Resource Catalog Management
Conformance Profile

Document Number: <###>

Document Version: V1.0

Date: August, 2017

Document Status: Draft

# NOTICE

Copyright © TeleManagement Forum 2017. All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to TM FORUM, except as needed for the purpose of developing any document or deliverable produced by a TM FORUM Collaboration Project Team (in which case the rules applicable to copyrights, as set forth in the [TM FORUM IPR Policy](http://www.tmforum.org/IPRPolicy/11525/home.html), must be followed) or as required to translate it into languages other than English.

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

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

Direct inquiries to the TM Forum office:

4 Century Drive, Suite 100

Parsippany, NJ 07054 USA

Tel No.  +1 973 944 5100

Fax No.  +1 973 944 5110

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

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

# Table of Contents

[NOTICE 2](#_Toc490587243)

[Table of Contents 3](#_Toc490587244)

[Introduction - API DESCRIPTION 6](#_Toc490587245)

[RESOURCE MODEL CONFORMANCE 7](#_Toc490587246)

[API MANDATORY AND OPTIONAL RESOURCES 7](#_Toc490587247)

[Resource Catalog MANDATORY AND OPTIONAL ATTRIBUTES 7](#_Toc490587248)

[Resource Category MANDATORY AND OPTIONAL ATTRIBUTES 8](#_Toc490587249)

[Resource Candidate MANDATORY AND OPTIONAL ATTRIBUTES 9](#_Toc490587250)

[Resource Specification MANDATORY AND OPTIONAL ATTRIBUTES 10](#_Toc490587251)

[Logical Resource Spec MANDATORY AND OPTIONAL ATTRIBUTES 11](#_Toc490587252)

[Physical Resource Spec MANDATORY AND OPTIONAL ATTRIBUTES 12](#_Toc490587253)

[Import Job MANDATORY AND OPTIONAL ATTRIBUTES 13](#_Toc490587254)

[Export Job MANDATORY AND OPTIONAL ATTRIBUTES 14](#_Toc490587255)

[NOTIFICATION MODEL CONFORMANCE 16](#_Toc490587256)

[API MANDATORY AND OPTIONAL NOTIFICATIONS 16](#_Toc490587257)

[API OPERATIONS CONFORMANCE 18](#_Toc490587258)

[API MANDATORY AND OPTIONAL OPERATIONS 18](#_Toc490587259)

[API GET OPERATION CONFORMANCE 20](#_Toc490587260)

[/resourceCatalog?fields=...&{filtering} 20](#_Toc490587261)

[/resourceCatalog/{id}?fields=...&{filtering} 20](#_Toc490587262)

[/resourceCategory?fields=...&{filtering} 20](#_Toc490587263)

[/resourceCategory/{id}?fields=...&{filtering} 21](#_Toc490587264)

[/resourceCandidate?fields=...&{filtering} 21](#_Toc490587265)

[/resourceCandidate/{id}?fields=...&{filtering} 21](#_Toc490587266)

[/resourceSpecification?fields=...&{filtering} 21](#_Toc490587267)

[/resourceSpecification/{id}?fields=...&{filtering} 21](#_Toc490587268)

[/logicalResourceSpec?fields=...&{filtering} 21](#_Toc490587269)

[/logicalResourceSpec/{id}?fields=...&{filtering} 21](#_Toc490587270)

[/physicalResourceSpec?fields=...&{filtering} 21](#_Toc490587271)

[/physicalResourceSpec/{id}?fields=...&{filtering} 21](#_Toc490587272)

[/importJob?fields=...&{filtering} 22](#_Toc490587273)

[/importJob/{id}?fields=...&{filtering} 22](#_Toc490587274)

[/exportJob?fields=...&{filtering} 22](#_Toc490587275)

[/exportJob/{id}?fields=...&{filtering} 22](#_Toc490587276)

[API POST OPERATION CONFORMANCE 23](#_Toc490587277)

[/resourceCatalog 23](#_Toc490587278)

[/resourceCategory 23](#_Toc490587279)

[/resourceCandidate 24](#_Toc490587280)

[/resourceSpecification 25](#_Toc490587281)

[/logicalResourceSpec 26](#_Toc490587282)

[/physicalResourceSpec 27](#_Toc490587283)

[/importJob 29](#_Toc490587284)

[/exportJob 29](#_Toc490587285)

[API PUT OPERATION CONFORMANCE 31](#_Toc490587286)

[API PATCH OPERATION CONFORMANCE 32](#_Toc490587287)

[/resourceCatalog/{id} 32](#_Toc490587288)

[/resourceCategory/{id} 33](#_Toc490587289)

[/resourceCandidate/{id} 34](#_Toc490587290)

[/resourceSpecification/{id} 35](#_Toc490587291)

[/logicalResourceSpec/{id} 36](#_Toc490587292)

[/physicalResourceSpec/{id} 37](#_Toc490587293)

[API DELETE OPERATION CONFORMANCE 39](#_Toc490587294)

[/resourceCatalog/{id} 39](#_Toc490587295)

[/resourceCategory/{id} 39](#_Toc490587296)

[/resourceCandidate/{id} 39](#_Toc490587297)

[/resourceSpecification/{id} 39](#_Toc490587298)

[/logicalResourceSpec/{id} 39](#_Toc490587299)

[/physicalResourceSpec/{id} 39](#_Toc490587300)

[/importJob/{id} 39](#_Toc490587301)

[/exportJob/{id} 39](#_Toc490587302)

[API CONFORMANCE TEST SCENARIOS 40](#_Toc490587303)

[ResourceCandidate resource TEST CASES 41](#_Toc490587304)

[ResourceSpecification resource TEST CASES 44](#_Toc490587305)

[LogicalResourceSpec resource TEST CASES 48](#_Toc490587306)

[PhysicalResourceSpec resource TEST CASES 52](#_Toc490587307)

[ImportJob resource TEST CASES 56](#_Toc490587308)

[ExportJob resource TEST CASES 58](#_Toc490587309)

[Release History 59](#_Toc490587310)

[Contributors to Document 60](#_Toc490587311)

# Introduction - API DESCRIPTION

This document is the REST API Conformance profile for the Resource Catalog Management REST API v2.0.

# 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 |
| ResourceCatalog | O |  |
| ResourceCategory | O |  |
| ResourceCandidate | M |  |
| ResourceSpecification | M |  |
| LogicalResourceSpec | M |  |
| PhysicalResourceSpec | M |  |
| ImportJob | M |  |
| ExportJob | M |  |

Only Mandatory resources are included in the basic certification.

## Resource Catalog MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| name | M (for resource creation)O (otherwise) |  |
| description | O |  |
| @type | M (in response messages)O (otherwise) |  |
| @schemaLocation | O |  |
| @baseType | O |  |
| version | O |  |
| validFor | O |  |
| lastUpdate | M (in response messages)O (otherwise) |  |
| lifecycleStatus | M (in response messages)O (otherwise) |  |
| relatedParty | O |  |
| category | O |  |

## Resource Category MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| name | M (for resource creation)O (otherwise) |  |
| description | O |  |
| @type | M (in response messages)O (otherwise) |  |
| @schemalLocation | O |  |
| @baseType | O |  |
| version | O |  |
| validFor | O |  |
| lifecycleStatus | M (in response messages)O (otherwise) |  |
| lastUpdate | M (in response messages)O (otherwise) |  |
| parentId | O |  |
| isRoot | M (in response messages)O (otherwise) |  |
| category | O |  |
| resourceCandidate | O |  |
| relatedParty | O |  |

## Resource Candidate MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| name | M (for resource creation)O (otherwise) |  |
| description | O |  |
| @type | M (in response messages)O (otherwise) |  |
| @schemaLocation | O |  |
| @baseType | O |  |
| version | O |  |
| validFor | O |  |
| lastUpdate | M (in response messages)O (otherwise) |  |
| lifecycleStatus | M (in response messages)O (otherwise) |  |
| category | O |  |
| resourceSpecification | O |  |

## Resource Specification MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| name | M (for resource creation)O (otherwise) |  |
| description | O |  |
| @type | M (for resource creation)O (otherwise) | Attribute non patchable |
| @schemaLocation | O |  |
| @baseType | O |  |
| version | O |  |
| validFor | O |  |
| lastUpdate | M (in response messages)O (otherwise) | Attribute non patchable |
| lifecycleStatus | M (in response messages)O (otherwise) |  |
| isBundle | M (in response messages)O (otherwise) |  |
| category | O |  |
| targetResourceSchema | O |  |
| feature | O |  |
| attachment | O |  |
| relatedParty | O |  |
| resourceSpecCharacteristic | O |  |
| resourceSpecRelationship | O |  |

## Logical Resource Spec MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| name | M (for resource creation)O (otherwise) |  |
| description | O |  |
| @type | M (for resource creation)O (otherwise) | Attribute non patchable |
| @schemaLocation | O |  |
| @baseType | O |  |
| version | O |  |
| validFor | O |  |
| lastUpdate | M (in response messages)O (otherwise) | Attribute non patchable |
| lifecycleStatus | M (in response messages)O (otherwise) |  |
| isBundle | M (in response messages)O (otherwise) |  |
| category | O |  |
| targetResourceSchema | O |  |
| feature | O |  |
| attachment | O |  |
| relatedParty | O |  |
| resourceSpecCharacteristic | O |  |
| resourceSpecRelationship | O |  |

## Physical Resource Spec MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| name | M (for resource creation)O (otherwise) |  |
| description | O |  |
| @type | M (for resource creation)O (otherwise) | Attribute non patchable |
| @schemaLocation | O |  |
| @baseType | O |  |
| version | O |  |
| validFor | O |  |
| lastUpdate | M (in response messages)O (otherwise) | Attribute non patchable |
| lifecycleStatus | O |  |
| isBundle | M (in response messages)O (otherwise) |  |
| category | O |  |
| model | M (in response messages)O (otherwise) |  |
| part | O |  |
| sku | O |  |
| vendor | M (in response messages)O (otherwise) |  |
| place | O |  |
| targetResourceSchema | O |  |
| feature | O |  |
| attachment | O |  |
| relatedParty | O |  |
| resourceSpecCharacteristic | O |  |
| resourceSpecRelationship | O |  |

## Import Job MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| contentType | O |  |
| path | O |  |
| status | O |  |
| url | M (for resource creation)O (otherwise) |  |
| completionDate | O |  |
| creationDate | O |  |
| errorLog | O |  |

## Export Job MANDATORY AND OPTIONAL ATTRIBUTES

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

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Mandatory or Optional** | **Comments** |
| id | M (in response messages)O (otherwise) | Generated by the server |
| href | M (in response messages)O (otherwise) | Url for the created resource |
| query | O |  |
| path | O |  |
| contentType | O |  |
| status | O |  |
| url | M (for resource creation)O (otherwise) |  |
| completionDate | O |  |
| creationDate | O |  |
| errorLog | O |  |

# NOTIFICATION MODEL CONFORMANCE

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

## API MANDATORY AND OPTIONAL NOTIFICATIONS

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

|  |  |  |
| --- | --- | --- |
| Notification Name | Mandatory or Optional | Comments |
| ResourceCatalogCreationNotification | O |  |
| ResourceCatalogRemoveNotification | O |  |
| ResourceCatalogBatchNotification | O |  |
| ResourceCategoryCreationNotification | O |  |
| ResourceCategoryRemoveNotification | O |  |
| ResourceCandidateCreationNotification | O |  |
| ResourceCandidateRemoveNotification | O |  |
| ResourceSpecificationCreationNotification | O |  |
| ResourceSpecificationRemoveNotification | O |  |
| LogicalResourceSpecCreationNotification | O |  |
| LogicalResourceSpecRemoveNotification | O |  |
| PhysicalResourceSpecCreationNotification | O |  |
| PhysicalResourceSpecRemoveNotification | O |  |

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

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

|  |  |  |
| --- | --- | --- |
| Uniform API Operation | Mandatory/Optional | Comments |
| GET  | M for all resources | GET must be used to retrieve a representation of a resource |
| POST  | M for resources: ResourceCatalogResourceCategoryResourceCandidateResourceSpecificationLogicalResourceSpecPhysicalResourceSpecImportJobExportJob | POST must be used to create a new resource |
| PUT  | O for all resources:  |  |
| PATCH  | M for resources: ResourceCatalogResourceCategoryResourceCandidateResourceSpecificationLogicalResourceSpecPhysicalResourceSpec | PATCH must be used to partially update a resource |
| DELETE  | M for resources: ResourceCatalogResourceCategoryResourceCandidateResourceSpecificationLogicalResourceSpecPhysicalResourceSpecImportJobExportJob | 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

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

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

## /resourceCatalog?fields=...&{filtering}

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

## /resourceCatalog/{id}?fields=...&{filtering}

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

## /resourceCategory?fields=...&{filtering}

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

## /resourceCategory/{id}?fields=...&{filtering}

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

## /resourceCandidate?fields=...&{filtering}

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

## /resourceCandidate/{id}?fields=...&{filtering}

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

## /resourceSpecification?fields=...&{filtering}

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

## /resourceSpecification/{id}?fields=...&{filtering}

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

## /logicalResourceSpec?fields=...&{filtering}

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

## /logicalResourceSpec/{id}?fields=...&{filtering}

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

## /physicalResourceSpec?fields=...&{filtering}

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

## /physicalResourceSpec/{id}?fields=...&{filtering}

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

## /importJob?fields=...&{filtering}

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

## /importJob/{id}?fields=...&{filtering}

This operation retrieves an import job entity.
Attribute selection is mandatory for all first level attributes.
Filtering on sub-resources is optional for all compliance levels.

## /exportJob?fields=...&{filtering}

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

## /exportJob/{id}?fields=...&{filtering}

This operation retrieves an export job entity.
Attribute selection is mandatory for all first level attributes.
Filtering on sub-resources is optional for all compliance levels.

# API POST OPERATION CONFORMANCE

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

|  |  |  |
| --- | --- | --- |
| POST | M |  |
| Status Code 201 | M  |  |
| Other Status Codes | NA | Status error code like 400, 404, 409 as applicable |

## /resourceCatalog

This operation creates a resource catalog entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a ResourceCatalog, 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 |  |  |
| @type | ResourceCatalog |  |
| @schemaLocation |  |  |
| @baseType | Catalog |  |
| version |  |  |
| validFor |  |  |
| lastUpdate |  |  |
| lifecycleStatus |  |  |
| relatedParty |  |  |
| category |  |  |

## /resourceCategory

This operation creates a resource category entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a ResourceCategory, 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 |  |  |
| @type |  |  |
| @schemalLocation |  |  |
| @baseType |  |  |
| version |  |  |
| validFor |  |  |
| lifecycleStatus |  |  |
| lastUpdate |  |  |
| parentId |  |  |
| isRoot |  |  |
| category |  |  |
| resourceCandidate |  |  |
| relatedParty |  |  |

## /resourceCandidate

This operation creates a resource candidate entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a ResourceCandidate, 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 |  |  |
| @type |  |  |
| @schemaLocation |  |  |
| @baseType |  |  |
| version |  |  |
| validFor |  |  |
| lastUpdate |  |  |
| lifecycleStatus |  |  |
| category |  |  |
| resourceSpecification |  |  |

## /resourceSpecification

This operation creates a resource specification entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a ResourceSpecification, 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 |  |
| @type |  |

|  |  |  |
| --- | --- | --- |
| **Non Mandatory Attributes** | **Default Value** | **Rule** |
| description |  |  |
| @schemaLocation |  |  |
| @baseType |  |  |
| version |  |  |
| validFor |  |  |
| lastUpdate |  |  |
| lifecycleStatus |  |  |
| isBundle |  |  |
| category |  |  |
| targetResourceSchema |  |  |
| feature |  |  |
| attachment |  |  |
| relatedParty |  |  |
| resourceSpecCharacteristic |  |  |
| resourceSpecRelationship |  |  |

**Additional Rules**

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

|  |  |
| --- | --- |
| **Context** | **Mandatory Sub-Attributes** |
| attachment | name |
| relatedParty | id or href |
| resourceSpecRelationship | type, id or href |
| feature | name or id |

## /logicalResourceSpec

This operation creates a logical resource spec entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a LogicalResourceSpec, 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 |  |
| @type |  |

|  |  |  |
| --- | --- | --- |
| **Non Mandatory Attributes** | **Default Value** | **Rule** |
| description |  |  |
| @schemaLocation |  |  |
| @baseType |  |  |
| version |  |  |
| validFor |  |  |
| lastUpdate |  |  |
| lifecycleStatus |  |  |
| isBundle |  |  |
| category |  |  |
| targetResourceSchema |  |  |
| feature |  |  |
| attachment |  |  |
| relatedParty |  |  |
| resourceSpecCharacteristic |  |  |
| resourceSpecRelationship |  |  |

**Additional Rules**

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

|  |  |
| --- | --- |
| **Context** | **Mandatory Sub-Attributes** |
| attachment | name |
| relatedParty | id or href |
| resourceSpecRelationship | type, id or href |
| feature | name or id |

## /physicalResourceSpec

This operation creates a physical resource spec entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a PhysicalResourceSpec, 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 |  |
| @type |  |
| model |  |
| vendor |  |

|  |  |  |
| --- | --- | --- |
| **Non Mandatory Attributes** | **Default Value** | **Rule** |
| description |  |  |
| @schemaLocation |  |  |
| @baseType |  |  |
| version |  |  |
| validFor |  |  |
| lastUpdate |  |  |
| lifecycleStatus |  |  |
| isBundle |  |  |
| category |  |  |
| part |  |  |
| sku |  |  |
| place |  |  |
| targetResourceSchema |  |  |
| feature |  |  |
| attachment |  |  |
| relatedParty |  |  |
| resourceSpecCharacteristic |  |  |
| resourceSpecRelationship |  |  |

**Additional Rules**

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

|  |  |
| --- | --- |
| **Context** | **Mandatory Sub-Attributes** |
| attachment | name |
| relatedParty | id or href |
| resourceSpecRelationship | type, id or href |
| feature | name or id |

## /importJob

This operation creates an import job entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a ImportJob, 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** |
| url |  |

|  |  |  |
| --- | --- | --- |
| **Non Mandatory Attributes** | **Default Value** | **Rule** |
| contentType |  |  |
| path |  |  |
| status |  |  |
| completionDate |  |  |
| creationDate |  |  |
| errorLog |  |  |

## /exportJob

This operation creates an export job entity.

**Mandatory and Non Mandatory Attributes**

The following tables provides the list of mandatory and non mandatory attributes when creating a ExportJob, 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** |
| url |  |

|  |  |  |
| --- | --- | --- |
| **Non Mandatory Attributes** | **Default Value** | **Rule** |
| query |  |  |
| path |  |  |
| contentType |  |  |
| status |  |  |
| completionDate |  |  |
| creationDate |  |  |
| errorLog |  |  |

# API PUT OPERATION CONFORMANCE

Since PUT operation is optional and not included in the certification this is not applicable in this conformance document.

# API PATCH OPERATION CONFORMANCE

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

|  |  |  |
| --- | --- | --- |
| PATCH | M |  |
| Status Code 200 | M  |  |
| Other Status Codes | NA | Status error code like 400, 404, 409 as applicable |

## /resourceCatalog/{id}

This operation allows partial updates of a resource catalog 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.
Notice that patching is possible only for 'admin' API users.

|  |  |
| --- | --- |
| **Patchable Attributes** | **Rule** |
| name |  |
| description |  |
| @schemaLocation |  |
| version |  |
| validFor |  |
| lifecycleStatus |  |
| relatedParty |  |
| category |  |

|  |  |
| --- | --- |
| **Non Patchable Attributes** | **Rule** |
| id |  |
| href |  |
| name |  |
| @type |  |
| @baseType |  |
| lastUpdate |  |

## /resourceCategory/{id}

This operation allows partial updates of a resource category 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.
Notice that patching is possible only for 'admin' API users.

|  |  |
| --- | --- |
| **Patchable Attributes** | **Rule** |
| name |  |
| description |  |
| @schemalLocation |  |
| @baseType |  |
| version |  |
| validFor |  |
| lifecycleStatus |  |
| parentId |  |
| isRoot |  |
| category |  |
| resourceCandidate |  |
| relatedParty |  |

|  |  |
| --- | --- |
| **Non Patchable Attributes** | **Rule** |
| id |  |
| href |  |
| @type |  |
| lastUpdate |  |

## /resourceCandidate/{id}

This operation allows partial updates of a resource candidate 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.
Notice that patching is possible only for 'admin' API users.

|  |  |
| --- | --- |
| **Patchable Attributes** | **Rule** |
| name |  |
| description |  |
| @schemaLocation |  |
| @baseType |  |
| version |  |
| validFor |  |
| lifecycleStatus |  |
| category |  |
| resourceSpecification |  |

|  |  |
| --- | --- |
| **Non Patchable Attributes** | **Rule** |
| id |  |
| href |  |
| @type |  |
| lastUpdate |  |

## /resourceSpecification/{id}

This operation allows partial updates of a resource specification 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.
Notice that patching is possible only for 'admin' API users.

|  |  |
| --- | --- |
| **Patchable Attributes** | **Rule** |
| name |  |
| description |  |
| @schemaLocation |  |
| @baseType |  |
| version |  |
| validFor |  |
| lifecycleStatus |  |
| isBundle |  |
| category |  |
| targetResourceSchema |  |
| feature |  |
| attachment |  |
| relatedParty |  |
| resourceSpecCharacteristic |  |
| resourceSpecRelationship |  |

|  |  |
| --- | --- |
| **Non Patchable Attributes** | **Rule** |
| id |  |
| href |  |
| lastUpdate |  |
| @type |  |

## /logicalResourceSpec/{id}

This operation allows partial updates of a logical resource spec 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.
Notice that patching is possible only for 'admin' API users.

|  |  |
| --- | --- |
| **Patchable Attributes** | **Rule** |
| name |  |
| description |  |
| @schemaLocation |  |
| @baseType |  |
| version |  |
| validFor |  |
| lifecycleStatus |  |
| isBundle |  |
| category |  |
| targetResourceSchema |  |
| feature |  |
| attachment |  |
| relatedParty |  |
| resourceSpecCharacteristic |  |
| resourceSpecRelationship |  |

|  |  |
| --- | --- |
| **Non Patchable Attributes** | **Rule** |
| id |  |
| href |  |
| lastUpdate |  |
| @type |  |

## /physicalResourceSpec/{id}

This operation allows partial updates of a physical resource spec 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.
Notice that patching is possible only for 'admin' API users.

|  |  |
| --- | --- |
| **Patchable Attributes** | **Rule** |
| name |  |
| description |  |
| @schemaLocation |  |
| @baseType |  |
| version |  |
| validFor |  |
| lifecycleStatus |  |
| isBundle |  |
| category |  |
| model |  |
| part |  |
| sku |  |
| vendor |  |
| place |  |
| targetResourceSchema |  |
| feature |  |
| attachment |  |
| relatedParty |  |
| resourceSpecCharacteristic |  |
| resourceSpecRelationship |  |

|  |  |
| --- | --- |
| **Non Patchable Attributes** | **Rule** |
| id |  |
| href |  |
| lastUpdate |  |
| @type |  |

# API DELETE OPERATION CONFORMANCE

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

|  |  |  |
| --- | --- | --- |
| DELETE | M |  |
| Status Code 204 | M  |  |
| Other Status Codes | NA | Status error code like 400, 404, 409 as applicable |

## /resourceCatalog/{id}

This operation deletes a resource catalog entity.

## /resourceCategory/{id}

This operation deletes a resource category entity.

## /resourceCandidate/{id}

This operation deletes a resource candidate entity.

## /resourceSpecification/{id}

This operation deletes a resource specification entity.

## /logicalResourceSpec/{id}

This operation deletes a logical resource spec entity.

## /physicalResourceSpec/{id}

This operation deletes a physical resource spec entity.

## /importJob/{id}

This operation deletes an import job entity.

## /exportJob/{id}

This operation deletes an export job 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}/catalogManagement/v2**, where {serverRoot} defines the certification endpoint.

## ResourceCandidate resource TEST CASES

|  |  |  |
| --- | --- | --- |
| **Test Case ID** | **Title** | **Description** |
| TC\_ResourceCandidate\_POST\_N1 | Create Resource Candidate  | Pre-requisite: it should be ResourceSpecification exist as result of TC\_ResourceSpecification\_POST\_N1 1. Send POST message to {apiRoot}/resourceCandidate/ with all mandatory parameters and supported optional parameters to create a new ResourceCandidate such as:

|  |
| --- |
| { “name”: “<anytext>”, “description”: “<anytext>”, “lifecycleStatus”: “Active”, "@type": "ResourceCandidate", “validFor”: { “startDateTime”: ”<any value with correct datetime format>”, “endDateTime”: “<any value with correct datetime format>” }, "resourceSpecification": { "id": “<any valid id value which identifying an existing resource specification>”,  "href": “<any valid href value which addressing an existing resource specification>”,  "name": “<anytext>”  }} |

1. Verify the response with code 201, location header set to url of the created resource and body including full representation of the created resource
 |
| TC\_ResourceCandidate\_POST\_N2 | Create Resource Candidate with missing mandatory parameter | Pre-requisite: None1. Send POST message to {apiRoot}/resourceCandidate/ without mandatory parameters like name to create a new ResourceCandidate
2. Verify that server rejects the request by sending the proper response code/error message
 |
| TC\_ResourceCandidate\_GET\_N3 | Search for Resource Candidates- no filtering | Pre-requisite: create multiple ResourceCandidate as per TC\_ResourceCandidate\_POST\_N11. Send a GET message to /{apiRoot}/resourceCandidate
2. Verify the response with code 200,
3. Ensure that all created ResourceCandidate items are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ResourceCandidate\_GET\_N4 | Search for Resource Candidates with filtering | Pre-requisite: create multiple ResourceCandidate as per TC\_ResourceCandidate\_POST\_N11. Send a GET message to /{apiRoot}/resourceCandidate with filtering of mandatory parametrs as defined in API GET Filtering Operation Conformance like name
2. Verify the response with code 200,
3. Ensure that all created ResourceCandidate items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ResourceCandidate\_GET\_N5 | Search for Resource Candidates with filtering and selected attributes | 1. Pre-requisite: create multiple ResourceCandidate as per TC\_ResourceCandidate\_POST\_N1
2. Send a GET message to /{apiRoot}/resourceCandidate with filtering of mandatory parametrs like name and selection attibutes as defined in API GET Filtering Operation Conformance
3. Verify the response with code 200,
4. Ensure that all created ResourceCandidate items which match the filter criteria are returned in the body of response
5. Ensure that the response message includes only selected parameters
6. Ensure that the body of the response for each resource and selected fields matches the values in the original POST request
 |
| TC\_ResourceCandidate\_DELETE\_N1 | Delete an existing Resource Candidate | Pre-requisite: create ResourceCandidate as per TC\_ResourceCandidate\_POST\_N11. Send a DELETE message to /{apiRoot}/resourceCandidate/{ID} in which ID is identifier of existing ResourceCandidate item
2. Verify the response with code 204,
3. Ensure that the ResourceCandidate item is deleted from the list of available resource candidates
 |
| TC\_ResourceCandidate\_PATCH\_N1 | update an existing Resource Candidate with only patchable parameters | Pre-requisite: create ResourceCandidate as per TC\_ResourceCandidate\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/resourceCandidate/{ID} in which ID is identifier of existing ResourceCandidate item. Include all patchable fields to update the ResourceCandidate
2. Verify the response with code 200,
3. Verify that response body includes all modified fields
 |
| TC\_ResourceCandidate\_PATCH\_N2 | update an existing Resource Candidate with non-patchable parameters | Pre-requisite: create ResourceCandidate as per TC\_ResourceCandidate\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/resourceCandidate/{ID} in which ID is identifier of existing ResourceCandidate item. Include non-patchable fields like lastUpdate, @type, href and id with different value(s) than existing one(s) to update the ResourceCandidate
2. Verify the response with error code 400,
 |

## ResourceSpecification resource TEST CASES

|  |  |  |
| --- | --- | --- |
| **Test Case ID** | **Title** | **Description** |
| TC\_ResourceSpecification\_POST\_N1 | Create Resource Specification  | Pre-requisite: None 1. Send POST message to {apiRoot}/resourceSpecification/with all mandatory parameters and supported optional parameters to create a new ResourceSpecification such as:

|  |
| --- |
| { “name”: “<anytext>”, “description”: “<anytext>”, “lifecycleStatus”: “Active”, "@type": "ResourceFunction", “validFor”: { “startDateTime”: ”<any value with correct datetime format>”, “endDateTime”: “<any value with correct datetime format>” }, "resourceSpecCharacteristic": [ { "name": “<anytext>”,  "description": “<anytext>”,  "valueType": “<anytext>”,  "validFor": { "startDateTime": ”<any value with correct datetime format>”, "endDateTime": ”<any value with correct datetime format>” },  "@type": "ResourceSpecCharacteristic",  "@schemaLocation": ”<any text in url format>”, "minCardinality": 0,  "maxCardinality": 1,  "resourceSpecCharRelationship": [  ],  "resourceSpecCharacteristicValue":[ { "value": “<anytext>”, "validFor": { "startDateTime": ”<any value with correct datetime format>”, "endDateTime": ”<any value with correct datetime format>” } }]} ], "resourceSpecRelationship": [], "feature": []} |

1. Verify the response with code 201, location header set to url of the created resource and body including full representation of the created resource
 |
| TC\_ResourceSpecification\_POST\_N2 | Create Resource Specification with missing mandatory parameter | Pre-requisite: None1. Send POST message to {apiRoot}/resourceSpecification/without mandatory parameters like name and type to create a new ResourceSpecification
2. Verify that server rejects the request by sending the proper response code/error message
 |
| TC\_ResourceSpecification\_GET\_N3 | Search for Resource Specifications- no filtering | Pre-requisite: create multiple ResourceSpecification items as per TC\_ResourceSpecification\_POST\_N11. Send a GET message to /{apiRoot}/resourceSpecification
2. Verify the response with code 200,
3. Ensure that all created ResourceSpecification items are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ResourceSpecification\_GET\_N4 | Search for Resource Specifications with filtering | Pre-requisite: create multiple ResourceSpecification as per TC\_ResourceSpecification\_POST\_N11. Send a GET message to /{apiRoot}/resourceSpecification with filtering of mandatory parametrs as defined in API GET Filtering Operation Conformance like name
2. Verify the response with code 200,
3. Ensure that all created ResourceSpecification items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ResourceSpecification\_GET\_N5 | Search for Resource Specifications with filtering and selected attributes | Pre-requisite: create multiple ResourceSpecification items as per TC\_ResourceSpecification\_POST\_N11. Send a GET message to /{apiRoot}/resourceSpecification with filtering of mandatory parametrs like name and selection attibutes as defined in API GET Filtering Operation Conformance
2. Verify the response with code 200,
3. Ensure that all created ResourceSpecification items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes only selected parameters
5. Ensure that the body of the response for each resource and selected fields matches the values in the original POST request
 |
| TC\_ResourceSpecification\_DELETE\_N1 | Delete an existing Resource Specification | Pre-requisite: create ResourceSpecification as per TC\_ResourceSpecification\_POST\_N11. Send a DELETE message to /{apiRoot}/resourceSpecification/{ID} in which ID is identifier of existing ResourceSpecification item
2. Verify the response with code 204,
3. Ensure that the ResourceSpecification item is deleted from the list of available resource specifications
 |
| TC\_ResourceSpecification\_PATCH\_N1 | update an existing Resource Specification with only patchable parameters | Pre-requisite: create ResourceSpecification as per TC\_ResourceSpecification\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/resourceSpecification/{ID} in which ID is identifier of existing ResourceSpecification item. Include all patchable fields to update the ResourceSpecification
2. Verify the response with code 200,
3. Verify that response body includes all modified fields
 |
| TC\_ResourceSpecification\_PATCH\_N2 | update an existing Resource Specification with non-patchable parameters | Pre-requisite: create ResourceSpecification as per TC\_ResourceSpecification\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/resourceSpecification/{ID} in which ID is identifier of existing ResourceSpecification item. Include non-patchable fields like lastUpdate, @type, href and id with different value(s) than existing one(s) to update the ResourceSpecification.
2. Verify the response with error code 400,
 |

## LogicalResourceSpec resource TEST CASES

|  |  |  |
| --- | --- | --- |
| **Test Case ID** | **Title** | **Description** |
| TC\_LogicalResourceSpec\_POST\_N1 | Create Logical Resource Specification  | Pre-requisite: None 1. Send POST message to {apiRoot}/logicalResourceSpec/with all mandatory parameters and supported optional parameters to create a new LogicalResourceSpec such as:

|  |
| --- |
| { “name”: “<anytext>”, “description”: “<anytext>”, “lifecycleStatus”: “Active”, "@type": "ResourceFunction", “validFor”: { “startDateTime”: ”<any value with correct datetime format>”, “endDateTime”: “<any value with correct datetime format>” }, "resourceSpecCharacteristic": [ { "name": “<anytext>”,  "description": “<anytext>”,  "valueType": “<anytext>”,  "validFor": { "startDateTime": ”<any value with correct datetime format>”, "endDateTime": ”<any value with correct datetime format>” },  "@type": "ResourceSpecCharacteristic",  "@schemaLocation": ”<any text in url format>”, "minCardinality": 0,  "maxCardinality": 1,  "resourceSpecCharRelationship": [  ],  "resourceSpecCharacteristicValue":[ { "value": “<anytext>”, "validFor": { "startDateTime": ”<any value with correct datetime format>”, "endDateTime": ”<any value with correct datetime format>” } }]} ], "resourceSpecRelationship": [], "feature": []} |

1. Verify the response with code 201, location header set to url of the created resource and body including full representation of the created resource
 |
| TC\_LogicalResourceSpec\_POST\_N2 | Create Logical Resource Specification with missing mandatory parameter | Pre-requisite: None1. Send POST message to {apiRoot}/logicalResourceSpec/without mandatory parameters like name to create a new LogicalResourceSpec
2. Verify that server rejects the request by sending the proper response code/error message
 |
| TC\_LogicalResourceSpec\_GET\_N3 | Search for Logical Resource Specifications- no filtering | Pre-requisite: create multiple LogicalResourceSpec items as per TC\_LogicalResourceSpec\_POST\_N11. Send a GET message to /{apiRoot}/logicalResourceSpec
2. Verify the response with code 200,
3. Ensure that all created LogicalResourceSpec items are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_LogicalResourceSpec\_GET\_N4 | Search for Logical Resource Specifications with filtering | Pre-requisite: create multiple LogicalResourceSpec as per TC\_LogicalResourceSpec\_POST\_N11. Send a GET message to /{apiRoot}/logicalResourceSpec with filtering of mandatory parametrs as defined in API GET Filtering Operation Conformance like name and @type
2. Verify the response with code 200,
3. Ensure that all created LogicalResourceSpec items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_LogicalResourceSpec\_GET\_N5 | Search for Logical Resource Specifications with filtering and selected attributes | Pre-requisite: create multiple LogicalResourceSpec items as per TC\_LogicalResourceSpec\_POST\_N11. Send a GET message to /{apiRoot}/logicalResourceSpec with filtering of mandatory parametrs like name and selection attibutes as defined in API GET Filtering Operation Conformance
2. Verify the response with code 200,
3. Ensure that all created LogicalResourceSpec items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes only selected parameters
5. Ensure that the body of the response for each resource and selected fields matches the values in the original POST request
 |
| TC\_LogicalResourceSpec\_DELETE\_N1 | Delete an existing Logical Resource Specification | Pre-requisite: create LogicalResourceSpec as per TC\_LogicalResourceSpec\_POST\_N11. Send a DELETE message to /{apiRoot}/logicalResourceSpec/{ID} in which ID is identifier of existing LogicalResourceSpec item
2. Verify the response with code 204,
3. Ensure that the LogicalResourceSpec item is deleted from the list of available logical resource specifications
 |
| TC\_LogicalResourceSpec\_PATCH\_N1 | update an existing Logical Resource Specification with only patchable parameters | Pre-requisite: create LogicalResourceSpec as per TC\_LogicalResourceSpec\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/logicalResourceSpec/{ID} in which ID is identifier of existing LogicalResourceSpec item. Include all patchable fields to update the LogicalResourceSpec
2. Verify the response with code 200,
3. Verify that response body includes all modified fields
 |
| TC\_LogicalResourceSpec\_PATCH\_N2 | update an existing Logical Resource Specification with non-patchable parameters | Pre-requisite: create LogicalResourceSpec as per TC\_LogicalResourceSpec\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/logicalResourceSpec/{ID} in which ID is identifier of existing LogicalResourceSpec item. Include non-patchable fields like lastUpdate, @type, href and id with different value(s) than existing one(s) to update the LogicalResourceSpec
2. Verify the response with error code 400,
 |

## PhysicalResourceSpec resource TEST CASES

|  |  |  |
| --- | --- | --- |
| **Test Case ID** | **Title** | **Description** |
| TC\_PhysicalResourceSpec\_POST\_N1 | Create Physical Resource Specification  | Pre-requisite: None 1. Send POST message to {apiRoot}/physicalResourceSpec/with all mandatory parameters and supported optional parameters to create a new PhysicalResourceSpec such as:

|  |
| --- |
| { “name”: “<anytext>”, “description”: “<anytext>”, “lifecycleStatus”: “Active”, "@type": "PhysicalDeviceSpec", "@schemaLocation": “<anytext>”, "@baseType":“PhysicalResourceSpec", “validFor”: { “startDateTime”: ”<any value with correct datetime format>”, “endDateTime”: “<any value with correct datetime format>” }, "isBundle": false,  "category": “<anytext>”,  "model": “<anytext>”, "part": “<anytext>”, "sku": "“<anytext>”,  "vendor": “<anytext>”, "resourceSpecCharacteristic": [ { "name": “<anytext>”,  "description": “<anytext>”,  "valueType": “<anytext>”,  "validFor": { "startDateTime": ”<any value with correct datetime format>”, "endDateTime": ”<any value with correct datetime format>” },  "@type": "ResourceSpecCharacteristic",  "@schemaLocation": ”<any text in url format>”, "minCardinality": 0,  "maxCardinality": 1,  "resourceSpecCharRelationship": [  ],  "resourceSpecCharacteristicValue":[ { "value": “<anytext>”, "validFor": { "startDateTime": ”<any value with correct datetime format>”, "endDateTime": ”<any value with correct datetime format>” } }]} ], "resourceSpecRelationship": [], "feature": []} |

1. Verify the response with code 201, location header set to url of the created resource and body including full representation of the created resource
 |
| TC\_PhysicalResourceSpec\_POST\_N2 | Create Physical Resource Specification with missing mandatory parameter | Pre-requisite: None1. Send POST message to {apiRoot}/physicalResourceSpec/without mandatory parameters like name to create a new PhysicalResourceSpec
2. Verify that server rejects the request by sending the proper response code/error message
 |
| TC\_PhysicalResourceSpec\_GET\_N3 | Search for Physical Resource Specifications- no filtering | Pre-requisite: create multiple PhysicalResourceSpec items as per TC\_PhysicalResourceSpec\_POST\_N11. Send a GET message to /{apiRoot}/physicalResourceSpec
2. Verify the response with code 200,
3. Ensure that all created PhysicalResourceSpec items are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_PhysicalResourceSpec\_GET\_N4 | Search for Physical Resource Specifications with filtering | Pre-requisite: create multiple PhysicalResourceSpec as per TC\_PhysicalResourceSpec\_POST\_N11. Send a GET message to /{apiRoot}/physicalResourceSpec with filtering of mandatory parametrs as defined in API GET Filtering Operation Conformance like name
2. Verify the response with code 200,
3. Ensure that all created PhysicalResourceSpec items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_PhysicalResourceSpec\_GET\_N5 | Search for Physical Resource Specifications with filtering and selected attributes | Pre-requisite: create multiple PhysicalResourceSpec as per TC\_PhysicalResourceSpec\_POST\_N11. Send a GET message to /{apiRoot}/physicalResourceSpec with filtering of mandatory parametrs like name and selection attibutes as defined in API GET Filtering Operation Conformance
2. Verify the response with code 200,
3. Ensure that all created PhysicalResourceSpec items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes only selected parameters
5. Ensure that the body of the response for each resource and selected fields matches the values in the original POST request
 |
| TC\_PhysicalResourceSpec\_DELETE\_N1 | Delete an existing Physical Resource Specification | Pre-requisite: create PhysicalResourceSpec as per TC\_PhysicalResourceSpec\_POST\_N11. Send a DELETE message to /{apiRoot}/physicalResourceSpec/{ID} in which ID is identifier of existing PhysicalResourceSpec item
2. Verify the response with code 204,
3. Ensure that the PhysicalResourceSpec item is deleted from the list of available physical resource specifications
 |
| TC\_PhysicalResourceSpec\_PATCH\_N1 | update an existing Physical Resource Specification with only patchable parameters | Pre-requisite: create PhysicalResourceSpec as per TC\_PhysicalResourceSpec\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/physicalResourceSpec/{ID} in which ID is identifier of existing PhysicalResourceSpec item. Include all patchable fields to update the PhysicalResourceSpec
2. Verify the response with code 200,
3. Verify that response body includes all modified fields
 |
| TC\_PhysicalResourceSpec\_PATCH\_N2 | update an existing Physical Resource Specification with non-patchable parameters | Pre-requisite: create PhysicalResourceSpec as per TC\_PhysicalResourceSpec\_POST\_N11. Send a PATCH message using json/merge to /{apiRoot}/physicalResourceSpec/{ID} in which ID is identifier of existing PhysicalResourceSpec item. Include non-patchable fields like lastUpdate, @type, href and id with different value(s) than existing one(s) to update the PhysicalResourceSpec
2. Verify the response with error code 400,
 |

## ImportJob resource TEST CASES

|  |  |  |
| --- | --- | --- |
| **Test Case ID** | **Title** | **Description** |
| TC\_ImportJob\_POST\_N1 | Create an Import Job  | Pre-requisite: None 1. Send POST message to {apiRoot}/importJob/with all mandatory parameters and supported optional parameters to create a new ImportJob such as:

|  |
| --- |
| { "url": “<http://catalogMangement/projects/myProject/>”,“path”: “c:/resourceCatalog/projects/myProject.zip”} |

1. Verify the response with code 201, location header set to url of the created resource and body including full representation of the created resource
 |
| TC\_ImportJob\_POST\_N2 | Create Import Job with missing mandatory parameter | Pre-requisite: None1. Send POST message to {apiRoot}/importJob/without mandatory parameters like url to create a new ImportJob
2. Verify that server rejects the request by sending the proper response code/error message
 |
| TC\_ImportJob\_GET\_N3 | Search for Import Jobs- no filtering | Pre-requisite: create multiple ImportJob items as per TC\_ImportJob\_POST\_N11. Send a GET message to /{apiRoot}/importJob
2. Verify the response with code 200,
3. Ensure that all created ImportJob items are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ImportJob\_GET\_N4 | Search for Import Jobs with filtering | Pre-requisite: create multiple ImportJob as per TC\_ImportJob\_POST\_N11. Send a GET message to /{apiRoot}/importJob with filtering of mandatory parametrs as defined in API GET Filtering Operation Conformance like name
2. Verify the response with code 200,
3. Ensure that all created ImportJob items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ImportJob\_GET\_N5 | Search for Import Jobs with filtering and selected attributes | Pre-requisite: create multiple ImportJob as per TC\_ImportJob\_POST\_N11. Send a GET message to /{apiRoot}/importJob with filtering of mandatory parametrs like name and selection attibutes as defined in API GET Filtering Operation Conformance
2. Verify the response with code 200,
3. Ensure that all created ImportJob items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes only selected parameters
5. Ensure that the body of the response for each resource and selected fields matches the values in the original POST request
 |
| TC\_ImportJob\_DELETE\_N1 | Delete an existing Import Job | Pre-requisite: create ImportJob as per TC\_ImportJob\_POST\_N11. Send a DELETE message to /{apiRoot}/importJob/{ID} in which ID is identifier of existing ImportJob item
2. Verify the response with code 204,
3. Ensure that the ImportJob item is deleted from the list of available import jobs
 |

## ExportJob resource TEST CASES

|  |  |  |
| --- | --- | --- |
| **Test Case ID** | **Title** | **Description** |
| TC\_ExportJob\_POST\_N1 | Create an Export Job  | Pre-requisite: (Resource) Catalog should be populated with (resource) catalog entities like ResourceSpecification, ResourceCandidate, and so on.1. Send POST message to {apiRoot}/exportJob/with all mandatory parameters and supported optional parameters to create a new ExportJob such as:

|  |
| --- |
| { "url": “<http://hostname:port/catalogMangement/resourceCandidate/>”,“path”: “c:/resourceCatalog/projects/RC/”} |

1. Verify the response with code 201, location header set to url of the created resource and body including full representation of the created resource
 |
| TC\_ExportJob\_POST\_N2 | Create Export Job with missing mandatory parameter | Pre-requisite: Pre-requisite: (Resource) Catalog should be populated with (resource) catalog entities like ResourceSpecification, ResourceCandidate, and so on.1. Send POST message to {apiRoot}/exportJob/without mandatory parameters like url to create a new ExportJob
2. Verify that server rejects the request by sending the proper response code/error message
 |
| TC\_ExportJob\_GET\_N3 | Search for Export Jobs- no filtering | Pre-requisite: create multiple ExportJob items as per TC\_ExportJob\_POST\_N11. Send a GET message to /{apiRoot}/exportJob
2. Verify the response with code 200,
3. Ensure that all created ExportJob items are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ExportJob\_GET\_N4 | Search for Export Jobs with filtering | Pre-requisite: create multiple ExportJob as per TC\_ExportJob\_POST\_N11. Send a GET message to /{apiRoot}/exportJob with filtering of mandatory parametrs as defined in API GET Filtering Operation Conformance like name
2. Verify the response with code 200,
3. Ensure that all created ExportJob items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes all mandatory parameters and specified optional parameters
5. Ensure that the body of the response for each resource matches the values in the original POST request
 |
| TC\_ExportJob\_GET\_N5 | Search for Export Jobs with filtering and selected attributes | Pre-requisite: create multiple ExportJob as per TC\_ExportJob\_POST\_N11. Send a GET message to /{apiRoot}/exportJob with filtering of mandatory parametrs like name and selection attibutes as defined in API GET Filtering Operation Conformance
2. Verify the response with code 200,
3. Ensure that all created ExportJob items which match the filter criteria are returned in the body of response
4. Ensure that the response message includes only selected parameters
5. Ensure that the body of the response for each resource and selected fields matches the values in the original POST request
 |
| TC\_ExportJob\_DELETE\_N1 | Delete an existing Export Job | Pre-requisite: create ExportJob as per TC\_ExportJob\_POST\_N11. Send a DELETE message to /{apiRoot}/exportJob/{ID} in which ID is identifier of existing ExportJob item
2. Verify the response with code 204,
3. Ensure that the ExportJob item is deleted from the list of available export jobs
 |

## Release History

|  |  |  |  |
| --- | --- | --- | --- |
| **Release Number** | **Date**  | **Release led by:** | **Description**  |
| Release 1.0 | 8/15/2017 | Kamal MaghsoudlouEricssonkamal.maghsoudlou@ericsson.com | First Release of Draft Version of the Document. |

## Contributors to Document

|  |  |
| --- | --- |
| * Kamal Maghsoudlou
 | Ericsson |
| * Mariano Belaunde
 | Orange |
| * Pierre Gauthier
 | TM Forum |