TM Forum Specification

Partnership Type Management API REST Specification

TMF668
Release 18.0.0
June 2018

Latest Update: TM Forum Release 18.0.0	Member Evaluation
Version 2.0.1	IPR Mode: RAND

NOTICE

TABLE OF CONTENTS

LIST OF TABLES

N/A

INTRODUCTION

The following document is the specification of the REST API for Partnership Type Management. It includes the model definition as well as all available operations.

SAMPLE USE CASES

Reader will find example of use cases using Usage API in “Open Digital Business Scenarios and Use Cases” document.

SUPPORT OF POLYMORPHISM AND EXTENSION PATTERNS

Support of polymorphic collections and types and schema based extension is provided by means of a list of generic meta-attributes that we describe below. Polymorphism in collections occurs when entities inherit from base entities, for instance a BillingAccount and SettlementAccount inheriting properties from the abstract Account entity.

Generic support of polymorphism and pattern extensions is described in the TMF API Guidelines v3.0 Part 2 document.

The @type attribute provides a way to represent the actual class type of an entity. For example, within a list of Account instances some may be instances of BillingAccount where other could be instances of SettlementAccount. The @type gives this information. All resources and sub-resources of this API have a @type attributes that can be provided when this is useful.

The @referredType can be used within reference entities (like for instance an AccountRef object) to explicitly denote the actual entity type of the referred class. Notice that in reference entities the @type, when used, denotes the class type of the reference itself, such as BillingAccountRef or SettlementAccountRef, and not the class type of the referred object. However, since reference classes are rarely sub-classed, @type is generally not useful in reference objects.

The @schemaLocation property can be used in resources to allow specifying user-defined properties of an Entity or to specify the expected *characteristics* of an entity.

The @baseType attribute gives a way to provide explicitly the base of class of a given resource that has been extended.

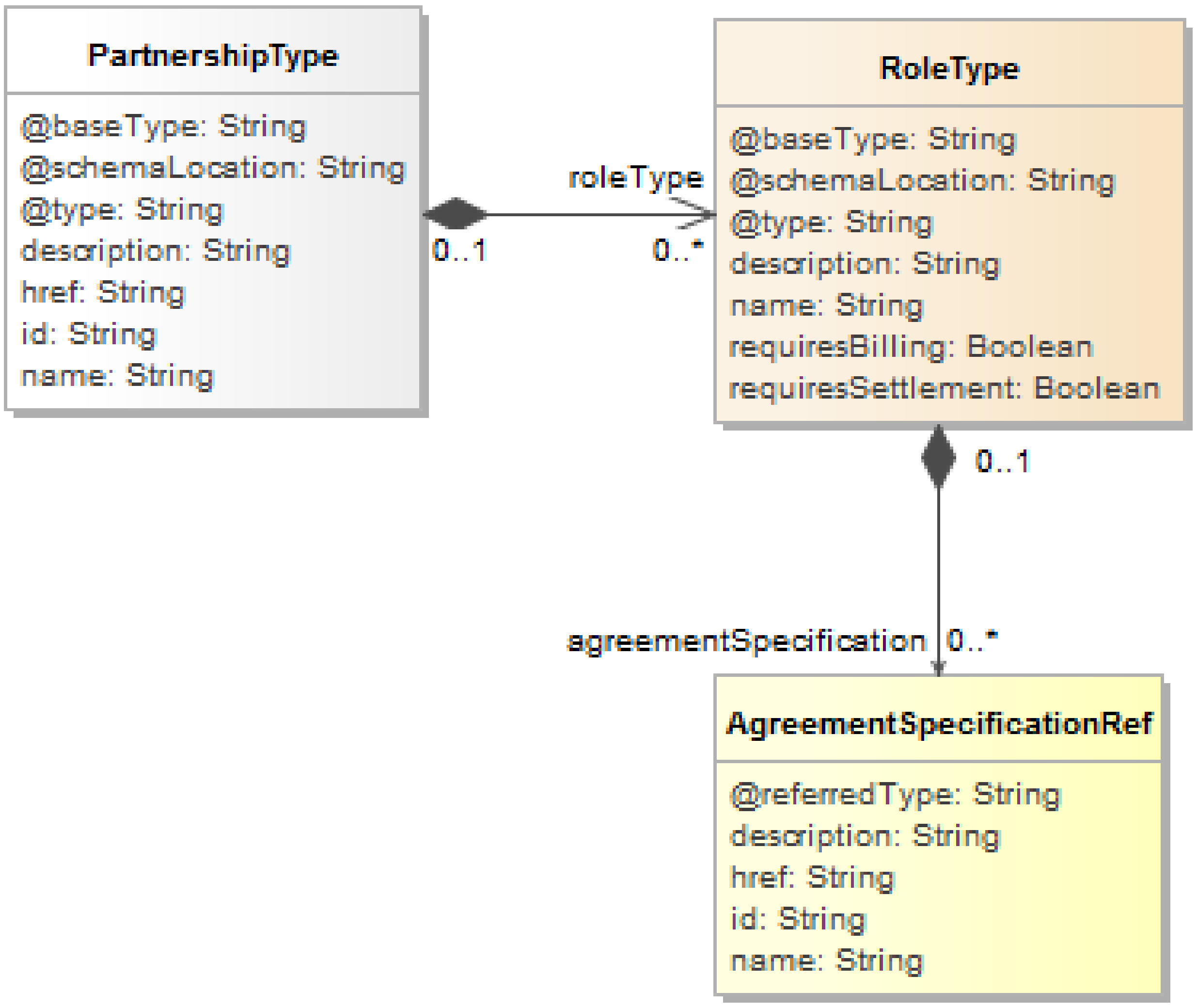
RESOURCE MODEL

Managed Entity and Task Resource Models

PARTNERSHIP TYPE RESOURCE

A partnership type contains all the information for the setup of a partnership of a given kind. This includes the list of identified role types for the partnership with the corresponding agreement specifications.

Resource model



Field descriptions

PartnershipType fields

@baseType	A string. Generic attribute indicating the base class type of the extension class of the current object. Useful only when the class type of the current object is unknown to the implementation.
@schemaLocation	A string. Generic attribute containing the link to the schema that defines the structure of the class type of the current object.
@type	A string. Generic attribute containing the name of the resource class type.
description	A string. An explanatory text regarding this partnership type.
href	A string. The reference url for this partnership type.
id	A string. The identifier of the partnership type.
name	A string. An identifying name for the partnership type.
roleType	A list of role types (RoleType [*]). A RoleType represents the type of a PartyRole, defined in the context of a given type of partnership, such as Buyer, Seller.

RoleType sub-resource

A RoleType represents the type of a PartyRole, defined in the context of a given type of partnership, such as Buyer, Seller.

@baseType	A string. Generic attribute indicating the base class type of the extension class of the current object. Useful only when the class type of the current object is unknown to the implementation.
@schemaLocation	A string. Generic attribute containing the link to the schema that defines the structure of the class type of the current object.
@type	A string. Generic attribute containing the name of the resource class type.
description	A string. An explanatory text documenting the role type.
name	A string. The name of the role type.
requiresBilling	A boolean. Indicates whether billing operations will be associated to parties playing the role.
requiresSettlement	A boolean. Indicates whether settlement operations will be associated to parties playing the role.

agreementSpecification	A list of agreement specification references (AgreementSpecificationRef [*]). An AgreementSpecification represents a template of an agreement that can be used when establishing partnerships.

AgreementSpecificationRef relationship

AgreementSpecification reference. An AgreementSpecification represents a template of an agreement that can be used when establishing partnerships.

@referredType	A string. Generic attribute indicating the name of the class type of the referred resource entity.
description	A string. A narrative that explains in detail what the agreement specification is about.
href	A string. Reference URL of the agreement specification.
id	A string. Unique identifier of the agreement specification.
name	A string. Name of the agreement specification.

Json representation sample

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

```
{
  "description": "This  partnership type ...",
  "href": "https:/host:port/tmf-api/partnershipTypeManagement/v2/partnershipType/4646",
  "id": "4646",
  "name": "Dream Partnership",
  "roleType": [
    [
      {
        "name": "ContentProvider",
        "agreementSpecification": [
          {
            "name": "ContentLicenseAgreement",
            "id": "33"
          }
        ]
      },
      {
        "name": "CloudProvider"
      },
      {
        "name": "Developer",
        "agreementSpecification": [
          {
            "name": "ProfitShareAgreement",
            "id": "32"
          }
```

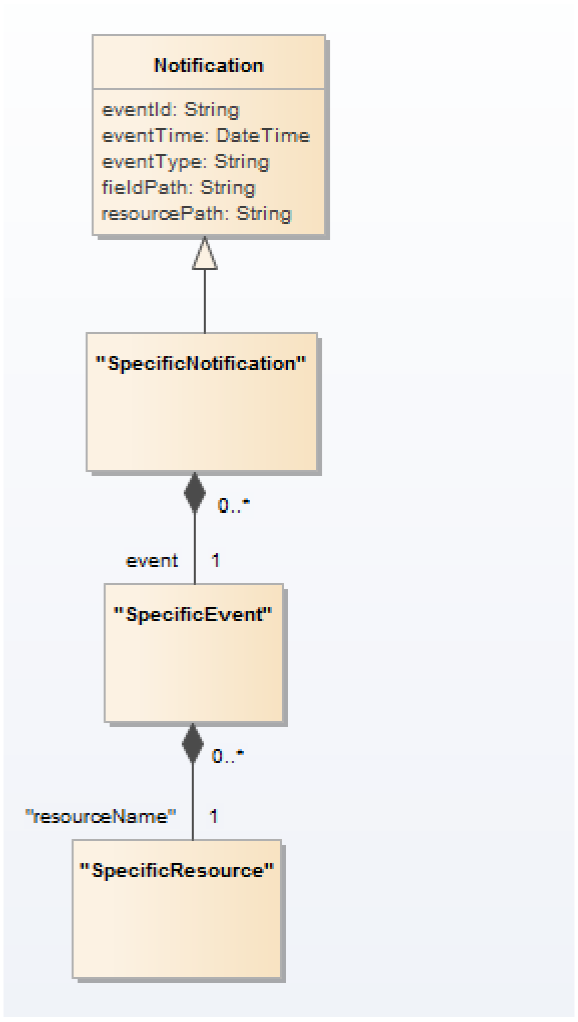
```
            ]
          },
          {
            "name": "Tester"
          }
        ]
      ]
  }
```

Notification Resource Models

2 notifications are defined for this API

Notifications related to PartnershipType:
- PartnershipTypeCreationNotification
- PartnershipTypeRemoveNotification

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).



PARTNERSHIP TYPE CREATION NOTIFICATION

Notification sent when a new PartnershipType resource is created.

Json representation sample

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

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"PartnershipTypeCreationNotification",
     "event": {
        "partnershipType" :
            {-- SEE PartnershipType RESOURCE SAMPLE --}
    }
}
```

PARTNERSHIP TYPE REMOVE NOTIFICATION

Notification sent when removing a PartnershipType resource.

Json representation sample

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

```
{
   "eventId":"00001",
   "eventTime":"2015-11-16T16:42:25-04:00",
   "eventType":"PartnershipTypeRemoveNotification",
    "event": {
      "partnershipType" :
         {-- SEE PartnershipType RESOURCE SAMPLE --}
   }
}
```

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 PARTNERSHIP TYPE

LIST PARTNERSHIP TYPES

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

Description

This operation list partnership type 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 PartnershipType resources.

Request

```
GET {apiRoot}/partnershipType
Accept: application/json
```

Response

```
200

[
{
   "description": "This  partnership type ...",
   "href": "https:/host:port/tmf-api/partnershipTypeManagement/v2/partnershipType/4646",
   "id": "4646",
   "name": "Dream Partnership",
   "roleType": [
      [
         {
            "name": "ContentProvider",
            "agreementSpecification": [
               {
                  "name": "ContentLicenseAgreement",
                  "id": "33"
               }
            ]
         },
         {
            "name": "CloudProvider"
```

```
          },
          {
            "name": "Developer",
            "agreementSpecification": [
              {
                "name": "ProfitShareAgreement",
                "id": "32"
              }
            ]
          },
          {
            "name": "Tester"
          }
        ]
      ]
    }
]
```

RETRIEVE PARTNERSHIP TYPE

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

Description

This operation retrieves a partnership type 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 PartnershipType resource.

Request

```
GET {apiRoot}/partnershipType/4646
Accept: application/json
```

Response

```
200

{
  "description": "This  partnership type ...",
    "href": "https:/host:port/tmf-api/partnershipTypeManagement/v2/partnershipType/4646",
    "id": "4646",
    "name": "Dream Partnership",
    "roleType": [
      [
        {
          "name": "ContentProvider",
          "agreementSpecification": [
            {
              "name": "ContentLicenseAgreement",
              "id": "33"
            }
          ]
        },
        {
          "name": "CloudProvider"
        },
        {
          "name": "Developer",
          "agreementSpecification": [
            {
              "name": "ProfitShareAgreement",
              "id": "32"
            }
          ]
        },
        {
          "name": "Tester"
        }
      ]
    ]
  }
```

CREATE PARTNERSHIP TYPE

POST /partnershipType

Note: this operation is available only to ADMIN API users

Description

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
@baseType		
@schemaLocation		
@type		
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

Usage Samples

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

Request

```
POST {apiRoot}/partnershipType
Content-Type: application/json

{
   "name": "Dream Partnership"
}
```

Response

```
201

{
   "href": "https:/host:port/tmf-api/partnershipTypeManagement/v2/partnershipType/4646",
   "id": "4646",
   "name": "Dream Partnership"
}
```

PATCH PARTNERSHIP TYPE

PATCH /partnershipType/{id}

Note: this operation is available only to ADMIN API users

Description

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
@baseType	
@schemaLocation	
@type	
href	
id	

Usage Samples

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

Request

```
PATCH {apiRoot}/partnershipType/4646
Content-Type: application/merge-patch+json

{
   "name": "new name"
}
```

Response

```
201

{
   "description": "This  partnership type ...",
   "href": "https:/host:port/tmf-api/partnershipTypeManagement/v2/partnershipType/4646",
   "id": "4646",
   "name": "new name",
   "roleType": [
      [
         {
            "name": "ContentProvider",
            "agreementSpecification": [
               {
                  "name": "ContentLicenseAgreement",
                  "id": "33"
               }
            ]
         },
         {
            "name": "CloudProvider"
         },
         {
            "name": "Developer",
            "agreementSpecification": [
               {
                  "name": "ProfitShareAgreement",
                  "id": "32"
               }
            ]
         },
         {
            "name": "Tester"
         }
      ]
   ]
}
```

DELETE PARTNERSHIP TYPE

DELETE /partnershipType/{id}

Note: this operation is available only to ADMIN API users

Description

This operation deletes a partnership type entity.

Usage Samples

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

Request
DELETE {apiRoot}/partnershipType/42
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.

ACKNOWLEDGEMENTS

DOCUMENT HISTORY

VERSION HISTORY

Version Number	Date	Release led by:	Description
1.0	15/15/2016	Pierre Gauthier TM Forum pgauthier@tmforum.org Mariano Belaunde Orange Labs	First Release of the Document.
2.0	11-Jun-2018	Mariano Belaunde Orange Labs	Alignment with Guidelines 3.0
2.0.1	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

CONTRIBUTORS TO DOCUMENT

Mariano Belaunde	Orange
Pierre Gauthier	TM Forum