REST API Design Guidelines Part 2

Advanced guidelines for RESTful APIs polymorphism, extension patterns, depth and expand directive, entity RefOrValue

TMF630
Team Approved Date: 08-May-2020

<table>
<thead>
<tr>
<th>Release Status: Production</th>
<th>Status: TM Forum Approved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version: 4.0.1</td>
<td>IPR Mode: RAND</td>
</tr>
</tbody>
</table>
NOTICE

Copyright © TM Forum 2020. 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, 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.

TM Forum invites any TM Forum Member or any other party that believes it has patent claims that would necessarily be infringed by implementations of this TM Forum Standards Final Deliverable, to notify the TM Forum Team Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the TM Forum Collaboration Project Team that produced this deliverable.

The TM Forum invites any party to contact the TM Forum Team Administrator if it is aware of any claim of ownership of any patent claims that would necessarily be infringed by implementations of this TM Forum Standards Final Deliverable by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the TM Forum Collaboration Project Team that produced this TM Forum Standards Final Deliverable. TM Forum may include such claims on its website but disclaims any obligation to do so.

TM Forum takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this TM Forum Standards Final Deliverable or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on TM Forum’s procedures with respect to rights in any document or deliverable produced by a TM Forum Collaboration Project Team can be found on the TM Forum website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this TM Forum Standards Final Deliverable, can be obtained from the TM Forum Team Administrator. TM Forum makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.
Direct inquiries to the TM Forum office:

4 Century Drive, Suite 100
Parsippany, NJ 07054, USA
Tel No. +1 973 944 5100
Fax No. +1 973 998 7916
TM Forum Web Page: www.tmforum.org
Table of Contents

NOTICE ............................................................................................................................................. 2
Table of Contents ............................................................................................................................. 4
Executive Summary ......................................................................................................................... 5
Conventions ..................................................................................................................................... 6
1. Polymorphic Collections and Types ........................................................................................... 7
   1.1 Query using base collection ............................................................................................................ 10
   1.2 CUD operations using base collection ........................................................................................... 14
2. Extension patterns .................................................................................................................. 17
   2.1 Schema based extension ................................................................................................................ 17
   2.2 Schema based extension for the Characteristic of an entity ........................................................ 21
   2.3 State extension pattern .................................................................................................................. 29
3. Depth and Expand Directive .................................................................................................... 31
4. EntityRefOrValue pattern ........................................................................................................ 36
5. EntityRef pattern ..................................................................................................................... 45
6. Administrative Appendix ........................................................................................................ 51
   6.1 Document History ........................................................................................................................... 51
       6.0.1. Version History ......................................................................................................................... 51
       6.0.2. Release History ......................................................................................................................... 51
   6.2 Acknowledgments .................................................................................................................... 53
Executive Summary

This document, “REST API Design Guidelines Part 2” provides information for the development of TM Forum APIs using REST.

It provides recommendations and guidelines for the implementation of the Polymorphic Operations, extension, depth and expand, EntityRefOrValue related patterns.
Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].
1. **Polymorphic Collections and Types**

In the following section we will use a simple example based on a Logical Resource Inventory Management system to illustrate the concept of polymorphic collections and types.

Anonymous collections are used to group resource of the same type or resources related to the same base type.

For example, the logical resource collection in a resource management system can be used to group all the resources of type Tpe and Link.

Another example can be the party collection in a party management system that can be used to group all the individual and organization resources.

- The name of a type in the design guideline represents the collection linked to the type of resource.
  - For example, “Tpe” represents the collection of all Tpe resources while “Link” represents the collection of all the “Link” resources.
- @type is a reserved resource attribute like ID is.
  - For example, a link entity will have a “@type”=“Link” attribute.
- The name of the @type attribute is the same as the name of the resource.
- @type scoping is implicit to the declaration of the collection type.
- The @baseType of an entity can be used to represent the collection of all entities with the same base type.
  - For example, the “logicalResource” collection will scope both the “Tpe” and “Link” resources.
- The subtype can be used to represent the collection of all entities of a sub
For example, “Tpe” only represent the “Tpe” resources.

* When querying on a base collection type, all concrete resource representations compatible with the base type are returned.
* Objects of abstract resources are never returned (since by definition an abstract resource cannot be instantiated).

GET /logicalResource will only return resource representations for “Tpe” or “Link” if “LogicalResource” is an abstract type.

Example:

<table>
<thead>
<tr>
<th>Request</th>
</tr>
</thead>
</table>
| GET /resourceInventoryManagement/logicalResource?@type="Link" &@type="Tpe"  
Accept: application/json |

<table>
<thead>
<tr>
<th>Response</th>
</tr>
</thead>
</table>
| 200  
{}

```
{"id": "42",
 "href": "http://server:port/resourceInventoryManagement/logicalResource/42",
 "category": "",
 "@type": "Link",
 "@baseType": "LogicalResource",
 @schemaLocation": "//server:port/resourceInventoryManagement/schema/Link.yml",
 "lifecycleState": "Active",
 "resourceSpecification": {
  "id": "4",
},
 "relatedParty": [{
  "role": "owner",
  "id": "42",
```
All operations SHOULD be relative to the base types.

Other collections MAY be used at the discretion of the API Designer. In this case the operations exposed on the base collection may or may not be present on the derived collection.

Example:

- GET /resource/42?@type=Link
- GET/logicalResource/42?@type=Link
- GET /link/42

An entity MUST declare in the Header the alternative URI that can be used to identify it. E.g. /link/42 for a
logical resource of type link.
ID MUST always be relative to the base collection.

### 1.1 Query using base collection

The following example operation shows how to retrieve collections of any logical resource items.

For example we can retrieve both tpe and links using this operation.

<table>
<thead>
<tr>
<th>REQUEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET /resourceInventoryManagement/logicalResource?@type=&quot;Link&quot; &amp;@type=&quot;Tpe&quot;</td>
</tr>
<tr>
<td>Accept: application/json</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
</tr>
<tr>
<td>Content-Type: application/json</td>
</tr>
<tr>
<td>Content-Range: 1-2/2</td>
</tr>
<tr>
<td>[{</td>
</tr>
<tr>
<td>&quot;id&quot;: &quot;42&quot;,</td>
</tr>
<tr>
<td>&quot;href&quot;: &quot;<a href="http://server:port/resourceInventoryManagement/logicalResource/42">http://server:port/resourceInventoryManagement/logicalResource/42</a>&quot;,</td>
</tr>
<tr>
<td>&quot;category&quot;: &quot;&quot;,</td>
</tr>
<tr>
<td>&quot;@type&quot;: Link&quot;,</td>
</tr>
<tr>
<td>&quot;@basetype&quot;: &quot;LogicalResource&quot;,</td>
</tr>
<tr>
<td>&quot;@schemaLocation&quot;: &quot;/server:port/resourceInventoryManagement/schema/Link.yml&quot;,</td>
</tr>
<tr>
<td>&quot;lifecycleState&quot;: &quot;Active&quot;,</td>
</tr>
</tbody>
</table>
"resourceSpecification": {
    "id": "4",
},
"relatedParty": [{
    "role": "owner",
    "id": "42",
    "href": "http://serverlocation:port/customerManagement/individual/42"
}]
},
{
    "id": "44",
    "href": "http://server:port/resourceInventoryManagement/logicalResource/44",
    "category": "",
    "@type": "Tpe",
    "@basetype": "LogicalResource",
    "@schemaLocation": "//server:port/resourceInventoryManagement/schema/Tpe.yml",
    "lifecycleState": "Active",
    "resourceSpecification": {
        "id": "4",
    },
    "relatedParty": [{
        "role": "owner",
        "id": "42",
        "href": "http://serverlocation:port/customerManagement/individual/42"
    }]
}
Or using ID based filtering to extract specific resources.

<table>
<thead>
<tr>
<th>REQUEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET /resourceInventoryManagement/logicalResource?id=42&amp;id=44</td>
</tr>
<tr>
<td>Accept: application/json</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
</tr>
<tr>
<td>Content-Type: application/json</td>
</tr>
<tr>
<td>[</td>
</tr>
<tr>
<td>{</td>
</tr>
<tr>
<td>&quot;id&quot;: &quot;42&quot;,</td>
</tr>
<tr>
<td>&quot;href&quot;: &quot;/resourceInventoryManagement/logicalResource/42&quot;,</td>
</tr>
<tr>
<td>&quot;category&quot;: &quot;&quot;,</td>
</tr>
<tr>
<td>&quot;@type&quot;: Link&quot;,</td>
</tr>
<tr>
<td>&quot;@basetype&quot;: &quot;LogicalResource&quot;,</td>
</tr>
<tr>
<td>&quot;@schemaLocation&quot;: &quot;/resourceInventoryManagement/schema/Link.yml&quot;,</td>
</tr>
</tbody>
</table>
1.2 CUD operations using base collection

The following example shows how a concrete resource can be created using the base collection.

For example we can create a resource of type MSISDN in the inventory.

Request

POST /resourceInventoryManagement/logicalResource
Content-Type: application/json
{
    "value": "07865443255",
    "@type": "MSISDN"
}
The following example shows how a concrete resource can be updated using the base collection.

For example we can change the lifecycleState of a logical resource of type MSISDN in the inventory.

<table>
<thead>
<tr>
<th>Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATCH /resourceInventoryManagement/logicalResource/42</td>
</tr>
<tr>
<td>Content-Type: application/json-patch+json</td>
</tr>
<tr>
<td>{</td>
</tr>
<tr>
<td>&quot;path&quot;: &quot;/lifecycleState&quot;,</td>
</tr>
<tr>
<td>&quot;value&quot;: &quot;Reserved&quot;,</td>
</tr>
<tr>
<td>&quot;op&quot;: &quot;replace&quot;</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
</tr>
<tr>
<td>{ Similar JSON as in GET response with state changed}</td>
</tr>
</tbody>
</table>

The following example shows how a concrete resource can be updated using the base collection.

For example, we can delete a logical resource with id 42.
<table>
<thead>
<tr>
<th>Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE /resourceInventoryManagement/logicalResource/42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
</tr>
</tbody>
</table>
2. **Extension patterns**

An extension is considered conformant or valid if it does not invalidate the core characteristics of an API and does not contradict the API Design Guidelines.

Various extensions mechanisms are supported:

- Extending the basic schema of an entity like Product to create a subclass in that case all the mandatory attributes and relationships of the base schema should be present in the extension.
- Adding characteristics to define run time extension
- Adding relationships

It is not recommended to extend an TMF Open API with opaque name value pairs. Instead schema defined characteristic MUST be used.

![Diagram](image)

2.1 **Schema based extension**

In the following section we will use a simple example to illustrate the concept of adding simple or complex characteristics to define run time extensions.

@schemaLocation is a reserved attribute like @type or id and it provides a link to the schema describing a REST resource.

An extension schema MUST include all the attributes of the base schema.
The following example shows how a physical resource can be extended with new attributes (operatingState attribute) at run-time:

```
{
  "id": "45",
  "href": "/resourceInventoryManagement/physicalResource/45",
  "publicIdentifier": "07467223333",
  "@type": "Equipment",
  "@baseType": "PhysicalResource",
  "@schemaLocation": "/resourceInventoryManagement/schema/Equipment.yml",
  "category": "Category 1",
  "lifecycleState": "Active",
  "manufactureDate": "2007-04-12",
  "serialNumber": "123456745644",
  "versionNumber": "11",
  "operatingState": "Working",
  "resourceSpecification": {
    "id": "6",
    "href": "/resourceCatalogManagement/resourceSpecification/6",
    "@type": "PhysicalResourceSpecification"
  },
  "relatedParty": [{
    "role": "Manufacturer",
```
All resources defined within an API MAY be extended, therefore attributes @type, @baseType and @schemaLocation MUST be supported optionally in the structure definition of all resources.

Those resources that are not expected to be extended are not required to explicitly include the attributes as part of the resource structure definition in the API spec documentation.

As an example, the following resources will be considered as expected to be extended
• Party: In SID they are defined as either Individual or Organization.

• PartyRole: In SID they are defined as either Customer, Partner, Employee, ...

• GeographicAddress: In SID it can be either AustralianPropertyAddress, JapanesePropertyAddress, AustralianRuralAddress...

• Characteristic: The structure of the characteristics object may depend on the implementation or the element whose characteristics are to be described.

As part of the API Data model the resources that have been identified as expected to be extended in implementations must include parameters @type, @baseType and @schemaLocation explicitly within the structure defined for those resources.

In Swagger 2.0 and 3.0 allOf SHOULD be used to extend an object.

Here is an example of Equipment.yml schema for the example above:

```
..............

PhysicalResource:
  title: PhysicalResource
  type: object
  properties:
    id:
      type: integer

..............

Equipment:
  title: Equipment
  type: object
  allOf:
    - $ref: "#/definitions/PhysicalResource"
```

2.2 Schema based extension for the Characteristic of an entity

Assuming that Characteristic/CharacteristicSpec pattern is used, the following example shows how we can extend an API at run time by adding simple or complex characteristics:

```json
{
    "id": "45",
    "href": "/resourceInventoryManagement/physicalResource/45",
    "publicIdentifier": "07467223333",
    "@type": "Equipment",
    "@baseType": "PhysicalResource",
    "@schemaLocation": "http://server:port/resourceInventoryManagement/schema/Equipment.yml",
    "category": "Category 1",
    "lifecycleState": "Active",
    "manufactureDate": "2007-04-12",
    "serialNumber": "123456745644",
    "versionNumber": "11",
    "operatingState": "Working"
}
```
"resourceSpecification": {
  "id": "6",
  "href": "/resourceCatalogManagement/resourceSpecification/6",
  "@type": "PhysicalResourceSpecification"
},
"resourceCharacteristic": [{
  "name": "physicalPort",
  "valueType": "Object",
  "value": {
    "@type": "PhysicalPort",
    "@schemaLocation": "http://host:port/schema/PhysicalPort.yml",
    "name": "LAN Port",
    "isActive": true
  }
},
{ "name": "color",
  "value": "red"
}],
"resourceRelationship": [{}]}
"type": "requires",

"resource": {
  "id": "46",
  "href": "/resourceInventoryManagement/logicalResource/46"
},

"resourceRelationshipCharacteristic": [{
  "name": "priority",
  "value": 2
},
{
  "name": "accuracy",
  "valueType": "Object",
  "value": {
    "@type": "Accuracy",
    "@schemaLocation": "http://server:port/resourceInventoryManagement/schema/Accurancy.yml",
    "unit": "second",
    "amount": "5"
  }
}]

"relatedParty": [{
  "type": "requires",
  "resource": {
    "id": "46",
    "href": "/resourceInventoryManagement/logicalResource/46"
  },
  "resourceRelationshipCharacteristic": [{
    "name": "priority",
    "value": 2
  },
  {
    "name": "accuracy",
    "valueType": "Object",
    "value": {
      "@type": "Accuracy",
      "@schemaLocation": "http://server:port/resourceInventoryManagement/schema/Accurancy.yml",
      "unit": "second",
      "amount": "5"
    }
  }]
}]},
"role": "Manufacturer",
"id": "43",
"href": "/PartyManagement/individual/43"
}],
"resourceAttachment": [{
  "id": "/documentManagement/document/123"
}],
"note": [{
  "text": "something about this resource"
}],
"place": {
  "id": "1979",
  "href": "/genericCommon/place/1979",
  "name": "Main Office",
  "role": "default delivery"
}
}
In the above example, ResourceCharacteristics class is a list of simple and complex characteristics where:

- **name**: A string. Name of the characteristic.
- **value**: A string (StringorObject). Value of the characteristic which can be simple or complex (object).
- **@schemaLocation**: A string. A link to the schema describing this characteristic.
- **@type**: A string. (Class) type of the characteristic.
- **valueType**: A string. A kind of value that the characteristic can take on, such as numeric, text and so forth.

For complex characteristics, the name of the characteristic is the name of the object while the value is a complex object and the schema describing the object is provided as a link in the @schemaLocation attribute.

The following example shows how a ResourceFunctionSpec can be extended using the same mechanism as above:

```json
{
    "id": "43",
    "href": "/catalogManagement/ResourceFunctionSpec/43",
    "name": "Firewall",
    "description": "This resource specification ...",
    "@type": "ResourceFunctionSpec",
    "@schemaLocation": "https://host:port/catalogManagement/schema/ResourceFunctionSpec.yaml",
    "@baseType": "LogicalResourceSpec",
    "version": "3.2",
    "validFor": {
        "startDateTIme": "2017-08-06T00:00",
        "endDateTIme": "2018-03-07T00:00"
    },
    "lastUpdate": "2017-08-09T00:00",
    "lifecycleStatus": "Active",
    "isBundle": false,
    "category": "Security",
}
```
"targetResourceSchema": {
   "@type": "ResourceFunction",
   "@schemaLocation": "https://host:port/catalogManagement/schema/ResourceFunction.yml",
},
"attachment": [
   {
      "description": "This attachment ...",
      "href": "/documentManagement/attachment/22",
      "id": "22",
      "type": "Document",
      "url": "http://xxxxx"
   }
],
"relatedParty": [
   {
      "id": "8406",
      "href": "/partyManagement/organization/8406",
      "role": "Supplier",
      "name": "Firewall Express",
      "validFor": {
         "startDateTime": "2017-08-05T00:00",
         "endDateTime": "2018-03-07T00:00"
      }
   }
],
"resourceSpecCharacteristic": [
   {
      "name": "OperatingSystem",
      "description": "This resource spec characteristic ...",
      "valueType": "String",
      "@valueSchemaLocation": "",
      "configurable": true,
      "validFor": {
         "startDateTime": "2017-08-12T00:00",
         "endDateTime": "2018-03-07T00:00"
      }
   },
   {
      "@type": "ResourceSpecCharacteristic",
      "minCardinality": 0,
      "maxCardinality": 1,
      "isUnique": true,
   }
]
"regex": "",  
"extensible": false,  
"resourceSpecCharRelationship": [  
{  
"type": "string",  
"name": "OperatingSystem",  
"id": "4690",  
"href": "/catalogManagement/resourceSpecification/4690",  
"@type": "ResourceFunctionSpec",  
"validFor": {  
"startDateDateTime": "2017-08-11T00:00",  
"endDateDateTime": "2018-03-07T00:00"  
}  
},  
"resourceSpecCharacteristicValue": [  
{  
"isDefault": true,  
"value": "Android KitKat",  
"validFor": {  
"startDateDateTime": "2017-08-06T00:00",  
"endDateDateTime": "2018-03-07T00:00"  
}  
}  
],  
"ScalabilityChar":  
"description": "Scalability parameters for this resource function",  
"valueType": "CapabilityScalable",  
"@valueSchemaLocation": "https://host:port/catalogManagement/schema/CapabilityScalable.yml",  
"configurable": true,  
"validFor": {  
"startDateDateTime": "2017-08-17T00:00",  
"endDateDateTime": "2018-03-12T00:00"  
},  
"@type": "ResourceSpecCharacteristic",  
"minCardinality": 0,  
"maxCardinality": 1,  
"isUnique": true,
"extensible": true,
"resourceSpecCharRelationship": [],
"resourceSpecCharacteristicValue": [
{
    "valueType": "Object",
    "value": {
        "minInstances": 1,
        "maxInstances": 1000,
    },
    "isDefault": true,
    "validFor": {
        "startDateTIme": "2017-08-17T00:00",
        "endDateTIme": "2018-03-12T00:00"
    },
    "@type": "CapabilityScalable",
    "@schemaLocation": "https://host:port/catalogManagement/schema/CapabilityScalable.yml",
}
],
"resourceSpecRelationship": [
{
    "type": "AdjacencyPair",
    "id": "2053",
    "href": "/catalogManagement/physicalBlackBoxSpec/2053",
    "name": "SWI.1",
    "validFor": {
        "startDateTIme": "2017-08-08T00:00",
        "endDateTIme": "2018-03-07T00:00"
    }
}
],
"feature": [
{
    "id": "8782",
    "name": "IPv4Addressing",
    "isBundle": false,
    "validFor": {
        "startDateTIme": "2017-08-16T00:00",
        "endDateTIme": "2018-03-12T00:00"
    },
    "isEnabled": true
### 2.3 State extension pattern

Many instances represented in TMForum Open-APIs have a long-lived "state" and hence a "lifecycle". The collection of lifecycle states are typically described in an enumerated type that is applied to the status attribute. The rules of use and extension are as follows:

1. The enumeration of states are considered "normative" but not comprehensive. They must not be removed from the specification, or semantically redefined by the implementation, but they can be added to - both in breadth (additional states) and depth (additional sub-states).
2. The transitions between states are "illustrative". While the state-transition diagram provides a helpful graph, a compliant implementation is free to transition between states as it wishes - though obviously it would be helpful to document any variation for potential consumers.
3. There is a distinction between the operational state of the Resource (such as up, down, error) and its administrative state (planned, in-operation, retired). These should not be confused or merged as they are independent of each other.

As a simple example, a ServiceProblem be acknowledged, inProgress or resolved - so it's status attribute might be described as:

```json
{
    "type": "string",
    "enum": ["acknowledged", "inProgress", "resolved"]
}
```
Extending with new states:
Implementations might consider adding new states by extending the enumeration accordingly:

```json
{
    "type": "string",
    "enum": ["acknowledged", "rejected", "inProgress", "resolved", "closed"]
}
```

Extending with sub-states:
Instances of the resource should be expected to be in one of the "macro" states, but might chose to extend them to add more detail. Although a ServiceProblem is "inProgress" it might be waiting on an external action or information. A dotted notation can be used to describe appropriate sub-states such as `inProgress.held` or `inProgress.pending`.

```json
{
    "type": "string",
    "enum": ["acknowledged", "rejected", "inProgress.held", "inProgress.pending", "resolved", "closed"]
}
```

As a result, consumers should always consider the possible presence of sub-states, and search for (say) `inProgress\(...)?$` rather than "inProgress" (such as in JSON-Path: @.status == "inProgress\(...)?$"), which would match "inProgress", "inProgress.pending" (with .pending as a regex-group) but not "inProgressPending".
3. Depth and Expand Directive

A common use case is to query a resource and “dereference” or “embed” the hyperlinked resource representations. This is achieved by issuing a GET with the DEPTH directive.

The GET /api/<resource>/?DEPTH=n operation is used to expand inline the referenced data up to the level of the specified depth level.

The “depth” is used to expand the level of referenced (related) entities such as bundled offerings and “target” references of relations.

If depth=0, then all referenced entities just RootEntity objects which contain only ID and @type (may contain name and href).

If depth = 1, then the first level of related target entities is expanded. References are replaced with the actual loaded entities with the corresponding level of details.

If depth = N it expands reference objects of related entities recursively. The last level contains only the references.

NOTE: expanding should not lead to infinite recursion if there is a circular reference. In such case the object is expanded once when first met and at other places remains a reference.

“expand” parameter is used to narrow down (filter) expanding of the referenced objects.

If expand is not specified, then by default all the paths should be expanded. If the expand parameter is specified, only those paths should be expanded according to depth parameter.

If depth is not specified, then the default value will depend on the expand parameter. If expand parameter is set, then depth is 1. If not set, then 0.

In the following example the productOffering.productSpecification is expanded up to the level of depth specified by the depth parameter:

```
GET /catalogManagement/productOffering/23/?depth=3&expand=productOffering.productSpecification
Accept: application/json
```

```json
{
  "id": "23",
}
```
"href": "/catalogManagement/productOffering/23",
"version": "2.0",
"lastUpdate": "2013-04-19T16:42:23-04:00",
"name": "Sensor mini",
"description": "A wireless sensor for small garden",
"isBundle": "false",
"lifecycleStatus": "Active",
"validFor": {
    "startDateTime": "2013-04-19T16:42:23-04:00",
    "endDateTime": "2013-06-19T00:00:00-04:00"
},
"category": [
    {
        "id": "14",
        "href": "/catalogManagement/category/14",
        "version": "2.0",
        "name": "Wireless sensors"
    }
],
"channel": [
    {
        "id": "13",
        "href": "/marketSales/channel/13",
        "name": "IOT Corner"
    }
],
"place": [
    {
        "id": "12",
        "href": "/marketSales/place/12",
        "name": "France"
    }
],
"serviceLevelAgreement": {
    "id": "28",
    "href": "/slaManagement/serviceLevelAgreement/28",
    "name": "Standard SLA"
},
"productSpecification": [
    {
        "id": "13",
        "href": "/catalogManagement/productSpecification/13",
        "version": "2.0",
        "name": "Sensor mini"
    }
]
"href": "/catalogManagement/productSpecification/13",
"version": "2.0",
"name": "wireless sensor mini",
"productSpecCharacteristic": [
  {
    "id": "34",
    "name": "Colour",
    "description": "Colour",
    "valueType": "string",
    "configurable": "true",
    "validFor": {
      "startDateDateTime": "2013-04-19T16:42:23-04:00",
      "endDateDateTime": ""
    },
    "ProductSpecCharacteristicValue": [
      {
        "valueType": "string",
        "default": "false",
        "value": "Black",
        "unitOfMeasure": "",
        "valueFrom": "",
        "valueTo": "",
        "validFor": {
          "startDateDateTime": "2013-04-19T16:42:23-04:00",
          "endDateDateTime": ""
        }
      },
      {
        "valueType": "string",
        "default": "false",
        "value": "White",
        "unitOfMeasure": "",
        "valueFrom": "",
        "valueTo": "",
        "validFor": {
          "startDateDateTime": "2013-04-19T16:42:23-04:00",
          "endDateDateTime": ""
        }
      }
    ]
  }
]
"endDateTime": ""
}
]
}
"productOfferingPrice": [
{
"id": "15",
"href": "/catalogManagement/productOfferingPrice/15",
"name": "Sale Price",
"description": "Sale price",
"validFor": {
 "startDateTime": "2013-04-19T16:42:23-04:00",
"endDateTime": "2013-06-19T00:00:00-04:00"
},
"priceType": "one time",
"unitOfMeasure": "",
"price": {
 "taxIncludedAmount": "12.00",
"dutyFreeAmount": "10.00",
"taxRate": "20.00",
"currencyCode": "EUR"
},
"recurringChargePeriod": "",
"productOfferPriceAlteration": {
"id": "15",
"href": "/catalogManagement/productOfferPriceAlteration/15",
"name": "Shipping Discount",
"description": "One time shipping discount",
"validFor": {
 "startDateTime": "2013-04-19T16:42:23-04:00",
"endDateTime": ""
},
"priceType": "One Time discount",
"id": "15",
"href": "/catalogManagement/productOfferPriceAlteration/15/15",
"name": "Shipping Discount",
"description": "One time shipping discount",
"validFor": {
 "startDateTime": "2013-04-19T16:42:23-04:00",
"endDateTime": ""
},
"priceType": "One Time discount",}
"unitOfMeasure": "",
"price": {
    "percentage": "100%"
},
"recurringChargePeriod": "",
"applicationDuration": "",
"priceCondition": "apply if total amount of the order is greater than 300.00"
}
4. **EntityRefOrValue pattern**

This pattern is useful when there is a need to pass to the server an object or a list of objects by value if the object(s) does not exist yet or by reference when the object(s) already exist on server side.

For EntityRefOrValue all attributes are optional.

If it is a Ref, id and href are mandatory but if it is Value then only the mandatory parameters from the Entity referenced are required (id and href can be also added if the reference is used to modify the contents of an existing entity).

To apply this pattern to a plain entity (for instance Product), the corresponding ref entity need to be defined in the API data model (in our example ProductRef).

The EntityRef MUST include necessarily an “id” and an “href” attributes, but other attributes from the plain entity may be included.

In the API data model:

- Any class may be used as a RefOrValue. Therefore we should use the plain class in the API data model. E.g if we need to support RefOrValue pattern for Product class, we should link in the model to the Product class. A constraint on the relationship (refOrValue) indicates whether the pattern is actually used or not used.

Example: In the following example the described pattern can be seen in the relation between entities Product and Place, Product and ProductRelationship and OrderItem and Product.
REST API Design Guidelines Part 2

In Swagger implementation:

- Swagger 2.0 we should allow for a less strongly type solution. We will create the swagger schema as if it was a value and it will be up to the implementation to implement ref or value. All the parameters in the value class should be optional.

Example: In the following example the product class is defined as if it was a value.

```
"definitions": {
```
"Product": {

  "description": "Configure the product characteristics (only configurable characteristics and necessary only if a non default value is selected) and/or identify the product that needs to be modified/deleted.",

  "type": "object",

  "properties": {

    "id": {

      "description": "",

      "type": "string"

    },

    "href": {

      "description": "",

      "type": "string"

    },

    "name": {

      "description": "",

      "type": "string"

    },

    "place": {

      "description": "",

      "type": "array",

      "items": {

        "description": "",

        "type": "string"}

    }

  }

}
"$ref": "/definitions/Place"

],

"characteristic": {
  "description": "",
  "type": "array",
  "items": {
    "$ref": "/definitions/ProductCharacteristic"
  }
},

"relatedParty": {
  "description": "",
  "type": "array",
  "items": {
    "$ref": "/definitions/RelatedPartyRef"
  }
},

...

"productRelationship": {
  "description": "",
  "type": "array",
  "items": {
    "$ref": "/definitions/ProductRelationship"
• In Swagger 3.0 we SHOULD make use of oneOf between value class and ref class.

Example: In the following example the Product is passed using oneOff.

......

"ProductRefOrValue": {
   "oneOf": [
      {
         "$ref": "/components/schemas/Product"
      }
   ]
}
In this example we provide another example of how RefOrValue pattern is defined in Swagger 3.0 for relatedPartyRefOrValue:

```
......
"relatedPartyRefOrValue": {
    "oneOf": [
        {
            "$ref": "/components/schemas/Party"
        },
        {
            "$ref": "/components/schemas/PartyRef"
        },
        {
            "$ref": "/components/schemas/PartyRole"
        },
        {
            "$ref": "/components/schemas/PartyRoleRef"
        }
    ]
}
```
Here is a JSON example of ProductRefOrValue usage in Shopping Cart API where product is passed by value:

Request

GET /shoppingCart/v1/shoppingCart/5074
Accept: application/json

Response

200

{
"href": "/shoppingCart/v1/shoppingCart/1203",
"id": "1203",
"validFor": {
"startDateTime": "2017-03-27T00:00",
"endDateTime": "2017-10-24T00:00"
},
"contactMedium": [
{
"type": "email",
"characteristic": {
"email": "JackSmith@mail.com"
}
}
],
"cartTotalPrice": [
{
"description": "Total Recurring Price.",
"name": "Monthly Price",
"priceType": "recurring",
"recurringChargePeriod": "monthly",
"email": "JackSmith@mail.com"
}
]
"price": {
  "dutyFreeAmount": {
    "money": {
      "unit": "EUR",
      "value": 29
    }
  },
  "taxIncludedAmount": {
    "money": {
      "unit": "EUR",
      "value": 31.9
    },
    "taxRate": 10
  }
},
"priceAlteration": [
  {
    "applicationDuration": 3,
    "description": "First 3 month get 50% off",
    "name": "3 month for half",
    "priceCondition": "new customer",
    "priceType": "recurring",
    "recurringChargePeriod": "monthly",
    "validFor": {
      "startDateTime": "2017-03-26T00:00",
      "endDateTime": "2017-10-24T00:00"
    },
    "price": {
      "percentage": 50
    }
  }
],
"cartItem": [
  {
    "action": "add",
    "id": "8307",
    "quantity": 1,
    "status": "Active",
    "note": {
      "author": "Mr Smith",
      "date": "2017-03-28T00:00",
      "text": "Please wrap with double bag"
},
"product": {
  "name": "Talk Simple 25",
  "place": [
    {
      "href": "/placeManagement/place/7396",
      "id": "7396",
      "name": "Main Store",
      "role": "default delivery"
    }
  ],
  "productOffering": {
    "href": "/productCatalog/productOffering/14277",
    "id": "14277",
    "name": "Talk Simple 25"
  }
}
}
5. **EntityRef pattern**

This pattern is useful when there is a need to pass to the server an object or a list of objects by reference when the object(s) already exists on server side.

The EntityRef MUST include necessarily an “id” and an “href” attributes, but other attributes from the plain entity MAY be included like name, @type,@schemaLocation,validFor, role.

Use @type if required. If present, @type indicates the presence of a generalization (sub classing of Entity as per generic extension patterns).

The presence of @type is generally not needed for EntityRef objects.

Here is an example of where EntityRef pattern has been used for relatedPartyRef, ChannelRef, BillingAccountRef, PaymentRef.

```json
{
    "id":"11",
    "href":"/productOrdering/productOrder/11.org",
    "externalId":"Telco01",
    "priority":"1",
    "description":"Order illustration UC1",
    "category":"Residential",
    "state":"Completed",
    "orderDate":"2017-11-03T08:46:47.945Z",
    "completionDate":"2017-11-03T08:46:47.945Z",
    "requestedStartDate":"2017-11-03T08:46:47.945Z",
    "requestedCompletionDate":"2017-11-03T08:46:47.945Z",
    "expectedCompletionDate":"2017-11-03T08:46:47.945Z",
    "notificationContact":"jean.pontus@tmforum.com",
    "@baseType":"ProductOrder",
    "@type":"ProductOrder",
    "ChannelRef":{
        "id":"1",
        "name":"store",
        "@referredType":"Store"
    },
    "note":{
        "text":"Order to illustrate UC1",
        "date":"2017-11-03T08:46:47.945Z",
        "author":"Jean-Lyuc Tymen"
    }
}
```
"relatedParty": [ 
{
   "id": "11",
   "href": "/tmfAPI/customer/11.org",
   "name": "Joe Doe",
   "role": "customer",
   "@referredType": "Customer",
}, 
{
   "id": "11",
   "href": "/tmfAPI/partyManagement/individual/11.org",
   "name": "Joe Doe",
   "@referredType": "Individual",
}, 
{
   "id": "12",
   "href": "/tmfAPI/customer/12.org",
   "name": "Arthur Pence",
   "role": "Seller",
   "@referredType": "Customer",
}
],
"billingAccount": {
   "id": "BA1513",
   "href": "/tmfAPI/billingAccount/BA1513.org",
   "name": "BA01"
},
"orderItem": [ 
{
   "id": "100",
   "action": "add",
   "state": "Completed",
   "quantity": 1,
   "productOffering": {
      "id": "14277",
      "name": "TMF25"
   },
   "orderItem": [ 
      {
         "id": "110",
         "action": "add",
         "state": "Completed",
         "quantity": 1,
      }
   ]
}
"itemPrice": [  {
    "name": "AccessFee",
    "description": "Access Fee",
    "priceType": "oneTime",
    "price": {
        "taxIncludedAmount": {
            "value": 0.99,
            "unit": "Euros"
        }
    }
  },
  "productOffering": {
    "id": "14305",
    "name": "TMF Mobile Telephony"
  },
  "product": {
    "characteristic": [
      {
        "name": "TEL_MSISDN",
        "value": "27457958"
      }
    ],
    "relatedParty": [
      {
        "id": "11",
        "href": "http://tmfAPI/customer/11.org",
        "name": "Joe Doe",
        "role": "user",
        "@referredType": "Customer"
      }
    ],
    "productSpecification": {
      "id": "14307",
      "version": "1",
      "name": "Mobile Telephony"
    }
  },
  "payment": [
    {
      "id": "2365",
      "name": "shop payment cash #2365"
    }
  ]


```json
{
  "id":"120",
  "action":"add",
  "state":"Completed",
  "quantity":1,
  "itemPrice":{
    "name":"monthly price",
    "description":"TMF Tariff Plan monthly fee",
    "priceType":"recurring",
    "recurringChargePeriod":"monthly",
    "price":{
      "taxIncludedAmount":{
        "value":20,
        "unit":"Euros"
      },
      "taxRate":18.6
    },
    "priceAlteration":{
      "name":"20Discount3Months",
      "description":"discount 20% for 3 months",
      "priceType":"recurring",
      "recurringChargePeriod":"monthly",
      "applicationDuration":"3",
      "priority":1,
      "price":{
        "percentage":20
      }
    }
  },
  "productOffering":{
    "id":"14344",
    "name":"TMF TariffPlan"
  },
  "product":{
    "productSpecification":{
      "id":"14395",
      "name":"TMF TariffPlan"
    }
  }
}
```


```json
},

"orderItemRelationship":[
{
  "id":"110",
  "type":"reliesOn"
}
],

"itemTerm":[
{
  "name":"12MonthsCommitment",
  "description":"12 Months commitment",
  "duration":{
    "value":12,
    "unit":"string"
  }
}
],

{
  "id":"130",
  "action":"add",
  "state":"Completed",
  "quantity":1,
  "productOffering":{
    "id":14354,
    "name":"CoverageOptions"
  },
  "product":{
    "characteristic":[
      {
        "name":"CoverageOption",
        "value":"National"
      }
    ],
    "productSpecification":{
      "id":14353,
      "name":"coverage"
    }
  },
  "orderItemRelationship":[
    {
      "id":110,
      "type":"reliesOn"
    }
  ]
}
```
6. Administrative Appendix

This Appendix provides additional background material about the TM Forum and this document. In general, sections may be included or omitted as desired; however, a Document History must always be included.

6.1 Document History

6.0.1. Version History

This section records the changes between this and the previous document version as it is edited by the team concerned. Note: this is an incremental number which does not have to match the release number and used for change control purposes only.

<table>
<thead>
<tr>
<th>Version</th>
<th>Date Modified</th>
<th>Modified by:</th>
<th>Description of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>23-Nov-2014</td>
<td>Pierre Gauthier, TM Forum</td>
<td>Description e.g. first issue of document</td>
</tr>
<tr>
<td>1.1</td>
<td>12-Mar-2015</td>
<td>Alicja Kawecki, TM Forum</td>
<td>Updated cover, footer and Notice to reflect TM Forum Approved status</td>
</tr>
<tr>
<td>2.0</td>
<td>08-Nov-2017</td>
<td>Peter Norbury</td>
<td>Update to Notice and change Document number</td>
</tr>
<tr>
<td>2.0.1</td>
<td>12-Dec-2017</td>
<td>Adrienne Walcott</td>
<td>Formatting/style edits prior to publishing</td>
</tr>
<tr>
<td>2.0.2</td>
<td>20-Mar 2018</td>
<td>Adrienne Walcott</td>
<td>Updated to reflect TM Forum Approved Status</td>
</tr>
<tr>
<td>4.0.0</td>
<td>14-May-2020</td>
<td>Alan Pope</td>
<td>Minor edits prior to publication</td>
</tr>
<tr>
<td>4.0.1</td>
<td>20-Jul-2020</td>
<td>Adrienne Walcott</td>
<td>Updated to reflect TM Forum Approved Status</td>
</tr>
</tbody>
</table>

6.0.2. Release History

This section records the changes between this and the previous Official document release. The release number is the ‘Marketing’ number which this version of the document is first being assigned to.

<table>
<thead>
<tr>
<th>Release Number</th>
<th>Date Modified</th>
<th>Modified by:</th>
<th>Description of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.5.0</td>
<td>08-Nov-2017</td>
<td>Peter Norbury</td>
<td>First release</td>
</tr>
<tr>
<td>17.5.1</td>
<td>20-Mar-2018</td>
<td>Adrienne Walcott</td>
<td>Updated to reflect TM Forum Approved Status</td>
</tr>
<tr>
<td>Pre-production</td>
<td>14-May-2020</td>
<td>Alan Pope</td>
<td>Minor edits prior to publication</td>
</tr>
<tr>
<td>Release Number</td>
<td>Date Modified</td>
<td>Modified by:</td>
<td>Description of changes</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------</td>
<td>--------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Production</td>
<td>20-Jul-2020</td>
<td>Adrienne Walcott</td>
<td>Updated to reflect TM Forum Approved Status</td>
</tr>
</tbody>
</table>
6.2 Acknowledgments

This document was prepared by the members of the TM Forum API team:

- Pierre Gauthier, TM Forum, Editor and Team Leader
- Nicoleta Stoica, Vodafone, Author
- Luis Velarde, Telefonica, Contributor
- Kamal Maghsoudlou, Ericsson, Contributor
- Mariano Belaunde, Orange, Contributor
- Ludovic Robert, Orange, Contributor