TM Forum Specification

Partnership Type Management API Conformance Profile

TMF668B

Release 18.0.0

June 2018

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

NOTICE

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

TABLE OF CONTENTS

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
PartnershipType	M	

PARTNERSHIP TYPE MANDATORY AND OPTIONAL ATTRIBUTES

The table below summarizes mandatory and optional attributes for resource "PartnershipType"

Attribute Name	**Mandatory or Optional**	**Comments**
description	O	
href	M (in response messages) O (otherwise)	Url for the created resource
id	M (in response messages) O (otherwise)	Generated by the server
name	M (for resource creation) O (otherwise)	
roleType	O	

NOTIFICATION MODEL CONFORMANCE

<Optional section in basic CONNECT certification>

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 fill the following table and indicate which ones are mandatory and which ones are optional.

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

Notification Name	Mandatory or Optional	Comments
PartnershipTypeCreationNotification	M	
PartnershipTypeRemoveNotification	M	

All attributes of the resource associated with the notification are mandatory

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

Fill the following table and indicate 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 resources: PartnershipType	POST must be used to create a new resource
PUT	O for resources: PartnershipType	PUT must be used to completely update a resource identified by its resource URI
PATCH	M for resources: PartnershipType	PATCH must be used to partially update a resource
DELETE	M for resources: PartnershipType	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

<Repeat for each resource in the API where GET operation can be exercised>

All the GET operations in this API share the same status code pattern.

GET	M	
Response Status Code 200	M	
Other Status Codes	NA	<describe any other relevant response or error code>

/PARTNERSHIPTYPE?FIELDS=...&{FILTERING}

This operation list partnership type entities.
Attribute selection is mandatory for all first level attributes.
Filtering is mandatory for first compliance level (L1) and optional otherwise.

/PARTNERSHIPTYPE/{ID}?FIELDS=...&{FILTERING}

This operation retrieves a partnership type entity.
Attribute selection is mandatory for all first level attributes.
Filtering on sub-resources is optional for all compliance levels.

API POST OPERATION CONFORMANCE

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

<Repeat for each resource in the API where POST operation can be exercised>

All the POST operations in this API share the same status code pattern.

POST	M	
Status Code 201	M	
Other Status Codes	NA	<describe any other relevant response or error code>

/PARTNERSHIPTYPE

This operation creates a partnership type entity.

Mandatory and Non Mandatory Attributes

The following tables provides the list of mandatory and non mandatory attributes when creating a PartnershipType, 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
name	

Non Mandatory Attributes	Default Value	Rule
description		
roleType		

Additional Rules

The following table provides additional rules indicating mandatory fields in sub-resources or relationships when creating a PartnershipType resource.

Context	Mandatory Sub-Attributes
roleType	name

API PUT OPERATION CONFORMANCE

<This section is only required if the PUT operation is marked as mandatory for the basic CONNECT certification>

<Repeat for each resource in the API where PUT operation can be exercised>

All the PUT operations in this API share the same status code pattern.

PUT	O	
Status Code 201	M	
Other Status Codes	NA	<describe any other relevant response or error code>

API PATCH OPERATION CONFORMANCE

<This section is only required if the PATCH operation is marked as mandatory for the basic CONNECT certification> - <Repeat for each resource in the API where PUT operation can be exercised>

All the PATCH operations in this API share the same status code pattern.

PATCH	O	
Status Code 201	M	
Other Status Codes	NA	<describe any other relevant response or error code>

/PARTNERSHIPTYPE/{ID}

This operation allows partial updates of a partnership type 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
name	
description	
roleType	

Non Patchable Attributes	Rule
href	
id	

API DELETE OPERATION CONFORMANCE

<This section is only required if the DELETE operation is marked as mandatory for the basic CONNECT certification>

<Repeat for each resource in the API where DELETE operation can be exercised>

All the DELETE operations in this API share the same status code pattern.

DELETE	O	
Status Code 200	M	
Other Status Codes	NA	<describe any other relevant response or error code>

/PARTNERSHIPTYPE/{ID}

This operation deletes a partnership type entity.

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}/partnershipTypeManagement/v2**, where {serverRoot} defines the certification endpoint

<The section below is a reference example (actual test scenarios may depend on the specific API)>

Partnership Type resource TEST CASES

Nominal Scenarios

TC_Trou_N1 – Create new Partnership Type with minimum required information

- Send a POST message to {apiRoot}/partnershipType/ with the following contents in the BODY

```
{
   "name": "some name",
}
```

- 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}/partnershipType/{IDtt1} where {IDtt1} indicates the identifier assigned by the server to the new ticket resource
  - The response message includes all mandatory parameters (“name” in this case)
  - The body of the response matches the values set in the original request

- Send a GET message to /{apiRoot}/partnershipType/
- Wait for a response from the server with the following characteristics
    - Response Code 200-OK
    - The body of the response includes one Partnership Type 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}/partnershipType/{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 Partnership Type 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}/partnershipType/ with the following contents in the BODY

```
{
  "description": "nanana",
  "name": "some name",
  "roleType":
  [
    {
      “name”: “name of role type 1”,
      “description”: “This is the role type description”
```

```
        },
        {
          “name”: “name of role type 2”,
          “description”: “This is the role type description”
        }
      ]
    }
```

- 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}/partnershipType/{IDtt2} where {IDtt2} indicates the identifier assigned by the server to the new Partnership Type resource

  - The response message includes all mandatory parameters that were not sent in the original request – if any.

  - The body of the response matches the values set in the original request

- Send a GET message to /{apiRoot}/partnershipType/

- Wait for a response from the server with the following characteristics

  - Response Code 200-OK

  - The body of the response includes one Partnership Type 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}/partnershipType/{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 Partnership Type 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}/partnershipType
- Wait for a response from the server with the following characteristics
    - Response Code 200-OK
    - The body of the response includes at least two partnership type 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}/ partnershipType?name=somename
- Wait for a response from the server with the following characteristics
    - Response Code 200-OK
    - The body of the response includes one Partnership Type resource referring to {IDtt1} and there is no reference to Partnership Type 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

Error Scenarios

TC_Trou_E1 – Unknown Partnership Type identifier

- Send a GET message to /{apiRoot}/partnershipType/{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}/partnershipType/ with the following contents in the BODY.

```
{
   “description”: “<anytext>”,
}
```

  Notice that this request is missing mandatory parameter ”name”

- 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}/partnershipType/ with the following contents in the BODY.

```
{
   “description”: “<anytext>”,
   “name”: “my partnership”,
   “roleType”:[
      {
         “description”: “some text”
      }
    ]
}
```

Notice that this request is missing mandatory parameters “name” when information element “roleType” 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.

ACKNOWLEDGEMENTS

DOCUMENT HISTORY

VERSION HISTORY

Version Number	Date	Release led by:	Description
2.0	15-Jul-2015	Pierre Gauthier TM Forum pgauthier@tmforum.org Mariano Belaunde Orange	First Release of Draft Version of the Document.
3.0	15-Jun-2018	Mariano Belaunde	Design Guidelines 3.0
3.0.1	28-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	Pierre Gauthier Mariano Belaunde	Initial Release