**TM Forum Specification**

**API Conformance Template**

<<Document Number>>

<<Release xx.x.x>>

<<Month Year>>

|  |  |
| --- | --- |
| **Latest Update: TM Forum Release xx.x** | **Document Status** |
| **Version x.x.x** | **IPR Mode: RAND** |

# NOTICE

Copyright © TM Forum 2018. 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. However, this document itself may not be modified in any way, including by removing the copyright notice or references to TM FORUM, except as needed for the purpose of developing any document or deliverable produced by a TM FORUM Collaboration Project Team (in which case the rules applicable to copyrights, as set forth in the [TM FORUM IPR Policy](http://www.tmforum.org/IPRPolicy/11525/home.html), must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by TM FORUM or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and TM FORUM DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Direct inquiries to the TM Forum office:

4 Century Drive, Suite 100

Parsippany, NJ 07054, USA

Tel No. +1 973 944 5100

Fax No. +1 973 944 5110

TM Forum Web Page: [www.tmforum.org](http://www.tmforum.org/)

# Table of Contents

[NOTICE 2](#_Toc514836059)

[Table of Contents 3](#_Toc514836060)

[Introduction - API DESCRIPTION 4](#_Toc514836061)

[RESOURCE MODEL CONFORMANCE 5](#_Toc514836062)

[API MANDATORY AND OPTIONAL RESOURCES 5](#_Toc514836063)

[ResourceXYZ MANDATORY AND OPTIONAL ATTRIBUTES 5](#_Toc514836064)

[NOTIFICATION MODEL CONFORMANCE 6](#_Toc514836065)

[API MANDATORY AND OPTIONAL NOTIFICATIONS 6](#_Toc514836066)

[<Notification Name> MANDATORY AND OPTIONAL ATTRIBUTES 6](#_Toc514836067)

[API OPERATIONS CONFORMANCE 8](#_Toc514836068)

[API MANDATORY AND OPTIONAL OPERATIONS 8](#_Toc514836069)

[API GET OPERATION CONFORMANCE 9](#_Toc514836070)

[GET /RESOURCEXYZ [for instance /troubleTicket] 9](#_Toc514836071)

[API POST OPERATION CONFORMANCE 11](#_Toc514836072)

[POST /ResourceXYZ [for instance POST /troubleTicket] 11](#_Toc514836073)

[API PUT OPERATION CONFORMANCE 14](#_Toc514836074)

[PUT RESOURCEXYZ [for instance /troubleTicket/{ID}] 14](#_Toc514836075)

[API PATCH OPERATION CONFORMANCE 15](#_Toc514836076)

[PATCH RESOURCEXYZ [for instance /troubleTicket/{ID}] 15](#_Toc514836077)

[API DELETE OPERATION CONFORMANCE 17](#_Toc514836078)

[DELETE /RESOURCEXYZ [for instance /troubleTicket/{ID}] 17](#_Toc514836079)

[API CONFORMANCE TEST SCENARIOS 18](#_Toc514836080)

[RESOURCEXYZ resource TEST CASES 18](#_Toc514836081)

[Release History 25](#_Toc514836082)

# Introduction - API DESCRIPTION

See the introduction of the related specification document.

# RESOURCE MODEL CONFORMANCE

## API MANDATORY AND OPTIONAL RESOURCES

For the resources defined by the API fill the following table and indicate which ones are mandatory and which ones are optional.

|  |  |  |
| --- | --- | --- |
| Resource Name | Mandatory or Optional | Comments |
| ResourceXYZ | M |  |

## ResourceXYZ MANDATORY AND OPTIONAL ATTRIBUTES

For the resources defined by the API the following table indicates which ones are mandatory and which ones are optional.

|  |  |  |
| --- | --- | --- |
| Attribute Name | Mandatory or Optional | Comments |
| id | M (in response messages)  O (otherwise) | Generated by the server and provided in the response upon resource creation.  Accepted in entity-creation requests if the server supports the incoming identifier as the reference to create new resources |
| href | M (in response messages)  O (otherwise) | Value in response must be the same as the one set in Location header provided upon entity creation |

# NOTIFICATION MODEL CONFORMANCE

The Pub/Sub models are common and described in the TMF REST Design Guidelines. Use the following templates to describe the Hub Mandatory and Optional attributes and filtering support.

## API MANDATORY AND OPTIONAL NOTIFICATIONS

For the Notifications defined by the API the following table indicates which ones are mandatory and which ones are optional.

|  |  |  |
| --- | --- | --- |
| Notification Name | Mandatory or Optional | Comments |
| resourceXYZCreationNotification | O |  |
| resourceXYZRemoveNotification | O |  |

## <Notification Name> MANDATORY AND OPTIONAL ATTRIBUTES

For every single notification supported by the API provide a table indicating which attributes are mandatory and which ones are optional. Add documentation on the required format (date for example) as required.

<The table below is a reference example (actual attributes may depend on the specific resource in the API)>

| Attribute Name | Mandatory or Optional | Comments |
| --- | --- | --- |
| id | M | . |
| eventTime | M |  |
| eventType | M |  |
| resourcePath | M/O | Mandatory for CustomerResourceXYZCreationNotification only  NA otherwise |
| event | M | Filled with the Customer info  Refer to the detailed mandatory/optional data regarding each notification below. |

# API OPERATIONS CONFORMANCE

For every single resource use the following templates and define what operations are optional and what operations are mandatory.

## API MANDATORY AND OPTIONAL OPERATIONS

The following table indicates which ones are mandatory and which ones are optional for each one of the resources in the API (default is for all resources).

|  |  |  |
| --- | --- | --- |
| Uniform API Operation | Mandatory/Optional | Comments |
| GET | M for all resources | GET must be used to retrieve a representation of a resource |
| POST | M for ResourceXYZ | POST must be used to create a new resource |
| PUT | O for all resources | PUT must be used to completely update a resource identified by its resource URI |
| PATCH | O for all resources | PATCH must be used to partially update a resource |
| DELETE | O for all resources | DELETE must be used to remove a resource |

# API GET OPERATION CONFORMANCE

For every single resource use the following template to specify the mandatory and optional features supported by the GET operation.

**Definitions**

**Filtered Search:** A filtered search can be applied using query parameters in order to obtain only the resource entities that meet the criteria defined by the filtering parameters included in the query request. Several elements can be applied to the filtered search. In that case logic, a logical AND is applied to combine the criteria (e.g.:?severity=<value> &status=<value>)

**Attribute selection (Filtered Response Data):** In order to apply a filter and limit the number of attributes included in the response, the GET request can include the “?fields=” query parameter. Several elements can be applied to the filter. In that case, a logical AND is applied to combine the values (e.g.:?fields=severity,status) will provide in the response only the values assigned to attributes category and channel. Attribute selection capabilities are the same for collections retrieval and individual resource queries

## GET /RESOURCEXYZ [for instance /troubleTicket]

<one line description of the operation>

|  |  |  |
| --- | --- | --- |
| GET | M |  |
| Response Status Code 200 | M |  |
| Other Status Codes | NA |  |

<Use the following to indicate which attributes can be used for filtering purpose and attribute selection and must be supported by the server. The table below is a reference example (actual attributes may depend on the specific resource in the API)>

Level refers to the structure of the resource data attributes, a second level attribute refers to an information element within an attribute with a complex structure

| **Attribute name** | **Filtered search**  **First Level** | **Filtered search**  **N Level** | **Attribute Selection First Level** | **Attribute Selection**  **N Level** |
| --- | --- | --- | --- | --- |
| id | NA | NA | M | NA (only has one level) |
| href | NA | NA | M | NA (only has one level) |
| correlationId | O | NA | M | NA (only has one level) |
| description | O | NA | M | NA (only has one level) |
| type | M | NA | M | NA (only has one level) |
| creationDate | O | NA | M | NA (only has one level) |
| status | O | NA | M | NA (only has one level) |
| relatedParty | NA (is complex structure) | O | M | O |

# API POST OPERATION CONFORMANCE

## POST /ResourceXYZ [for instance POST /troubleTicket]

This Uniform Contract operation is used to <one line description of the operation>

|  |  |  |
| --- | --- | --- |
| POST | M |  |
| Status Code 201 | M |  |
| Other Status Codes | NA |  |

The response to this operation must include a Location header set to /troubleTicket /ticket /{ID} where {ID} indicates the identifier assigned by the server to the new Product Offering resource created

Specify the attributes required when an entity is created (and their default values if not) as well as attributes with special considerations:

<The table below is a reference example (actual attributes may depend on the specific resource in the API)>

| Attribute name | Mandatory | Default | Rule |
| --- | --- | --- | --- |
| id | N |  | Accepted in entity-creation requests if the server supports the incoming identifier as the reference to create new resources |
| description | Y |  |  |
| type | Y |  |  |
| relatedParty.href | N |  | Mandatory if relatedParty included  The consumer must indicate the identifier for every relatedParty assigned to the ticket |

Further specify any rules on the creation of the entity

|  |  |
| --- | --- |
| Rule name | Rule |
|  |  |
|  |  |
|  |  |

The response from the server must include a BODY with the contents of the new resource created, filled with at least the same information elements that were included in the request and are supported by the server. Notice that the value stored by the server may be different than the one set in the request (e.g.: status may be differently set by the server after processing than the one requested by the requestor)

If the POST request includes optional parameters (as per the model resource definition) that are not supported by the server, then the server must reject the request (replying with a 4xx error response) indicating the parameter not supported.

Specify the parameters must be supported by the server when included in the request to create a new resource

* parameter1
* parameter2
* parameter3
* ...

Specify the attributes that must be included in the BODY of the response, even if they are not included in the request because they are mandatory in the definition of the resource to be created as per the resource model defined

* parameter4
* parameter5
* parameter6
* ...

# API PUT OPERATION CONFORMANCE

## PUT RESOURCEXYZ [for instance /troubleTicket/{ID}]

This Uniform Contract operation is used to partially update the representation of a managed

|  |  |  |
| --- | --- | --- |
| PUT | O |  |
| Status Code 201 | M |  |
| Other Status Codes | NA |  |

Specify the attributes required when an entity is completely updated (and their default values if not) as well as attributes with special considerations: This is typically the same as for resource creation since the PUT completely updates the resource.

# API PATCH OPERATION CONFORMANCE

## PATCH RESOURCEXYZ [for instance /troubleTicket/{ID}]

This Uniform Contract operation is used to partially update the representation of a managed entity or a task.

|  |  |  |
| --- | --- | --- |
| PATCH | O |  |
| Status Code 201 | M |  |
| Other Status Codes | NA |  |

Specify which attributes are patchable using the following table:

<The table below is a reference example (actual attributes may depend on the specific resource in the API)>

| Attribute Name | | Patchable ? (Y/N) | Rule |
| --- | --- | --- | --- |
| id | | N |  |
| href | | N |  |
| externalId | | N |  |
| description | | Y |  |
| state | | Y | in respect of the state diagram in the spec |
| relatedParty | | Y/N | Yes for adding a new related party (PATCH list)  No for changing an already related party |
| orderItem | | Y |  |
|  | id | N |  |
|  | action | N |  |
|  | state | Y | in respect of the state diagram in the spec |
|  | billingAccount | N |  |
|  | appointment | Y |  |
|  | productOffering | N |  |
|  | product | Y |  |

Further document any rules that must be implemented when patching attributes

|  |  |
| --- | --- |
| Rule name | Rule |
|  |  |

# API DELETE OPERATION CONFORMANCE

## DELETE /RESOURCEXYZ [for instance /troubleTicket/{ID}]

This Uniform Contract operation is used to delete a managed entity or a task.

|  |  |  |
| --- | --- | --- |
| DELETE | O |  |
| Status Code 200 | M |  |
| Other Status Codes | NA |  |

# API CONFORMANCE TEST SCENARIOS

This section describes the test scenarios required for the basic CONNECT certification of the API.

Test Cases must be executed in the order defined for each resource because the result from one of the scenarios will be input for the next one.

Requests must be addressed to the endpoint provided for certification, specifically they must be addressed to the URI defined by the concatenation of the {apiRoot} and the specific resource, where the {apiRoot} is defined as **{serverRoot}/troubleTicket/v1**, where {serverRoot} defines the certification endpoint

## RESOURCEXYZ resource TEST CASES

**Nominal Scenarios**

**TC\_Trou\_N1 – Create new Ticket with minimum required information**

* Send a POST message to {apiRoot}/troubleTicket/ with the following contents in the BODY

{

"description": "<anytext>",

"severity": "High",

"type": "device"

}

* Wait for a response from the server with the following characteristics
  + Response Code 201-Created
  + Include a location header in the body set to /{apiRoot}/troubleTicket/{IDtt1} where {IDtt1} indicates the identifier assigned by the server to the new ticket resource
  + The response message includes all mandatory parameters (including description, severity and type that were not sent in the original request)
  + The body of the response matches the values set in the original request
* Send a GET message to /{apiRoot}/troubleTicket/
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicket resource with ID set to {IDtt1}, the same identifier as assigned by the server to the new resource created
  + The response message includes all mandatory parameters
  + The body of the response for the resource with identifier {IDtt1} matches the values set in the original request
* Send a GET message to /{apiRoot}/troubleTicket/{IDtt1}
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The response message includes all mandatory parameters
  + The body of the response includes a TroubleTicket resource structure that matches the values in the original request

**TC\_Trou\_N2 – Create new Ticket with minimum set of parameters supported by server**

* Send a POST message to {apiRoot}/troubleTicket/ with the following contents in the BODY

{

"description": "nanana",

"severity": "Low",

"type": "connectivity",

"status": "nanana",

"correlationId": "123",

"note":

[

{

“author”: “writer n2\_1”,

“text”: “This is the first note in N2”

},

{

“author”: “writer n2\_2”,

“text”: “This is the second note in N2”

}

]

}

* Wait for a response from the server with the following characteristics
  + Response Code 201-Created
  + Include a location header in the body set to /{apiRoot}/troubleTicket/{IDtt2} where {IDtt2} indicates the identifier assigned by the server to the new TroubleTicket resource
  + The response message includes all mandatory parameters (including creationDate, status and statusChangeDate that were not sent in the original request)
  + The body of the response matches the values set in the original request
* Send a GET message to /{apiRoot}/troubleTicket/
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicket resource with ID set to {IDtt2}, the same identifier as assigned by the server to the new resource created
  + The response message includes all mandatory parameters
  + The body of the response for the resource with identifier {IDtt2} matches the values set in the original request
* Send a GET message to /{apiRoot}/troubleTicket/{IDtt2}
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The response message includes all mandatory parameters
  + The body of the response includes a TroubleTicket resource structure that matches the values in the original request

**TC\_Trou\_N3 – Search for Tickets with specific characteristics**

* Send a GET message to /{apiRoot}/troubleTicket
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes at least two ticket resources referring to {IDtt1} and {IDtt2}
  + The body of the response for the resource with each identifier matches the values in the corresponding original request
* Send a GET message to /{apiRoot}/ troubleTicket?severity=High
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicket resource referring to {IDtt1} and there is no reference to TroubleTicket resource {IDtt2}
  + The response message includes all mandatory parameters
  + The body of the response for the resource with identifier {IDtt1} matches the values in the original request
* Send a GET message to /{apiRoot}/troubleTicket?type=connectivity
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicket resource referring to {IDtt2} and there is no reference to TroubleTicket resource {IDtt1}
  + The response message includes all mandatory parameters
  + The body of the response for the resource with identifier {IDtt2} matches the values in the original request

**TC\_Trou\_N4 – Filtered retrieval of Tickets**

* Send a GET message to /{apiRoot}/troubleTicket/{IDtt1}?fields=description
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicket resource referring to {IDtt1} and including only attributes name and status, matching the values in the original request
* Send a GET message to /{apiRoot}/ troubleTicket /{IDtt2}?fields=severity,status
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicket resource referring to {IDtt2} and including only attributes severity and status, matching the values in the original request

Notice that this test case is using parameters ”description”, ”severity” and ”status” to filter the data included in the response but any other parameter could be used

**TC\_Trou\_N5 – Filtered Search and Filtered data response**

* Send a GET message to /{apiRoot}/troubleTicket?severity=High&fields=description
* Wait for a response from the server with the following characteristics
  + Response Code 200-OK
  + The body of the response includes one TroubleTicketresource referring to {IDtt1} and there is no reference to TroubleTicket resource {IDtt2}
  + The body of the response for the resource with each identifier includes only attribute description, matching the values in the corresponding original request

Notice that this test case is using the parameter ”description” to filter the data included in the response but any other parameter could be used

**Error Scenarios**

**TC\_Trou\_E1 – Unknown Trouble Ticket identifier**

* Send a GET message to /{apiRoot}/troubleTicket/{IDtt3}, where {IDtt3} does not match any of the identifiers previously creted in the server
* Wait for a response from the server with the following characteristics
  + Response Code 404-Not Found

**TC\_Trou\_E2 – Invalid Request – Missing mandatory parameter**

* Send a POST message to {apiRoot}/troubleTicket/ with the following contents in the BODY.

{

“description”: “<anytext>”,

“severity”: “High”

}

Notice that this request is missing mandatory parameter ”type” but any other mandatory parameter could be used

* Wait for an error response from the server indicating the mandatory parameter is missing in the request

**TC\_Trou\_E3 – Invalid Request – Missing parameter mandatory in context**

* Send a POST message to {apiRoot}/troubleTicket/ with the following contents in the BODY.

{

“description”: “<anytext>”,

“severity”: “High”,

“type”: “problem”,

“note”:

{

“author”: “writer e3\_1”

}

}

Notice that this request is missing mandatory parameters “text” when information element “note” is included in the request, but any other parameter that becomes mandatory based on the context could be used

* Wait for an error response from the server indicating the mandatory parameter is missing in the request

## Release History

|  |  |  |  |
| --- | --- | --- | --- |
| **Release Number** | **Date** | **Release led by:** | **Description** |
| Release 1.0 | 07/15/2015 | Pierre Gauthier  TM Forum  pgauthier@tmforum.orf | First Release of Draft Version of the Document. |
| Release 1.1 | 03/15/2017 |  | Updated version including Test scenarios |