

# TM Forum Specification

## REST API Design Guidelines Part 4

*Advanced guidelines for RESTful APIs lifecycle management, common tasks*

**TMF630**

**Team Approved Date: 08-May-2020**

<b>Release Status: Production</b>	<b>Status: TM Forum Approved</b>
<b>Version 4.0.1</b>	<b>IPR Mode: RAND</b>

## 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 a 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](http://www.tmforum.org)

# List of Contents

<b>Notice</b> .....	<b>2</b>
<b>List of Contents</b> .....	<b>4</b>
<b>List of Figures</b> .....	<b>5</b>
<b>List of Tables</b> .....	<b>6</b>
<b>Executive Summary</b> .....	<b>7</b>
<b>Conventions</b> .....	<b>8</b>
<b>1. Export and Import Data Tasks</b> .....	<b>9</b>
1.1 Export Job Resource .....	9
1.2 Import Job Resource .....	10
1.3 Creating Export Jobs.....	11
1.4 Creating Import Jobs .....	13
1.5 Query Export Jobs.....	13
1.6 Query Import Jobs .....	14
1.7 Export Job Completion Notification .....	14
1.8 Import Job Completion Notification.....	15
<b>2. Entity Versioning and Lifecycle Management</b> .....	<b>17</b>
2.1 Query all versioned resources .....	18
2.2 Query a specific versioned resource .....	19
2.3 Query current version of a resource .....	20
2.4 Create new version of a resource.....	20
2.5 Modify an existing version of a resource .....	22
2.6 Role based Access Control .....	23
<b>3. Administrative Appendix</b> .....	<b>24</b>
3.1 Document History .....	24
3.1.1 <i>Version History</i> .....	24
3.1.2 <i>Release History</i> .....	25
3.2 Acknowledgments.....	25

## List of Figures

N/A

## List of Tables

N/A

## Executive Summary

This document, “REST API Design Guidelines Part 4” provides information for the development of TM Forum APIs using REST.

It provides recommendations and guidelines for the implementation of the Common Export and Import Tasks and the Entity Lifecycle Management 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. Export and Import Data Tasks

This section describes the common tasks used for exporting and importing resources representations to files.

Two common task resources are defined:

- ImportJob used to import resources from a File
- ExportJob used to export resources to a File

## 1.1 Export Job Resource

An ExportJob resource represents a TASK used to export resources to a File

The ExportJob resource supports the following properties:

Attribute name	Description
query	Used to scope the exported data (identical to GET filter construct using target ID as base)  "query": "type=productOffering&version=2.0"
path	URL of the root resource acting as the source for streaming content to the file specified by the ExportJob  ../catalogManagement/catalog/42  i.e., the relative path to the exportJob when it was created can also be used
content-type	The format of the exported data .By default "application/json"
status	notstarted, running, succeeded, failed
url	URL of the File containing the data to be exported

Attribute name	Description
	<p>a file URL, which is of the form</p> <p><code>file://host/path</code></p> <p>where host is the fully qualified domain name of the system on which the path is accessible, and path is a hierarchical directory path of the form <code>directory/directory/.../name</code></p>
<code>completionDate</code>	Date at which the Job was completed.
<code>creationDate</code>	Date at which the Job was created.
<code>errorLog</code>	Reason for Failure

## 1.2 Import Job Resource

An ImportJob resource represent a TASK used to import resources from a File

The ImportJob resource supports the following properties:

Attribute name	Description
<code>content-type</code>	The format of the imported data .By default “application/json”
<code>path</code>	<p>URL of the root resource where the content of the file specified by the ImportJob must be applied</p> <p><code>../catalogManagement/catalog/42</code></p>

Attribute name	Description
	i.e. the relative path to the importJob when it was created
status	notstarted, running, succeeded, failed
url	URL of the File containing the data to be imported a file URL, which is of the form  file://host/path  where host is the fully qualified domain name of the system on which the path is accessible, and path is a hierarchical directory path of the form directory/directory/.../name
completionDate	Date at which the Job was completed.
creationDate	Date at which the Job was created.
errorLog	Reason for Failure if status is failed

### 1.3 Creating Export Jobs

ExportJob Tasks are created as resources. The ExportJob is attached to a specific resource acting as the root for the collection of resources to be streamed to a File.

An ExportJob can be attached to a specific Catalog in a Catalog application or may be attached to the Product Offerings within a Catalog.

- ../catalogManagement/catalog/42/exportJob  
Export all the resources within the Catalog 42
- ../catalogManagement/catalog/42/productOffering/exportJob  
Export all the ProductOffering resources

For example:

REQUEST
POST catalogManagement/catalog/{10}/exportJob Content-type: application/json  <pre>{</pre>
RESPONSE
201 Content-Type: application/json Location: /api/catalogManagement/exportJob/54  <pre>{   "id": "54",   "href": "/catalogManagement/exportJob/54",   "status": "running",   "path": "catalogManagement/catalog/10",   "content-type": "application/json",   "errorLog": "",   "creationDate": "2013-04-19T16:42:23-04:00",   "completionDate": "",   "url": "ftp://ftp.myCatalog.com/productCatalog/54" }</pre>

## 1.4 Creating Import Jobs

ImportJob Tasks are created as resources. The ImportJob may be attached to the URL of the root resource where the content of the file specified by the ImportJob will be applied.

For example, to apply the content of the import file to the catalog 10:

REQUEST
POST catalogManagement/catalog/{10}/importJob Content-type: application/json  <pre>{</pre>
RESPONSE
201 Content-Type: application/json Location: /api/catalogManagement/importJob/554  <pre>{   "id": "554",   "href": "/catalogManagement/importJob/554",   "status": "running",   "path": "catalogManagement/catalog/10",   "content-type": "application/json",   "errorLog": "",   "creationDate" : "2013-04-19T16:42:23-04:00",   "completionDate" : "",   "url": "ftp://ftp.myCatalog.com/productCatalog/partner54" }</pre>

## 1.5 Query Export Jobs

ExportJob resources can be found under the API/exportJob collection and may be retrieved using the normal GET constructs.

For example:

GET API/catalogManagement/exportJob/{10} Accept: application/json
<b>RESPONSE</b>
<pre>{   "id": "10",   "status": "Succeeded" }</pre>

## 1.6 Query Import Jobs

ImportJob resources can be found under the API/importJob collection and may be retrieved using the normal GET constructs.

GET /catalogManagement/importJob/{25} Accept: application/json
<b>RESPONSE</b>
<pre>{   "id": "25",   "status": "Succeeded" }</pre>

## 1.7 Export Job Completion Notification

This event provides notification that an export task has been completed.

For example:

REQUEST
<pre>POST /client/listener Accept: application/json  {   "event": {     "id": "54",     "href": "/catalogManagement/exportJob/54",     "status": "running",     "path": "catalogManagement/catalog/{10}",     "content-type": "application/json",     "errorLog": "",     "creationDate": "2013-04-19T16:42:23-04:00",     "completionDate": "",     "url": "ftp://ftp.myCatalog.com/productCatalog/54"   },   "eventType": "exportJobCompleted" }</pre>

## 1.8 Import Job Completion Notification

This event provides a notification that an import task has been completed.

REQUEST
<pre>POST /client/listener Accept: application/json  {   "event": {     "id": "554",     "href": "/catalogManagement/importJob/554",     "status": "running",</pre>

```
    "path": "catalogManagement/catalog/10" ,  
    "content-type": "application/json",  
    "errorLog": "",  
    "creationDate" : "2013-04-19T16:42:23-04:00",  
    "completionDate" : "2013-04-19T16:42:23-04:15",  
    "url": "ftp://ftp.myCatalog.com/productCatalog/partner54"  
  },  
  "eventType": "importJobCompleted"  
}
```



## 2. Entity Versioning and Lifecycle Management

In Product Lifecycle Management there is a requirement to distinguish between entities existing with different PLM version numbers and accessible via different ACL mechanisms.

For example, the same Product Offerings may exist in a Catalog but with different version numbers.

It may be possible for an administrator to see all the existing versions or for a partner to see only a subset of all the existing versions.

The entity version number is not dependent on the version number of the API. For example, in PLM the same API (running at a specific version number) may be used to retrieve entities with different PLM version numbers.

In order to distinguish resources representing entities running with different version numbers and accessible through the same API version the following directive can be used `/id:(version=x)` and the version attribute is added to each entity.

```
{
  "id": "42",
  "href": "/catalogManagement/productOffering/42",
  "version": "1.0",
  "lastUpdate": "2013-04-19T16:42:23-04:00",
  "name": "Virtual Storage Medium",
  "description": "Virtual Storage Medium",
  "isBundle": "true",
  "lifecycleStatus": "Active",
  .....
}
```

Note that the resources in this case may have the same ID but may be distinguished by the inclusion of the version number in their ID i.e. `/42:(version=1.0)`, `/42:(version=2.0)`.

In the following examples we will assume that two versions of the VirtualStorage Product Offer exist in the Product Catalog an Inactive and Active version respectively version 1.0 and version 2.0.

## 2.1 Query all versioned resources

Admin user of Catalog have access to all the versions of the resources while non admin users have by default access to only the latest version of the entities in the Catalog.

Users with different roles may have access to different versions of the entities in the catalog. For example, user A may have access to only the version 1.0 of the entities while user B may have access to version 1.0 and version 2.0.

For example, the following request on the admin endpoint will return all the versioned resources matching a specific ID.

REQUEST
GET api/admin/catalogManagement/productOffering/?id=VirtualStorage Accept: application/json
RESPONSE
200 Content-Type: application/json  <pre> [[   {     "id": " VirtualStorage ",     "href": "/catalogManagement/productOffering/VirtualStorage",     "version": "1.0",     "lastUpdate": "2013-04-19T16:42:23-04:00",     "name": "Virtual Storage Medium",     "description": "Virtual Storage Medium",     "isBundle": "true",     "lifecycleStatus": "Inactive",     .....   },   {     "id": " VirtualStorage ",     "href": "/catalogManagement/productOffering/ VirtualStorage ",     "version": "2.0",     "lastUpdate": "2013-04-19T16:42:23-04:00",     "name": "Virtual Storage Medium",     "description": "Virtual Storage Medium",     "isBundle": "true",     "lifecycleStatus": "Active",     .....   } ]]           </pre>

]

## 2.2 Query a specific versioned resource

In general, a non admin API user only has visibility to the latest version number or visibility to a subset of versioned resources.

It may be possible for an admin API user to retrieve a resource with a specific version number by using an ID and versioning filtering criteria.

REQUEST
GET api/admin/catalogManagement/productOffering/?id=VirtualStorage&version=1.0 Accept: application/json
RESPONSE
200 Content-Type: application/json  <pre>[   {     "id": "42",     "href": "/catalogManagement/productOffering/42",     "version": "1.0",     "lastUpdate": "2013-04-19T16:42:23-04:00",     "name": "Virtual Storage Medium",     "description": "Virtual Storage Medium",     "isBundle": "true",     "lifecycleStatus": "Active",     .....   } ]</pre>

### 2.3 Query current version of a resource

By default, only the most current version is returned (for admin and non admin).

REQUEST
GET api/admin/catalogManagement/productOffering/VirtualStorage Accept: application/json
RESPONSE
200 Content-Type: application/json  <pre>{   "id": "42",   "href": "/catalogManagement/productOffering/42",   "version": "2.0",   "lastUpdate": "2013-04-19T16:42:23-04:00",   "name": "Virtual Storage Medium",   "description": "Virtual Storage Medium",   "isBundle": "true",   "lifecycleStatus": "Active",   .... }</pre>

### 2.4 Create new version of a resource

POST is used to create a new version of a resource.

The constraint is that the version numbers for the resource having the same ID must differ.

REQUEST
POST catalogManagement/productOffering Content-type: application/json

```
{
  "id": "VirtualStorage",
  "version": "3.0",
  "name": "Virtual Storage Medium",
  "description": "Virtual Storage Medium",
  "isBundle": "true",
  "lifecycleStatus": "Active",
  "validFor": {
    "startDateTime": "2013-04-19T16:42:23-04:00",
    "endDateTime": "2013-06-19T00:00:00-04:00"
  },
  ...
}
```

**RESPONSE**

201

Content-Type: application/json

```
{
  "id": "VirtualStorage",
  "href": "/catalogManagement/productOffering/42",
  "version": "3.0",
  "lastUpdate": "2013-04-19T16:42:23-04:00",
  "name": "Virtual Storage Medium",
  "description": "Virtual Storage Medium",
  "isBundle": "true",
  "lifecycleStatus": "Active",
  "validFor": {
    "startDateTime": "2013-04-19T16:42:23-04:00",
    "endDateTime": "2013-06-19T00:00:00-04:00"
  },
  ...
}
```

## 2.5 Modify an existing version of a resource

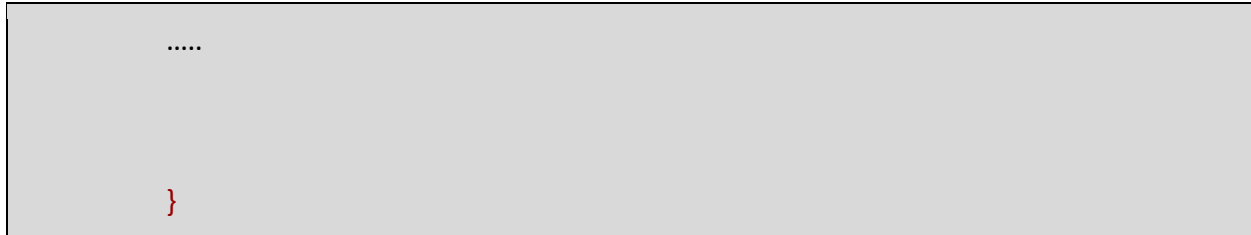
By default, PATCH or PUT will be acting only on the latest version of the Resource. For example, PATCH `.../productOffering/VirtualStorage` will only update the VirtualStorage ProductOffering at Version 2.0 (which is the most current).

To update a specific version of an entity the `(Version=X)` directive is added to the ID (i.e. `/id:(version=x)`).

Note that this capability is only available to API users having the proper authorizations to change the catalog entities with specific version numbers.

For example, to change the VirtualStorage versioned at 1.0 we could use `/productOffering/VirtualStorage(Version=1.0)`

REQUEST
<pre>PATCH /catalogManagement/productOffering/VirtualStorage(Version=1.0) Content-type: application/json-patch+json  {   "lifecycleStatus": "Active" }</pre>
RESPONSE
<pre>201 Content-Type: application/json  {   "id": "VirtualStorage",   "href": "/catalogManagement/productOffering/VirtualStorage",   "version": "1.0",   "lifecycleStatus": "Active",</pre>



## 2.6 Role based Access Control

The user presents their credentials for authentication

If the credentials are valid

1. The user is given access to the catalog
2. As defined by their role(s)
3. As defined by their access rights
4. As defined by the access type: CRUD, discover
5. As defined by the pre-defined filter

For example, if they issue a get on a catalog that a party has no access they get an error response

Or if they try to modify an area of the catalog but do not have Write Access they get an error response

We anticipate that the OAUTH2 or Open ID Connect will be used as the authorization APIs and that ACL are established between authorized parties with regards to the content of the Catalog (i.e., GET but also enable of update operations on specific entities).

## 3. 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.

### 3.1 Document History

#### 3.1.1 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.

Version Number	Date Modified	Modified by:	Description of changes
1.0	23-Nov-2014	Pierre Gauthier TM Forum	Description e.g. first issue of document
1.1.1	12/03/2015	Alicja Kawecki, TM Forum	Updated cover, footer and Notice to reflect TM Forum Approved status
2.0.0	08-Nov-2017	Peter Norbury	Update to Notice and changed document number
2.0.1	12-Dec-2017	Adrienne Walcott	Formatting/style edits prior to publishing
2.0.2	20-Mar-2018	Adrienne Walcott	Updated to reflect TM Forum Approved Status
4.0.0	14-May-2020	Alan Pope	Minor edits prior to publication
4.0.1	20-Jul-2020	Adrienne Walcott	Updated to reflect TM Forum Approved Status



### 3.1.2 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.

Release Number	Date Modified	Modified by:	Description of changes
17.5.0	08-Nov-2017	Peter Norbury	first release of document
17.5.1	20-Mar-2018	Adrienne Walcott	Updated to reflect TM Forum Approved Status
Pre-production	14-May-2020	Alan Pope	Updated to v4.0
Production	20-Jul-2020	Adrienne Walcott	Updated to reflect TM Forum Approved Status

### 3.2 Acknowledgments

This document was prepared by the members of the TM Forum <<team name>> team:

- Pierre Gauthier, TM Forum, **Editor** and Team Leader
- Maxime Delon Orange
- Veronique Mauneau Orange
- John Morey Ciena