TM Forum Specification

Trouble Ticket API REST Specification

TMF621
Release 18.0.0
June 2018

Latest Update: TM Forum Release 18.0.0	Member Evaluation
Version 3.0.1	IPR Mode: RAND

NOTICE

TABLE OF CONTENTS

LIST OF TABLES

N/A

INTRODUCTION

The following document is the specification of the REST API for the trouble ticket resource. It includes the model definition as well as all available operations. Possible actions are creating and retrieving a trouble ticket, partially updating trouble ticket. Furthermore, the GET allows filtering using standard filter criteria.

The Trouble Ticket API provides a standardized client interface to Trouble Ticket Management Systems for creating, tracking and managing trouble tickets as a result of an issue or problem identified by a customer or another system. Examples of Trouble Ticket API originators (clients) include CRM applications, network management or fault management systems, or other Trouble Ticket management systems (e.g. B2B).

The API supports the ability to send requests to create a new trouble ticket specifying the nature and severity of the trouble or issue as well as all necessary related information. The API also includes mechanisms to search for and update existing trouble tickets. Notifications are defined to provide information when a trouble ticket has been updated, including status changes. A basic set of states of a trouble ticket has been specified (as an example) to handle trouble ticket lifecycle management.

Trouble Ticketing API manages trouble ticket resource:

- A trouble ticket represents a record, or an issue raised by requestor that need to be solved, used for reporting and managing the resolution of problems, incidents or request
- Main trouble ticket attributes are its description, severity, type, related dates (creation, expected resolution, resolution), state and related information (change reason and change date), related parties (originator, owner, reviser, etc.), related entities (product, product order, customer bill) and notes

Trouble Ticket API performs the following operations on trouble ticket

- Retrieval of a trouble ticket or a collection of trouble ticket depending on filter criteria
- Partial update of a trouble ticket
- Creation of a trouble ticket
- Notification of events on trouble ticket:
    - Trouble ticket state change
    - Trouble ticket change
    - Trouble ticket resoled
    - Trouble ticket created
    - Trouble ticket Information required

SAMPLE USE CASES

Reader will find examples of use cases using Trouble Ticket API in “Open Digital Business Scenarios and Use Cases” document

Use Case 1: User Raise Issue With Bill

- The user checks his last bill and he is not convinced that the charged amount is appropriate
- The user wants to dispute the bill and opens an issue (a commercial trouble ticket) to initiate the claim
- The new issue is added to the list of issues already open by the user.
- After some time, the user checks the status of the complaint in the app to understand if it has been resolved and the resolution details.

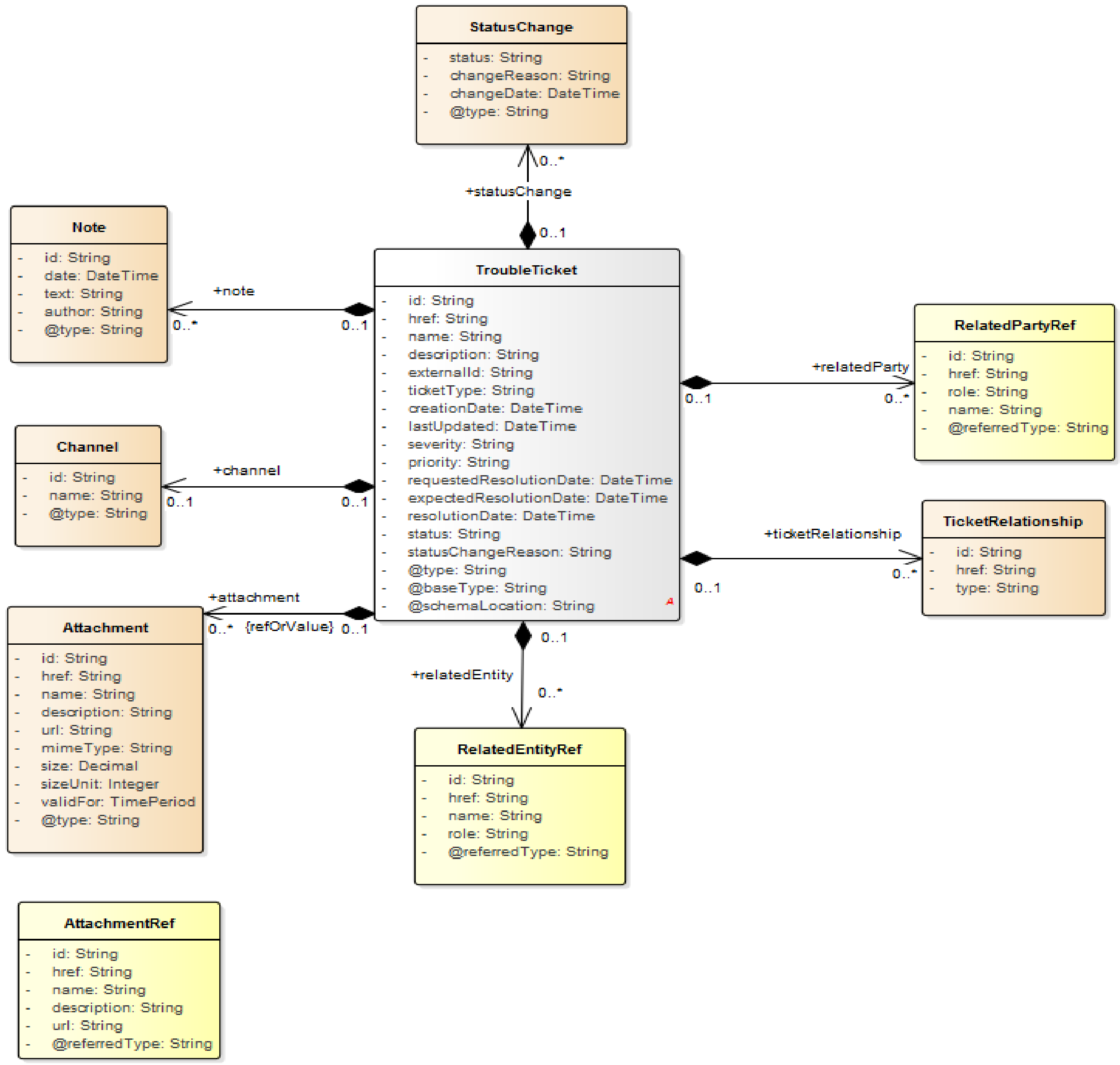
RESOURCE MODEL

Managed Entity and Task Resource Models

Trouble Ticket resource

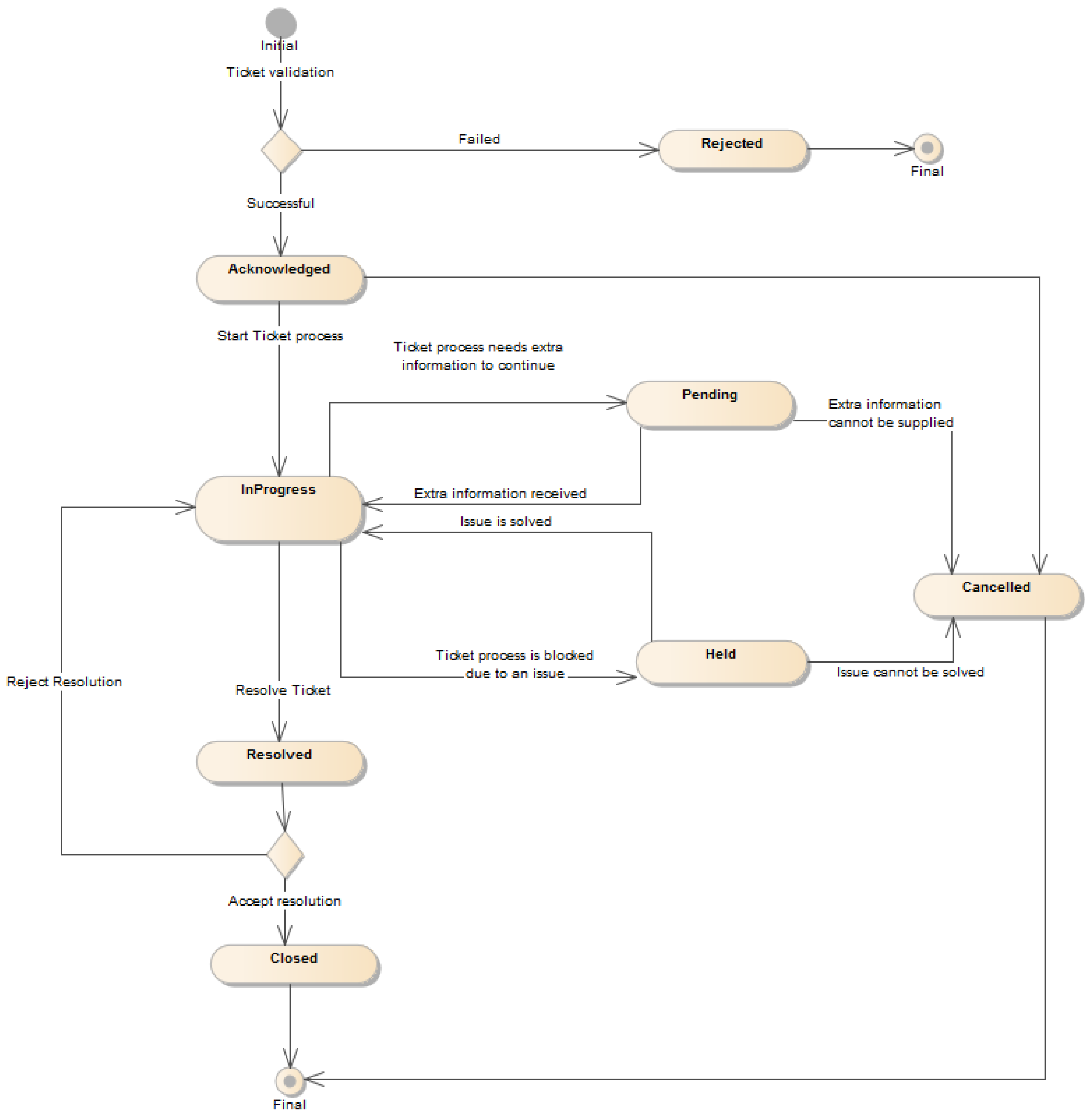
A trouble ticket is a record of an issue that is created, tracked, and managed by a Trouble Ticket Management system.

Resource model



Lifecycle

Note that an implementation of the specification may enrich the list of states depicted in the diagram. The state machine specifying the typical state change transitions is provided below.



Field descriptions

TroubleTicket fields

Id	A string. Unique identifier of the trouble ticket.
href	A string. Hyperlink, a reference to the trouble ticket entity.
Name	A string. Name of the trouble ticket, typically a short description provided by the user that create the ticket
externalId	A string. Additional identifier coming from an external system.
ticketType	A string. represent a business type of the trouble ticket e.g. incident, complain, request.
creationDate	A date time (DateTime). The date on which the trouble ticket was created.
lastUpdate	A date time (DateTime). The date and time that the ticked was last update.
description	A string. Description of the trouble.
severity	A string. The severity of the issue. Indicate the implication of the issue on the expected functionality e.g. of a system, application, service etc. Severity values can be for example: Critical, Major, Minor.
priority	A string. The priority of the trouble ticket and how quickly the issue should be resolved. Example: Critical, High, Medium, Low. The value is set by the trouble ticket management system considering the severity, ticketType etc...
requestedResolutionDate	A date time (DateTime). The resolution date requested by the user.
expectedResolutionDate	A date time (DateTime). The expected resolution date determined by the Trouble Ticket system.
resolutionDate	A date time (DateTime). The date and time the trouble ticket was resolved.
status	A string. The current status of the trouble ticket.
@baseType	A string. The base type (class) of the resource. Here can be 'TroubleTicket'.

@type	A string. The (class) type of the trouble ticket. e.g. BillingTicket, NetworkTicket, ResourceTicket.
@schemaLocation	A string. Link to the schema describing this REST resource.
relatedEntity	A list of related entity references (RelatedEntityRef [*]). An entity that is related to the trouble ticket such as a bill, a product, etc. The entity against which the trouble ticket is associated.
statusChange	A list of status changes (StatusChange [*]). The status change history that are associated to the trouble ticket. Populated by the server
statusChangeReson	The reason for changing the status
note	A list of notes (Note [*]). The note(s) that are associated to the trouble ticket.
relatedParty	A list of related party references (RelatedPartyRef [*]). The related party(ies) that are associated to the trouble ticket.
ticketRelationship	A list of ticket relationships (TicketRelationship [*]). A list of trouble ticket relationships (TroubleTicketRelationship [*]). Represents a relationship between trouble tickets.
channel	A channel (Channel). Channel reference. The channel defines the channel for selling product offerings.
attachment	A list of attachments (Attachment [*]). File(s) attached to the trouble ticket. e.g. picture of broken device, scanning of a bill or charge.

Attachment sub-resource

Complements the description of an element (for instance a product) through video, pictures...

description	A string. A narrative text describing the content of the attachment.
href	A string. Reference of the attachment.
id	A string. Unique identifier of the attachment.
url	A string. Uniform Resource Locator, is a web page address (a subset of URI).
mimeType	A string. The mime type of the document as defined in RFC 2045 and RFC 2046 specifications.
size	The size in Bytes of the of the document or attachment. If this component contains the embedded data then the size is the size of the embedded

	data; if it is a reference without the data then it is the size of the referenced document.
name	A string. The name of the file.
sizeUnit	An integer. The unit size for expressing the size of the file (MB,kB...).
validFor	A time period. Period of validity of the attachment.
@type	Indicates the (class) type of attachment.

AttachmentRef fields

id	A string. Unique identifier of the attachment
href	A string. URL serving as reference for the attachment.
name	A string. The name of the file
description	A string. A narrative text describing the content of the attachment.
url	A string. Uniform Resource Locator, is a web page address (a subset of URI).
@referredType	A string. Indicates the (class) type of the attachment

Channel sub-resource

Channel reference. The channel defines the channel for selling product offerings.

id	A string. Unique identifier of the channel.
name	A string. Name of the channel.
@type	A string. Indicates the (class) type of channel.

Note sub-resource

Extra information about a given entity.

Id	A String. Unique identifier of the note
date	A date time (DateTime). Date of the note.
author	A string. Author of the note.
text	A string. Text of the note.

@type	Indicates the (class) type of note

StatusChange sub-resource

Holds the status notification reasons and associated date the status changed. Populated by the server

status	A string. The status of the trouble ticket.
changeDate	A date time (DateTime). The date and time the status changed.
changeReason	A string. The reason why the status changed
@type	Indicates the (class) type of status change

TicketRelationship sub-resource

Represents a relationship between trouble tickets.

id	A string. Unique identifier of the related trouble ticket.
href	A string. Hyperlink, a reference to the related trouble ticket entity
type	A string. Type of the trouble ticket relationship can be isChiled, dependent etc...

RelatedEntityRef relationship

Related Entity reference. Reference to an arbitrary entity from a context entity.

herf	A string. The hyperlink to access an entity.
id	A string. The identifier of an entity.
name	A string. The name of the related entity if applicable (e.g. name of the customer, name of the bill, name of the product etc...).
role	A string. The role of the related entity in the context of the contained resource (e.g. disputedBill, damagedDevice
@referredType	A string. Indicates the type (class) of related entity. For example, Product Order Customer Bill, Payment, etc.

RelatedPartyRef relationship

RelatedParty reference. A related party defines party or party role linked to a specific entity.

id	A string. Unique identifier of a related party.
href	A string. Reference of the related party, could be a party reference or a party role reference.
role	A string. Role of the related party.
name	A string. Name of the related party.
@referredType	A string. Indicates the type (class) of related party. For example, Organization or Individual (if party), Customer, Supplier, etc. (if party role).

Json representation sample

We provide below the json representation of an example of a 'Trouble Ticket' resource object

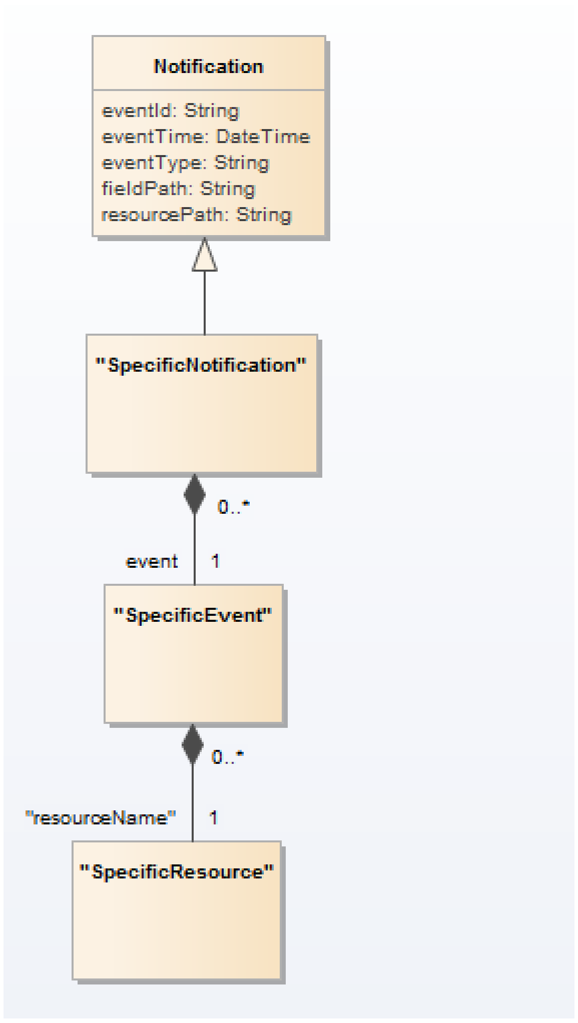
```
{
   "id": "3180",
   "href": "https://host:port/troubleTicket/v2/troubleTicket/3180",
   "name": “Compliant over last bill",

   "externalId": "213",
   "ticketType": " Bill Dispute",
   "creationDate": "2018-05-01T00:00",
   "lastUpdate": "2018-05-01T00:00",
   "description": "I do not accept the last VOD charge, since the movie was constantly interrupted, I had to
quick watching the movie in the middle ",
   "reason": " Bad Quality",
   "severity": "Urgent",
   "priority": "Hight",
   "requestedResolutionDate": "2018-05-01T00:00",
   "expectedResolutionDate": "2018-05-01T00:00",
   "resolutionDate": "2018-05-01T00:00",
   "status": "Pending",

    "statusChangeReason": "Need more information from the customer ",
   "@type": "TroubleTicket",
   "@schemaLocation": "https://host:port/troubleTicket/v2/schema/troubleTicket.yml",
   "relatedEntity": [
      {
      "id": "3472",

      "href": "https://host:port/customerBillManagement/v2/customerBill/8297",

      "role": "Disputed Bill",

      "name": “December Bill"
      "@referredType": "CustomerBill"

    }
  ],
  "statusChange": [
    {
      "status": " Pending ",
      "changeReason": " Need more information from the customer ",
      "changeDate": "2018-05-01T00:00"
    }
  ],
  "note": [
    {
      "date": "2018-05-01T00:00",
      "author": "Mr Jack Hide",
      "text": " This is quite important "
    }
  ],
  "relatedParty": [
    {
      "id": "6675",
      "href": "https://host:port/partyManagement/v2/individual/6675",
      "role": "owner",
      "name": "Gustave Flaubert",
      "@referredType": "Individual"
    },

   {
      "id": "6675",
      "href": "https://host:port/customerManagement/v2/customer/8897",
      "role": "customer",
      "name": "Mr Jack Hide",
      "@referredType": "Customer"
    }



  ],
  "ticketRelationship": [
    {
      "type": "a  ...",

       "id": "a  ...",
       "href": "a  ..."
    }
  ],
  "channel": {
        "id": "8774",
        "name": "Self Service",
        "@type": "Channel"
      },
      "attachment": [
        {
          "description": "Scanned disputed bill",
          "href": "http://hostname:port/documentManagement/v2/attachment/44",
          "id": "44",
          "url": "http://xxxxx",
          "name": "December Bill ",

         "@referredType": "Attachment "
          }
        }
      ]
  }
```

Notification Resource Models

5 notifications are defined for this API

Notifications related to trouble ticket:
- TroubleTicketChangeNotification
- TroubleTicketStatusChangeNotification
- TroubleTicketCreationNotification
- TroubleTicketResolvedNotification
- TroubleTicketInformationRequiredNotification

The notification structure for all notifications in this API follow the pattern depicted by the figure below.
A notification resource (depicted by "SpecificNotification" placeholder) is a sub class of a generic Notification structure containing an id of the event occurrence (eventId), an event timestamp (eventTime), and the name of the notification resource (eventType).
This notification structure owns an event structure ("SpecificEvent" placeholder) linked to the resource concerned by the notification using the resource name as access field ("resourceName" placeholder).



Trouble Ticket Change Notification

Notification sent when changing a trouble ticket resource.

Json representation sample

We provide below the json representation of an example of a 'TroubleTicketChangeNotification' notification object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"TroubleTicketChangeNotification",
     "event": {
       "troubleTicket" :
          {-- SEE TroubleTicket RESOURCE SAMPLE --}
    }
}
```

Trouble Ticket Status Change Notification

Notification status change case for resource trouble ticket

Json representation sample

We provide below the json representation of an example of a 'TroubleTicketStatusChangeNotification' notification object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"TroubleTicketStatusChangeNotification",
     "event": {
        "troubleTicket" :
            {-- SEE TroubleTicket RESOURCE SAMPLE --}
    }
}
```

Trouble Ticket Creation Notification

Notification sent when a new trouble ticket resource is created.

Json representation sample

We provide below the json representation of an example of a 'TroubleTicketCreationNotification' notification object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"TroubleTicketCreationNotification",
     "event": {
        "troubleTicket" :
            {-- SEE TroubleTicket RESOURCE SAMPLE --}
    }
}
```

Trouble Ticket Resolved Notification

Notification resolved case for resource trouble ticket

Json representation sample

We provide below the json representation of an example of a 'TroubleTicketResolvedNotification' notification object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"TroubleTicketResolvedNotification",
     "event": {
        "troubleTicket" :
            {-- SEE TroubleTicket RESOURCE SAMPLE --}
    }
}
```

Trouble Ticket Information Required Notification

Notification sent when information from user is required concerning a trouble ticket resource

- “resourcePath” allows to precise if it is a data at order level or at orderItem level (and which one of them) that is missing
- “fieldPath” details which field is missing. Its structure is quite similar to GET filter criteria:
    - “missing=” points at the missing field
    - “&<criteria>” can be used to identify a specific element in lists

Json representation sample

We provide below the json representation of an example of a 'TroubleTicketInformationRequiredNotification' notification object for example: attachment is missing

```
{
    "eventId":"00001",
    "eventTime":"2018-11-16T16:42:25-04:00",
    "eventType":"TroubleTicketInformationRequiredNotification",
     "resourcePath":"/troubleTicket/3180 ",

     "fieldPath":"missing=attachment”,
    "event": {
      "troubleTicket" : {

          "id": "3180",
          "href": "https://host:port/troubleTicket/v2/troubleTicket/3180",
          "name”: “Compliant over last bill"

     }
    }
 }
```

API OPERATIONS

Remember the following Uniform Contract:

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource.
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute Task Resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods.

Filtering and attribute selection rules are described in the TMF REST Design Guidelines.

Notifications are also described in a subsequent section.

Operations on Trouble Ticket

List Trouble Ticket

GET /troubleTicket?fields=...&{filtering}

Description

This operation list trouble ticket entities.
Attribute selection is enabled for all first level attributes.
Filtering may be available depending on the compliance level supported by an implementation.

Usage Samples

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

Request

```
GET /troubleTicket/v2/troubleTicket
Accept: application/json
```

Response

```
200
```

```
[

{
   "id": "3180",
   "href": "https://host:port/troubleTicket/v2/troubleTicket/3180",
   "name”: “Compliant over last bill",

   "externalId": "213",
   "ticketType": " Bill Dispute",
   "creationDate": "2018-05-01T00:00",
   "lastUpdate": "2018-05-01T00:00",
   "description": "I do not accept the last VOD charge, since the movie was constantly interrupted, I had to
```

```
quick watching the movie in the middle ",
  "reason": " Bad Quality",
  "severity": "Urgent",
  "priority": "Hight",
  "requestedResolutionDate": "2018-05-01T00:00",
  "expectedResolutionDate": "2018-05-01T00:00",
  "resolutionDate": "2018-05-01T00:00",
  "status": "Pending",

   "statusChangeReason": "Need more information from the customer ",
  "@type": "TroubleTicket",
  "@schemaLocation": "https://host:port/troubleTicket/v2/schema/troubleTicket.yml",
  "relatedEntity": [
    {
    "id": "3472",

    "href": "https://host:port/customerBillManagement/v2/customerBill/8297",

    "role": "Disputed Bill",

    "name": “December Bill"

    "@referredType": "CustomerBill"

    }
  ],
  "statusChange": [
    {
      "status": " Pending ",
      "changeReason": " Need more information from the customer ",
      "changeDate": "2018-05-01T00:00"
    }
  ],
  "note": [
    {
      "date": "2018-05-01T00:00",
      "author": "Mr Jack Hide",
      "text": " This is quite important "
    }
  ],
  "relatedParty": [
    {
      "id": "6675",
      "href": "https://host:port/partyManagement/v2/individual/6675",
      "role": "owner",
      "name": "Gustave Flaubert",
      "@referredType": "Individual"
    },

   {
      "id": "6675",
      "href": "https://host:port/customerManagement/v2/customer/8897",
      "role": "customer",
      "name": "Mr Jack Hide",
      "@referredType": "Customer"
    }




  ],
  "ticketRelationship": [
    {
      "type": "a  ...",

       "id": "a  ...",
       "href": "a  ..."
    }
  ],
  "channel": {
    "id": "8774",
    "name": "Self Service",
    "@type": "a string ..."
  },
  "attachment": [
    {
      "description": "Scanned disputed bill",
      "href": "http://hostname:port/documentManagement/v2/attachment/44",
      "id": "44",
      "url": "http://xxxxx",
      "name": "December Bill ",

      "@referredType": "Attachment "
      }
    }
  ]
}

]
```

Retrieve Trouble Ticket

GET /troubleTicket/{id}?fields=...&{filtering}

Description

This operation retrieves a trouble ticket entity.
Attribute selection is enabled for all first level attributes.
Filtering on sub-resources may be available depending on the compliance level supported by an implementation.

Usage Samples

Here's an example of a request for retrieving a trouble ticket resource.

Request

```
GET /troubleTicket/v2/troubleTicket/3180
Accept: application/json
```

Response

```
200

{
   "id": "3180",
   "href": "https://host:port/troubleTicket/v2/troubleTicket/3180",
   "name”: “Compliant over last bill",

   "externalId": "213",
   "ticketType": " Bill Dispute",
   "creationDate": "2018-05-01T00:00",
   "lastUpdate": "2018-05-01T00:00",
   "description": "I do not accept the last VOD charge, since the movie was constantly interrupted, I had to
quick watching the movie in the middle ",
   "reason": " Bad Quality",
   "severity": "Urgent",
   "priority": "Hight",
   "requestedResolutionDate": "2018-05-01T00:00",
   "expectedResolutionDate": "2018-05-01T00:00",
   "resolutionDate": "2018-05-01T00:00",
   "status": "Pending",

    "statusChangeReason": "Need more information from the customer ",
   "@type": "TroubleTicket",
   "@schemaLocation": "https://host:port/troubleTicket/v2/schema/troubleTicket.yml",
   "relatedEntity": [
     {
     "id": "3472",

     "href": "https://host:port/customerBillManagement/v2/customerBill/8297",

     "role": "Disputed Bill",

     "name”: “December Bill"

     "@referredType": "CustomerBill"

     }
   ],
   "statusChange": [
     {
       "status": " Pending ",
       "changeReason": " Need more information from the customer ",
       "changeDate": "2018-05-01T00:00"
     }
   ],
   "note": [
     {
       "date": "2018-05-01T00:00",
       "author": "Mr Jack Hide",
       "text": " This is quite important "
     }
   ],
   "relatedParty": [
     {
       "id": "6675",
       "href": "https://host:port/partyManagement/v2/individual/6675",
       "role": "owner",
       "name": "Gustave Flaubert",
       "@referredType": "Individual"
     },

    {
       "id": "6675",
       "href": "https://host:port/customerManagement/v2/customer/8897",
       "role": "customer",
           "name": "Mr Jack Hide",
           "@referredType": "Customer"
       }




   ],
   "ticketRelationship": [
       {
           "type": "a ...",

            "id": "a ...",
            "href": "a ..."
       }
   ],
   "channel": {
       "id": "8774",
       "name": "Self Service",
       "@type": "a string ..."
   },
   "attachment": [
       {
           "description": "Scanned disputed bill",
           "href": "http://hostname:port/documentManagement/v2/attachment/44",
           "id": "44",
           "url": "http://xxxxx",
           "name": "December Bill ",

           "@referredType": "Attachment "
           }
       }
   ]
}
```

Create Trouble Ticket

POST /troubleTicket

Description

This operation creates a trouble ticket entity.

Mandatory and Non Mandatory Attributes

The following tables provides the list of mandatory and non mandatory attributes when creating a trouble ticket, including any possible rule conditions and applicable default values. Notice that it is up to an implementer to add additional mandatory attributes.

Mandatory Attributes	Rule
description	
severity	
ticketType	

Non Mandatory Attributes	Default Value	Rule
externalId		
name		
ticketType		
creationDate		Populated by the server
lastUpdate		Populated by the server
reason		
priority		
requestedResolutionDate		
expectedResolutionDate		
resolutionDate		
status		
@type		
@baseType		
@schemaLocation		
relatedEntity		
statusChange		Populated by the server
note		

Non Mandatory Attributes	Default Value	Rule
relatedParty		
ticketRelationship		
channel		
attachment		

Usage Samples

Here's an example of a request for creating a trouble ticket resource. In this example the request only passes mandatory attributes.

Request

```
POST /troubleTicket/v2/troubleTicket
Content-Type: application/json

{
    "description": "Compliant over last invoice",
    "severity": "Urgent",
    “ticketType”: “billingTicket”

    "@type": "TroubleTicket”
}
```

Response

```
201

{
    "id": "3180",
    "href": "https://host:port/troubleTicket/v2/troubleTicket/3180",
    "description": "Compliant over last invoice",
    "severity": "Urgent",
    “ticketType”: “billingTicket”
    "@type": " TroubleTicket"
}
```

Patch Trouble Ticket

PATCH /troubleTicket/{id}

Description

This operation allows partial updates of a trouble ticket entity. Support of json/merge (https://tools.ietf.org/html/rfc7386) is mandatory, support of json/patch (http://tools.ietf.org/html/rfc5789) is optional.

Note: If the update operation yields to the creation of sub-resources or relationships, the same rules concerning mandatory sub-resource attributes and default value settings in the POST operation applies to the PATCH operation. Hence these tables are not repeated here.

Patchable and Non Patchable Attributes

The tables below provide the list of patchable and non patchable attributes, including constraint rules on their usage.

Patchable Attributes	Rule
externalId	
name	
ticketType	
description	
reason	
severity	
priority	
requestedResolutionDate	
expectedResolutionDate	
resolutionDate	
status	

Patchable Attributes	Rule
relatedEntity	
note	
relatedParty	
ticketRelationship	
channel	
attachment	

Non Patchable Attributes	Rule
id	
href	
creationDate	
lastUpdate	
statusChange	
@baseType	
@type	
@schemaLocation	

Usage Samples

Here's an example of a request for patching a trouble ticket resource.

Request

```
PATCH /troubleTicket/v2/troubleTicket/3180
Content-Type: application/merge-patch+json

{
    "description": "New description.."
}
```

Response

201

```
{
   "id": "3180",
   "href": "https://host:port/troubleTicket/v2/troubleTicket/3180",
   "name”: “Compliant over last bill",

    "externalId": "213",
   "ticketType": "a string ...",
   "creationDate": "2018-05-01T00:00",
   "lastUpdate": "2018-05-01T00:00",
   "description": "New description..",
   "reason": "a string ...",
   "severity": "Urgent",
   "priority": "a string ...",
   "requestedResolutionDate": "2018-05-01T00:00",
   "expectedResolutionDate": "2018-05-01T00:00",
   "resolutionDate": "2018-05-01T00:00",
   "status": "Received",
   "@baseType": "a string ...",
   "@type": "a string ...",
   "@schemaLocation": "a string ...",
   "relatedEntity": [
      {
         "herf": "a string ...",
         "id": "8244",
         "name": "a string ..."
      }
   ],
   "statusChange": [
      {
         "status": "a  ...",
         "changeReason": "a  ...",
         "changeDate": "a  ..."
      }
   ],
   "note": [
      {
        “id”: “7896”
```

```
           "date": "2018-05-01T00:00",
            "author": "Mr Hide",
            "text": "This is quite important"
        }
    ],
    "relatedParty": [
        {
            "id": "6675",
            "href": "https://host:port/partyManagement/organization/6675",
            "role": "owner",
            "name": "Gustave Flaubert",
            "@referredType": "a string ..."
        }
    ],
    "ticketRelationship": [
        {
            "type": "a  ...",
                "id": "a  ...",
                "href": "a  ..."
        }
    ],
    "channel": {
        "id": "8774",
        "name": "a string ...",
        "@type": "a string ..."
    },
    "attachment": [
        {
             "description": "Scanned disputed bill",
            "href": "http://hostname:port/documentManagement/v2/attachment/44",
            "id": "44",
            "url": "http://xxxxx",
            "name": "December Bill ",

            "@referredType": "Attachment "
            }
        }
    ],
}
```

Delete Trouble Ticket

DELETE /troubleTicket/{id}

Description

This operation deletes a trouble ticket entity. Typically restricted to admin role

Usage Samples

Here's an example of a request for deleting a trouble ticket resource.

Request

```
DELETE /troubleTicket/v2/troubleTicket/3180
```

Response

```
204
```

API NOTIFICATIONS

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines reproduced below.

Register listener

POST /hub

Description

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

Behavior

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

Usage Samples

Here's an example of a request for registering a listener.

Request

```
POST /api/hub

Accept: application/json


{"callback": "http://in.listener.com"}
```

Response

```
201

Content-Type: application/json

Location: /api/hub/42


{"id":"42","callback":"http://in.listener.com","query":null}
```

Unregister listener

DELETE /hub/{id}

Description

Clears the communication endpoint address that was set by creating the Hub..

Behavior

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

Usage Samples

Here's an example of a request for un-registering a listener.

Request

```
DELETE /api/hub/42

Accept: application/json
```

Response

```
204
```

Publish Event to listener

POST /client/listener

Description

Clears the communication endpoint address that was set by creating the Hub.

Provides to a registered listener the description of the event that was raised. The /client/listener url is the callback url passed when registering the listener.

Behavior

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

Usage Samples

Here's an example of a notification received by the listener. In this example “EVENT TYPE” should be replaced by one of the notification types supported by this API (see Notification Resources Models section) and EVENT BODY refers to the data structure of the given notification type.

Request

```
POST /client/listener

Accept: application/json


{

   "event": {

            EVENT BODY

         },

   "eventType": "EVENT_TYPE"

}
```

Response

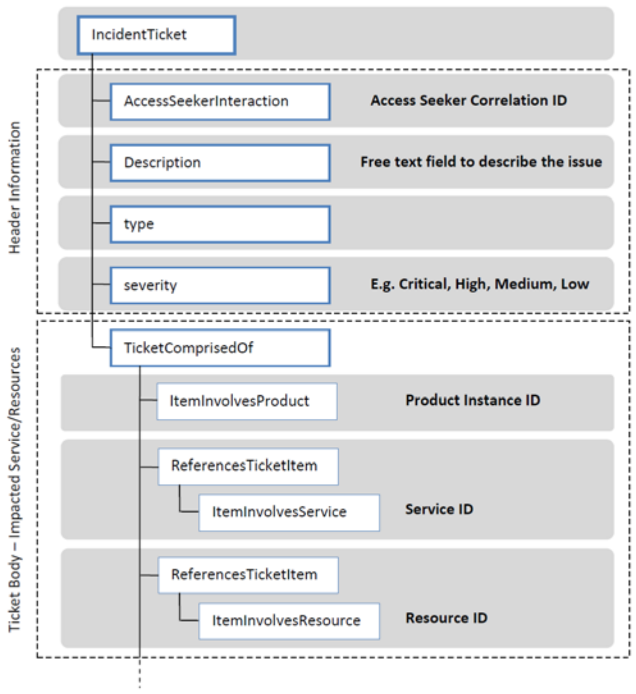
201

For detailed examples on the general TM Forum notification mechanism, see the TMF REST Design Guidelines.

APPENDIX A. : ALIGNMENT WITH NBN CO. SPECIFICATIONS

1. The following diagram From NBN Co shows the generic, high level structure for a trouble ticket submission message. NBN Co Operations Manual: [sfaa-wba2-operations-manual_20150213.pdf]

Unlike NBNCo, **TMF Trouble Ticket is atomic, that is it does NOT contain TicketItems**.



NBNCo	TM Forum	Comments
AccessSeekerInteraction	*externalId*	
	name	
Description	*description*	
Type	*ticketType*	
Severity	*severity*	
interactionDate	*creationDate*	
interactionDateComplete	*not supported*	use statusChangeDate
plannedCompletionDate	*requestedResolutionDate*	
interactionStatus	*status*	
interactionSubStatus	*not supported*	
	statusChange.changeReason	

NBNCo	TM Forum	Comments
	statusChange.changeDate	
resolvedDate	*resolutionDate*	
	troubleTicket.relatedParty[]	Refers to end-user, CSR, …
TicketItem.InvolvesProduct/Service	*troubleTicket.relatedEntity[]*	Product, Service, Resource, …

The following tables list the supported/non-supported processes and touchpoints as defined in

NBNCo - B2B Interaction Business Processes – Technical Specification, 02/01/2013

TT-BP001 : Assurance Ticket Proccess		**supported**
PH-TP001	requestTroubleTicketCreate	TroubleTicketCreatedNotification
PH-TP002	queryTroubleTicketDetails	GET trubleTicket
PH-TP002.1	responseTroubleTicketDetails	HTTP response to GET
PH-TP004	notifyKeepCustomerInformed	TroubleTicketChangedNotification
PH-TP005	notifyTroubleTicketAcknowledged	TroubleTicketStatusChangedNotification
PH-TP006	notifyTroubleTicketAccepted	TroubleTicketStatusChangedNotification
PH-TP007	notifyTroubleTicketRejected	TroubleTicketStatusChangedNotification
PH-TP030	requestTroubleTicketClearance	TroubleTicketClearanceRequestNotification
PH-TP030.1	responseTroubleTicketClearance	PATCH troubleTicket.status ('resolved'->'closed')
PH-TP014	notifyTroubleTicketResolved	TroubleTicketStatusChangedNotification
PH-TP020	notifyInformationRequired	TroubleTicketInformationRequiredNotification
PH-TP022	notifyTroubleTicketClosed	TroubleTicketStatusChangedNotification

TT-BP003 : Query Trouble Ticket History or Details	**supported (*)**

PH-TP002	queryTroubleTicketDetails	GET troubleTicket
PH-TP002.1	responseTroubleTicketDetails	HTTP response to GET
	(*) history not supported	

TT-BP005: TroubleTicketAmendment		supported
PH-TP011	requestTroubleTicketAmend	PATCH troubleTicket
PH-TP011.1	responseTroubleTicketAmend	HTTP response

TT-BP006: Trouble ticket Jeopardy		not supported
PH-TP004	notifyCustomerJeopardy	not supported

TT-BP007: Planned changed / hazard		not supported
CM-TP001	notifyPlannedChange	not supported
CM-TP004	notifyKeepCustomerInformed	not supported

TT-BP008: Notify Network Fault		not supported
PH-TP004	notifyTroubleTicketCreated	not supported

TT-BP009: RequestMoreTime		not supported
PH-TP020	notifyInformationRequired	InformationRequiredNotification
PH-TP004	notifyKeepCustomerInformed	TicketChangedNotification

PH-TP025	requestMoreTime	not supported
PH-TP025.1	responseMoreTime	not supported
PH-TP026	notifyInformationRequiredReminder	not supported
PH-TP022	notifyTroubleTicketClosed	TicketStatusChangedNotification

TT-BP010: QueryTroubleTicketAttachment		**not supported**
PH-TP029	queryTroubleTicketAttachment	not supported
PH-TP029.1	responseTroubleTicketAttachment	not supported

ACKNOWLEDGEMENTS

Version History

Release Number	Date	Release led by:	Description
1.0	27/07/2013	Christian Traxler Jean Luc Tymen Andreas Polz John Storrie Jerome Hannebelle Pierre Gauthier	Initial.
2.0	15/04/2016	Pierre Gauthier Mariano Belaunde	Regenerated from API Data Model and re-branded.
3.0	04/04/2018	Jacob Avraham	Align with REST Design Guideline (DG3) Enhanced model with new requirements
3.0.1	26-Jun-2018	Adrienne Walcott	Formatting/style edits prior to R18 publishing.

Release History

Release Number	Date	Release led by:	Description
Release 18.0.0	25-Jun-2018	Christian Traxler Jean Luc Tymen Andreas Polz John Storrie Jerome Hannebelle Pierre Gauthier	Initial.

Contributors to Document

Pierre Gauthier pgauthier@tmforum.org	TM Forum
Jacob Avraham jacoba@amdocs.com	Amdocs
Sophie Bouleau sophie.bouleau@orange.com	Orange
Ludovic Robert ludovic.robert@orange.com	Orange
Luis Velarde Tazon luis.velardetazon@telefonica.com	Telefonica
Guillermo Martinez Del Camino guillermo.martinezdelcamino@telefonica.com	Telefonica