIdentifier[prenegotiationId]: 0..1`
	`  clearingData.phone: 0..1`
	`  clearingData.requestedDate: 1`
	`  clearingData.specialAgreements: 0..1`
	`  requestedResolutionDate: 0..1`
	`  responseDeadline: 1`

Attribute properties

parameter	type	Description
name	string	attribute name in JSON path format (fully qualified, starting with root ClearingTicket)
label	string	short name of the attribute, can be used as label e.g., in a GUI
description	string	long description of the attribute, can be used as help text e.g., in a GUI
keyAttribute	string	Addresses the attribute holding the type of an OBJECT or the virtual key of an ARRAY. As an ARRAY in this situation is always an ARRAY of OBJECT, the virtual key is similar to the type of an ARRAY element. This information helps to treat an ARRAY like a MAP

datatype	string	The technical type of the attribute: • **STRING**: a string • **DATE**: a date • **DATETIME**: a date and time information with time zone • **BOOLEAN**: true or false • **NUMBER**: a numeric string • **OBJECT**: container for other attributes • **ARRAY**: a list • **ARRAY_INDEX**: a special attribute of a list member which can be used to identify this entry (even if the list is not really a map). Such A list entry can be addressed by adding the value of this attribute to the list's name with in brackets, e.g., myList[myId] addresses an entry of list myList, where the attribute marked as ARRAY_INDEX has the value myId
minLength	number	minimum length for the value of an attribute if its value is not null
maxLength	number	maximum length for the value of an attribute
regexp	string	regular expression to check content
multiline	boolean	The indicator for a multi-line attribute (if the attribute may contain new line / line feeds). Default is false.
example	string or number	Example value for the attribute, can be used as input support e.g. in a GUI.
occurrence	string or number	The default occurrence of this attribute, if not explicitly specified within a scenario. For syntax, see **presence definition**.
mandatoryInStructure	boolean	The indicator for an attribute that is mandatory, as soon as the surrounding structure exists. Default is false. Along with `exclusiveGroup` this attribute might get the meaning of mayBeMandatory, as only one of the exclusive groups might be filled, hence `mandatoryInStructure` is also mutually exclusive.

exclusiveGroup	string	If an attribute (or a group of attributes) within a structure exclusively excludes another attribute (or groups of attributes) this is indicated by assigning it to a group.
		
		Example:
		In structure phone either sn or blockEnd, blockPrefix, blockStart excusively.
		So there are 2 groups for clearingData.phone:
		• phone-sn
		  ○ clearingData.phone.sn
		• phone-block
		  ○ clearingData.phone.blockEnd
		  ○ clearingData.phone.blockPrefix
		  ○ clearingData.phone.blockStart
		Similar for clearingData.customer:
		• customer-individual
		  ○ clearingData.customer.individual
		• customer-organization
		  ○ clearingData.customer.organization

setByPlatform	boolean	Indicator for an attribute that is always set by the platform, whether or not a value has been passed by the caller.

Scenario properties

parameter	type	Description
id	string or number	scenario id
name	string	scenario description
mainKey	string or number	id of the corresponding main scenario, e.g., 1.0
responseDeadline	integer	response deadline in days
fillWithOriginator	string	fill given attribute with originator id (clearingData.ekpAuf or clearingData.ekpAbg)

fillWithProcessor	string	fill given attribute with processor id (clearingData.ekpAuf or clearingData.ekpAbg)
attributes	object	the presence information per attribute. attributes is an object containing all presence information per attribute as `"attribute name" : occurrence` For syntax of occurrence, see presence definition.

Presence definition

value	description
Empty or 0	• FIELD: This field is not needed and must not be filled. • OBJECT: no inner attribute of the OBJECT must be filled • ARRAYS: must be empty.
1	• FIELD: This field is mandatory. It has to be present and must be filled. • OBJECT: any inner attribute of the OBJECT might be filled, but those marked as `mandatoryInStructure` must be filled • ARRAYS: must have exactly one entry.
0..1	• FIELD: This field is optional and it might be filled. • OBJECT: any inner attribute of the OBJECT might be filled, but those marked as `mandatoryInStructure` must be filled • ARRAYS: might be empty or have exactly one entry.
n..m	Only for ARRAYS: This array might have n-m entries. If n > 0, then at least these entries must exist.

9 Change Log

9.1 Version 1.0 to version 1.1

9.1.1 Lifecycle: Status initial and acknowledge

As soon as a new ticket (to be created) passes the platform validation it will be stored on the platform and get the status `acknowledged`.

Otherwise, the create request fails with response `400: Bad Request` or `422 = unprocessable content` and no ticket will be stored.

Details are in Chapter: 4.2.3 Create a clearing ticket and 3.2 States and Transitions.

9.1.2 List of attribute changes

- Added attribute `severityChangeReason`
- Removed `date` – from `SeverityType`
- Removed `severityChangeReason` – from `Note` attribute `role`'s description
- Added a list of supported `role`s to `clearingData.attachment` and `resolveAttachment`. See 5.1 Attachment Resource Model.
- Added `clearingData.externalIdentifier[ONT]`

9.1.3 Adjustments

- Clarification of naming: *ClearingTicket* vs. *TroubleTicket*, see 1.1.1 ClearingTicket
- New rule for checking an attachment file type against a blacklist (forbidden file types) rather than a whitelist (allowed file types) as done before. See: Unsupported file types: Attachments Blacklist
- All http status codes which do not have an explicit meaning for a certain REST method have been removed from the OpenAPI definition. See 1.3 HTTP Response Codes for details.
- New chapter *8* Appendix with sub chapters
    - 8.1 Special Rules:
        - Severity
        - Date and time
        - requestedResolutionDate
        - resolvedSuccessfully
        - prolongationChange
    - 8.1.6 Rules for Attachments
        - Clearing_Platform_Metadata.yaml.xlsx
        - Clearing_Platform_Metadata.yaml

9.2 Version 1.1 to version 1.2

- Chapter 3.1 Transition rules and 3.2 States and Transitions:
  - added new feature: **autoclose:**
    a resolved ticket will be automatically closed by the platform after 30 days of inactivity.
- Chapter 4.2.6 Set clearing ticket clearing data
  - adjusted: a notification **is sent** for data change
- Chapter 8.1.1 Rules for severity:
  - added clarification regarding changing the `requestedResolutionDate` into the past.
- Chapter 8.1.5 Rules for prolongationChange:
  - added some more German translations
  - added `setByPlatform`
  - added `fillWithProcessor`

9.2.1 Version 1.2 to version 1.2.1

- Return Codes:
  - 400 Bad Request: Clarification, that there will be **no** `Error` object.
- Attribute `externalId`: "***Unique***" has been removed
  - ~~*Unique*~~ *Identifier assigned by the originator for this clearing ticket.*
  - Clarification of the *presence* concept by enhancing the documentation for FIELD, OBJECT and ARRAY
- Chapter 4.2.6 Set clearing ticket clearing data
  - Clarification, which data might be changed:
    only data within structure `clearingData`
- Chapter 7.2 Notification API
  - Clarification, which http return codes are allowed: **only** `200: OK`
- Chapter 9.1.2 List of attribute changes
  - Here we added a new attribute `severityChangeReason,` but we missed to remove the description for the previous version using a Note with role 'severityChangeReason'. This has now been done.

The following has been added for adjusted Version 4.8.2025:

- New Chapter: 8.1.6 Rules for Attachments
  - Increase max size for an attachment 3 MB → 5 MB
  - Add a list of mandatory roles for specific scenarios: Attachment roles per scenario
  - Add new rules for mandatory status change comments for pending → inProgress and all status changes towards → cancelled
- Attribute `lastUpdate`
  - Value for the originator ticket and the corresponding processor ticket must always be identical.

- Modification of Transition -> finish
    - In either case, the processor can provide additional information in the changeReason field and may also include attachments. If resolvedSuccessfully is false, providing changeReason is mandatory

9.2.2 Version 1.2.1 to version 1.2.2

Introduction of unsupported scenarios
See Create a clearing ticket and List of supported scenarios

- Check of limitations regarding the supported scenarios
- New API for **supported** scenarios see List of supported scenarios
- List of **unsupported** scenarios now part of ITU Carrier Resource Model

9.2.3 Version 1.2.2 to version 1.3

- Chapter 8.1.3 Rules_for_requestedResolutionDate
    - added clarification for newly created tickets with severity critical
    - added clarification for severity changes from regular to critical
- Chapter 5.2.3 Delete_Attachment
    - added endpoint for attachment deletion. Attachments may only be deleted if they have not yet been linked to a ticket.
- Chapter 4.2.3 Create a clearing ticket
    - added duplicate validation