TM Forum Specification

Resource Function Activation and Configuration API REST Specification

TMF664

Release 17.5.0

January 2018

Release: TM Forum Release 17.5.0	**Status: Member Evaluation**
Version: 0.3.2	**IPR Mode: RAND**

NOTICE

Direct inquiries to the TM Forum office:

4 Century Drive, Suite 100

Parsippany, NJ 07054 USA

Tel No. +1 973 944 5100

Fax No. +1 973 944 5110

TM Forum Web Page: www.tmforum.org

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

The following document is intended to provide details of the REST API for Resource Function provisioning and lifecycle management of Resource Functions (Network Service, VNF and PNF in ETSI NFV terminology) composed from Physical and Virtual Resource Functions.

It is based on the requirements specified in TR255 & IG1147.

SAMPLE USE CASES

Please refer to TR255 & IG1147 for detailed background and use cases. Sample payloads for a few complex use cases have been included in this section to assist the user in consuming this API. It is suggested that the reader reads the subsequent sections to familiarize herself with the API spec before coming back to this section.

USE CASE 1: Request creation of a composite resource function including creation of some of the component resource functions.

The resource function API allows users to request creation of a composite resource function from a number of other atomic and composite resource functions and connected in a topology. There will be situations where the components need to be created as part of the same request. In this case, the ids of the components are not known and hence not available to include in the connectivity section of the API.

See the simple example below where the requirement is to create a new vCDN resource function. In this example, the Virtual Router Z along with the connection points I, H and K are available. However, the components with a red border i.e. the virtual content server W along with its connection points J & L as well as the link A need to be created as part of the request. There is then a requirement to connect these newly created components together in a certain topology.

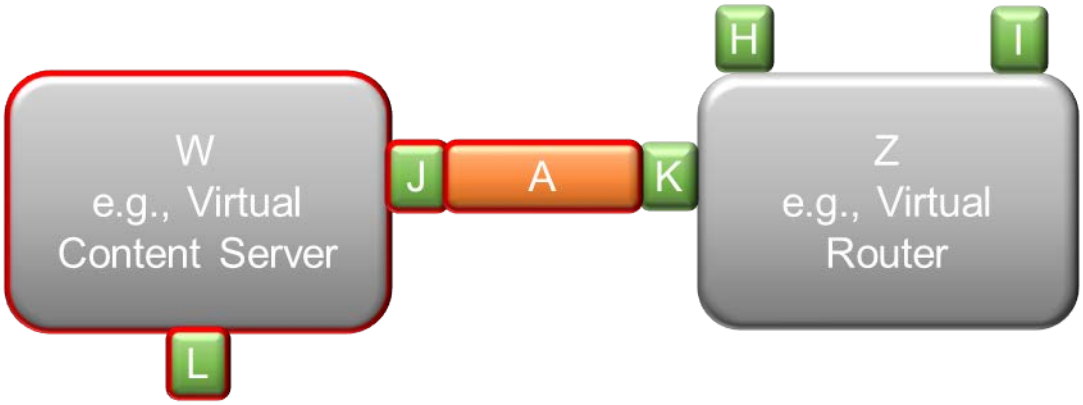


Figure 1: Creation request for a composite Resource Function along with its atomic components

The approach will be for the consumer of the API to provide its own ID based on some algorithm provided by the provider. This will ensure that the ID is unique in the providers system and will not interfere with IDs of existing resource functions. An example could be an ID based on a certain prefix like 'CID' followed by a number generated from date and time combination.

The request will look like this. (only the relevant parts of the payload are provided).

```
"resourceRelationship": [
      (Array of supporting resource functions that need to be created or reference
s to resource functions that have already been created)
       {
        "type":"contains"
        "id":"CID-2017102689098", ( Generated by consumer based on algorithm provi
ded)
        …… ( Other parameters for Resource function A in the diagram)
```



```
        },
       {
        “type”:”contains”
        “id”:”CID-2017102689100”, ( Generated by consumer based on algorithm provi
ded)
        …… ( Other parameters for Resource function J in the diagram)
        },
       {
        “type”:”contains”
        “id”:”CID-2017102689102”, ( Generated by consumer based on algorithm provi
ded)
        …… ( Other parameters for Resource function W in the diagram)
        },
        {
        “type”:”contains”
        “id”:”CID-2017102689104”, ( Generated by consumer based on algorithm provi
ded)
        …… ( Other parameters for Resource function W in the diagram)
        },
        {
        “type”:”contains”
        “id”:”RF-1980987”, ( Id of existing resource function K)
        },
        {
        “type”:”contains”
        “id”:”RF-2980487”, ( Id of existing resource function Z)
        },
        ……. ( Ids of other existing resource functions
  ],
    "connectivity": [
      {
        (Connect A to J in diagram. The URL to use is given by provider)
        "source": "http://serverlocation:port/resourceFunction/CID-2017102689098",
        "target": "http://serverlocation:port/resourceFunction/ CID-2017102689100"
,
        "relationship": "AdjacentTo"
      }
      {
        (Connect J to W in diagram. The URL to use is given by provider)
```

```
        "source": "http://serverlocation:port/resourceFunction/ CID-2017102689102"
,
        "target": "http://serverlocation:port/resourceFunction/ CID-2017102689100"
,
        "relationship": "AdjacentTo"
      }
      ……. ( Other connections)
    ],
```

The provider may want to persist the Ids supplied by the consumer as the ID for the resource function or it may replace them with ids of its own. This is an implementation decision.

USE CASE 2: Request creation of a flow on an existing topology.

The diagram below has been taken from TR255A where the requirement is to create a new flow (red line) on the existing topology.

Figure 2: Flow creation on existing topology

The Flow is just another type of resource function. The payload for request to create this flow will look like this.

```
[
  {
    "name": "CDN Cluster flow 1",
    "description": "Consumer provided description of the flow",

    "resourceSpecification": {
      "id": "RS-6789",
      "href": "http://serverlocation:port/resourceSpecification/Flowspec-6789"
    },
    "characteristic": [
     (The route is partially specified i.e. L,X,Z. These characteristics are speci
fic to the resource function of type Flow and based on requirements in TR255A)
 # The route characteristic needs an array of ResourceFunction Ids and Function i.
e forward as input.
      {
        "name": "route",
        "valueType": "array",
        "value": "[[ID-L,forward],[ID-X,forward],[ID-Z,forward]]"
```

```
      }
      {
        "name": "flowContext",
        "valueType": "string",
        "value": "Content Upload"
      }
      {
        "name": "flowType",
        "valueType": "string",
        "value": "Point-To-Point"
      }
    ],
```

```
    "feature": [
```

The Traffic Characteristics feature is specific to the resource function of type Flow and based on requirements in TR255A)

```
      {
        "name": "Traffic Characteristics",
        "listOfCharacteristics":
```

```
         "characteristic": [
```

```
      {
        "name": "capacity",
        "valueType": "string",
        "value": "1000"
      }
      {
        "name": "bandwidthProfileType",
        "valueType": "string",
        "value": "MEF10.1"
      }
      {
        "name": "committedBurstRate",
        "valueType": "string",
        "value": "100"
      }
```

```
Please refer to TR255A for other characteristics that may need to be provided. The
se will be included in the resource specification for the Flow.


      }
    ],
    "resourceRelationship": [
      (Array of supporting resource functions that need to be created or reference
s to resource functions that have already been created)
       {
             “id”:”RF-1980987”,
              "source": http://serverlocation:port/resourceFunction/ RF-1980987 (Th
is is the id of the composite resource function represented in the diagram above)
        },
  ],
```

MIGRATION FROM ENTITY PROVISIONING API

This API was originally named EntityProvisioning API. This note provides guidance on migration from the EntityProvisioning API to this Resource Function Activation and Configuration API.

Resource Model Comparison

The Entity Provisioning API had 3 main resources.

1. Network Service
2. VNF – Virtual Network Function
3. PNF – Physical Network Function

It lacked the connectivity resources in R16.5.

A network service is composed from a number of VNFs and PNFs connected in a particular topology. It was felt that all these 3 objects could be modelled using the Resource Function entity.

This API specification introduces Resource Function which is used to represent a Network Service as well as a Network Function. The Network Service and Network Function class definitions and associations in TR244 (which, in turn, builds on concepts from the SID addenda on Logical Resource and Service) are utilized to define the Resource Function.

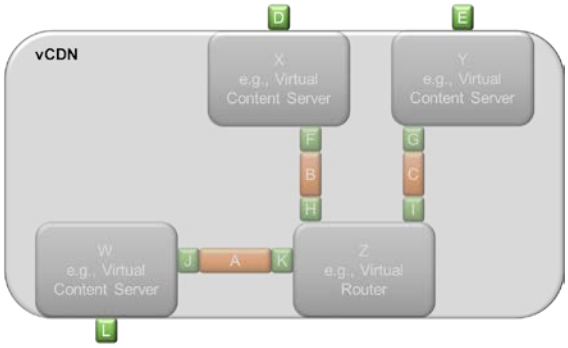
After further consideration, the distinction between NS and NF was deemed to be artificial, and it was decided that the same functionality could be handled by one or the other. Further, there was an issue with the word "Network" as this limits the scope. Resource Function is more general as "Resource" can refer to "Network" but also other types of resources such as storage and compute.

Hence, it was decided that RF composite would be used to replace NS (both atomic and composite) and composite NFs, and RF atomic would be used to replace atomic NFs. It was decided that having atomic and composite NS and NF is redundant. This applies to both virtual and physical NFs.

The concept of Resource Function Specification is defined in TM Forum TR264 (this will eventually be moved to the SID model proper). From TR264 (with some minor editing):

> A **ResourceFunction** specifies a function as a behavior to transform inputs of any nature into outputs of any nature independently from the way it is provided. It is typically created by a function designer who may not have specific knowledge on realization architecture (for example using English text explanations with diagrams, as in RFC standards, or preferably a machine interpretable language). NetworkFunction, OfficeFunction as well as GameFunction are examples of specialization (of ResourceFunction).
>
> A **ResourceFunctionSpec** is associated to a ResourceFunction in order to indicate the expected mandatory and optional characteristics.

Let us use the example of the virtual CDN network service to illustrate the mapping between the Entity Provisioning API to the Resource Function Activation and Configuration API.



Resource Model

NetworkService

…Attributes with direct mapping to resource function

……

-resourceFunctionSpecification

-supportingNetworkFunction{

VNFX

VNFY

VNFZ

VNFW

}

Resource Model

ResourceFunction

…Attributes with direct mapping to network service

……

-resourceFunctionSpecification

-resourceRelationship{

ResourceFunctionX

ResourceFunctionY

ResourceFunctionZ

ResourceFunctionW

}

RESOURCE MODEL

Managed Entity and Task Resource Models

Resource Function

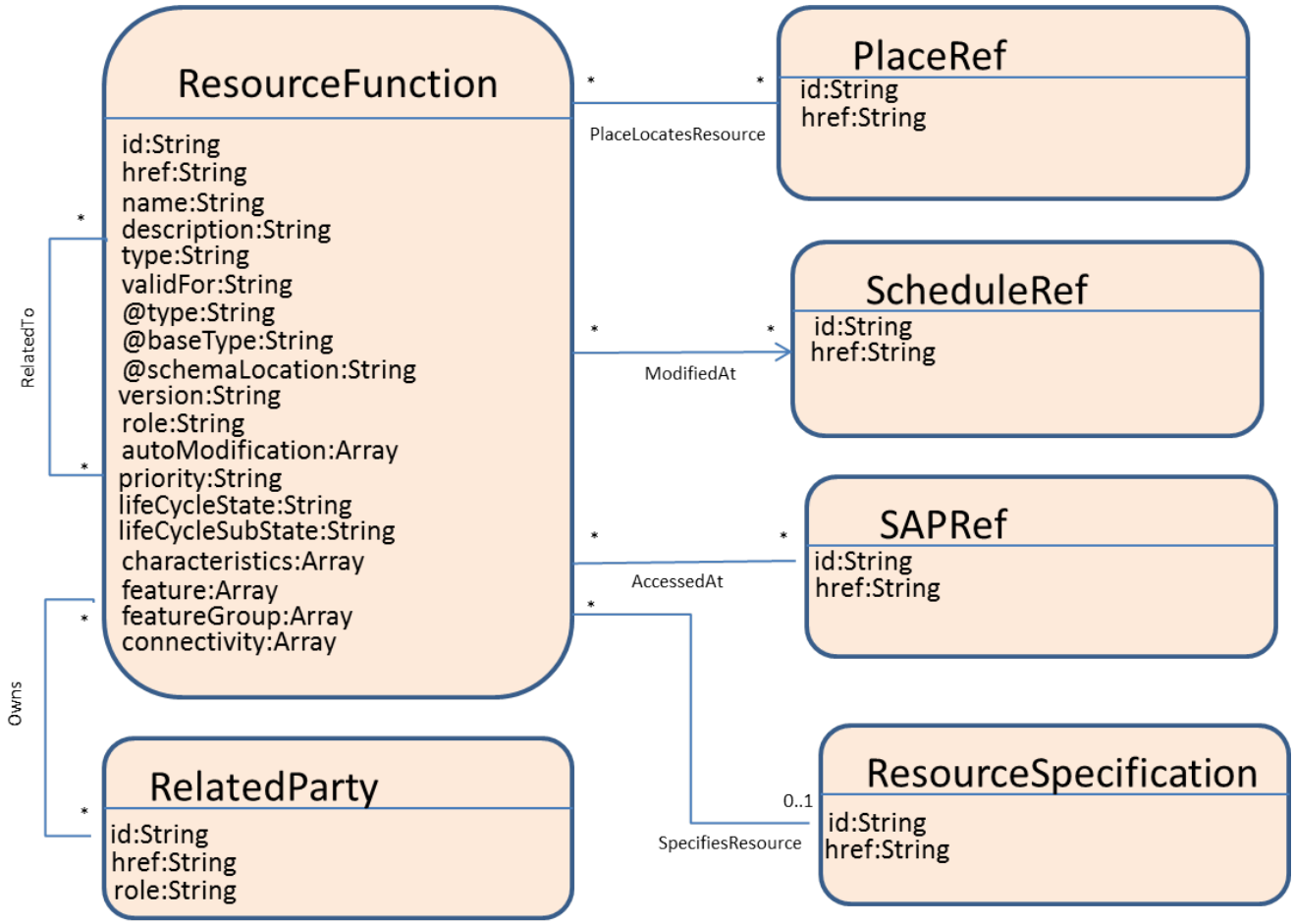


Figure 3: Resource Function Managed Entity Resource Model

A Resource Function can be atomic or composite i.e. composed from one or many Resource Functions. A Resource Function is equivalent to a Network Service, VNF or PNF as defined in ETSI NFV.

```
[
 {
   "id": "17898",
```

```
"href": "http://serverlocation:port/resourceFunction/17898",
"name": "CDN Cluster",
"description": "CDN capability spread across multiple geographies",
"type": "Content Delivery",
"validFor": "string",
"@type": "string",
"@baseType": "string",
"@schemaLocation": "string",
"version": "1.2",
"role": "Backup Media Store",
"place": {
  "href": "http://serverlocation:port/location/4980",
  "id": "4980"
},
"autoModification": {
  "name": "scaleStorage",
  "value": "scaleIn"
},
"priority": "2",
"lifeCycleState": "planning",
"lifeCycleSubState": "string",
"schedule": [
  {
    "id": "SCH-78906",
    "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
  }
],
"sap": [
  {
    "id": "SAP-49876",
```



```
      "href": "http://serverlocation:port/resourceFunction/SAP-49876"
    }
  ],
  "resourceSpecification": {
    "id": "RS-6789",
    "href": "http://serverlocation:port/resourceSpecification/RS-6789"
  },
  "characteristic": [
    {
      "name": "bandwidth",
      "value": "100MB"
    }
  ],
  "feature": [
    {
      "name": "Tarpit"
    }
  ],
  "featureGroup": [
    {
      "name": "Filtering"
    }
  ],
  "resourceRelationship": [
    {
      "type": "contains",
      "resource": {
        "id": "1234",
        "href": "http://serverlocation:port/resourceFunction/1234"
      }
```



```
    }
  ],
  "connectivity": [
    {
      "source": "http://serverlocation:port/resourceFunction/6789",
      "target": "http://serverlocation:port/resourceFunction/1234",
      "relationship": "ConnectsTo"
    }
  ],
  "relatedParty": [
    {
      "id": "1234",
      "href": "http://serverlocation:port/partyManagement/partyRole/1234",
      "role": "Admin"
    }
  ]
 }
]
```

Field	Description
id	Identifier of the Resource Function instance. Required to be unique. Used in URIs as the identifier of the service (for modify or delete use cases).
href	Reference to the service.
name	A user friendly moniker for the Resource Function.
description	A description of the Resource Function (What does it provide).
type	Type of Resource Function as specified by the provider of the API.
validFor	Period for which the resource function is valid
@type	Class type of the Resource Function
@baseType	Immediate base (class) of the Resource Function i.e. LogicalResource
@schemaLocation	A link to the schema describing this Resource Function
version	Version of the Resource Function as specified by the provider of the API.
role	Role of Resource Function. Used when Resource Function is a component of a composite Resource Function and the exact role of the service within the composite is not clear from descriptor/location.
place	Location at which Resource Function is required.
autoModification	List of the kinds of auto-modifications that are applied to a given Resource Function e.g what can be scaled.
priority	Priority of the Resource Function. Decides what happens in a contention scenario.
lifeCycleState	Provides the stage at which Resource Function has been instantiated. This is a compound state defined in TR255.
lifeCycleSubState	Substate applicable Resource Function. List of values possible depends on the lifeCycleState. Please see section on Resource Function state model.
schedule	This is a reference to a schedule. Allows consumers to schedule modifications to the service at certain times"
sap	The service access points available on the resource function.
resourceSpecification	Pointer to the resource specification that will be used to create this resource function
characteristic	List of resource characteristics. This is based on the resource specification
feature	List of features. Features are a collection of related Characteristics
featureGroup	List of feature groups. Feature group could contain a list of Characteristics or Features or both.
resourceRelationship	This is a list of composite and atomic resource functions from which this resource function is composed. These are all the vertices of the graph. Here the type is set to ‘contains’.
connectivity	These are the edges of the graph and provide details of how the resource functions listed in resourceRelationship are connected
relatedParty	List of related parties including their roles.

Resource Function State Model

This material has been reproduced from TR255.

The following state machine applies to both atomic and composite RFs:

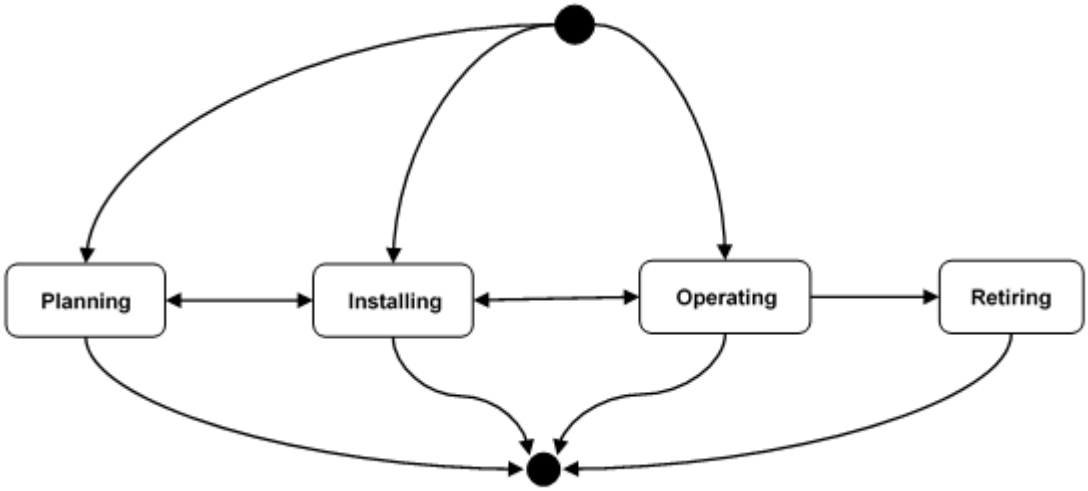


Figure 4: Resource Function state model

The states are defined as below.

Note: "entity" is used to refer to both atomic and composite RFs.

Planning phase

During the Planning phase the entity is scheduled for deployment in accordance with a specific plan.

During this phase, the entity is assigned the composite state value of “Planning.”

Installing phase

During this phase, the entity undergoes a full commissioning process until it is finally ready for work

During this phase, the entity is assigned the composite Inventory state value of “Installing.”

Resources may also be installed in the network regardless of any specific plan.

Operating phase

The entity is fully provisioned and ready to support consumers (assuming the entity is administratively activated and operationally working).

Retiring phase

During this phase, the entity undergoes all necessary procedures for its decommissioning and phasing out.

During this phase, the entity is assigned the composite state value “Retiring.”

The states (or “phases,” to be more precise) can be decomposed into sub-states. Currently, only the Planning and Operating phases are divided into sub-states. In the future releases of TR255, additional phases may be further decomposed.

The sub-states for the Planning phase are as follows:

- **Proposed** - a requirement for the entity has been identified, and the entity has been proposed to address the requirement, but entity characteristics or deployment details have not been agreed.

Feasibility Checked – a check has been done to see if the proposed entity can be instantiated as requested. At this point, the design of the entity is not complete and nothing has been ordered with regard to the given entity.

Designed – the characteristics of the entity and its deployment have been completely identified but nothing exists in the network at this point in support of the resource. Firm agreement has been reached to satisfy a requirement using the resource.

Ordered – an order for delivery of an entity type or an instance of an existing entity type has been agreed.

The sub-states for the Operating phase are as follows:

Administrative sub-states - the states listed below are typically set by a management / control system or by some policy (e.g., automatically put the entity in Deactivated sub-state once it reaches Installing / Accepted).

Activated- the entity is working, has been configured and activated and can be used by client(s). The entity is fully operation in the sense that it can meet all requirements for which it was designed.

Deactivated - the entity is working, but cannot be used by a client.

Operational sub-states - the Operational sub-state gives an indication of how well (or how full) the entity is functioning.

Working – the entity is completely working as intended.

Meeting All SLAs – the entity is presently meeting all the SLAs promised to each of its clients, but is not completely functional.

An example would be an entity (say a VNF) that when fully operational can support 5 clients but presently can only support 3 clients. However, the VNF currently only has 3 three clients and the VNF is providing the promised SLA to those 3 clients.

Meeting Some SLAs – the entity is meeting only some of the SLAs promised to its clients

Meeting No SLAs – the entity is meeting none of the SLAs promised to its clients

Not Working – the entity is completely non-functionally.

The above definition of operational sub-states is a departure from earlier work based on ITU-T X.731 and similar work in the IETF where there are only two values for the operational (i.e., enabled and disabled). In general, but in particular for virtualized networks, it is recognized that a consideration of the SLA with respect to the resource is needed (and thus the expanded set of values in the operational sub-state in this document).

ResourceFunction/Heal – Task Resource

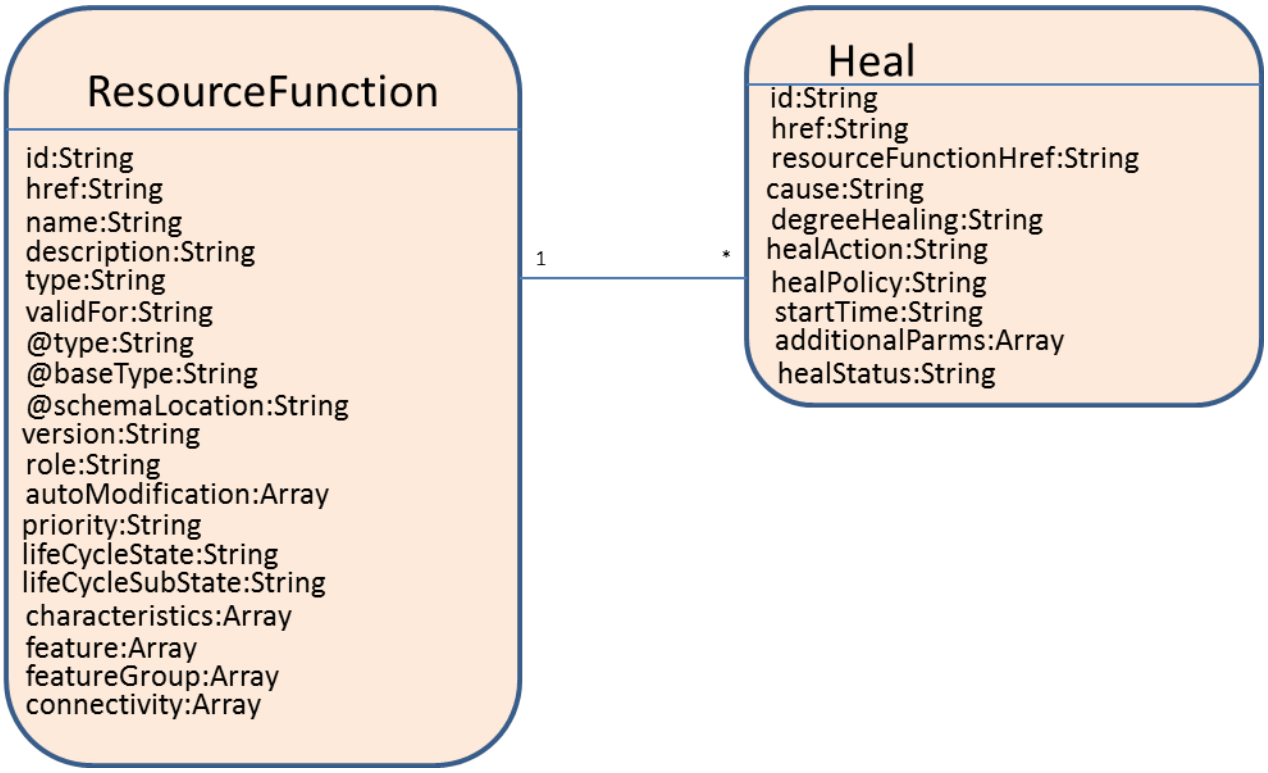


Figure 5: Heal task resource model

Task resource used to request heal of the Resource Function.

```
{
  "id": "490987",
  "href": "http://serverlocation:port/resourceFunction/scale/490987",
  "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "cause": "SLA violation",
  "degreeOfHealing": "Complete - Restore to state before failure",
  "healAction": "/conf/lab/healfast.sh",
  "healPolicy": {
    "id": "Pol-3490",
    "href": "http://serverlocation:port/policy/Pol-3490"
  },
  "startTime": "00:00:00",
```

```
  "additionalParms": [
    {
      "name": "string",
      "value": "string"
    }
  ],
  "healStatus": "In Progress"
}
```


Field	Description
Id	Identifier of the Heal task resource. Required to be unique. Used in URIs as the identifier of the Heal task resource
href	Reference to the heal task resource
resourceFunctionHref	Reference to the Resource Function that needs to be healed
cause	Reason why the heal is being requested
degreeHealing	Indicates the degree of healing required
healAction	Exact action to be taken as part of the heal process or a pointer to a script to be run
healPolicy	Reference to the policy to be applied
startTime	The time when the heal action needs to commence. This allows a delay to be added.
additionalParms	Additional parameters to be sent to the heal action as name value pairs.
healStatus	Status of the heal process.

ResourceFunction/Scale – Task Resource

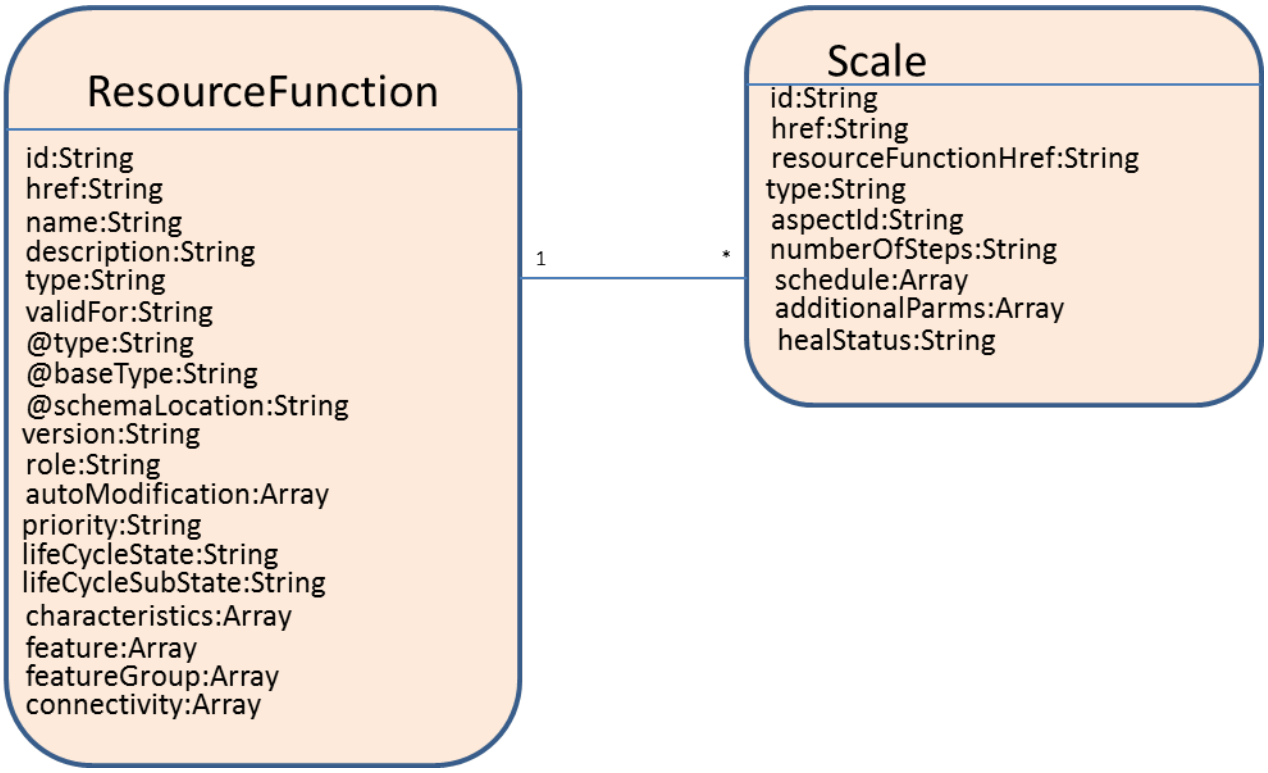


Figure 6: Scale task resource model

Task resource used to request scale of the Resource Function.

```
{
  "id": "180987",
  "href": "http://serverlocation:port/resourceFunction/scale/180987",
  "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "type": "Scale Out",
  "aspectId": "Quick Access Memory",
  "numberOfSteps": "2",
  "schedule": [
    {
      "id": "SCH-78906",
      "href": "http://serverlocation:port/resourceSpecification/SCH-
78906"
```



```
    }
  ],
"additionalParms": [
    {
      "name": "string",
      "value": "string"
    }
  ],
  "scaleStatus": "In Progress"
}
```

Field	Description
Id	Identifier of the Scale task resource. Required to be unique. Used in URIs as the identifier of the Scale task resource
href	Reference to the scale task resource
resourceFunctionHref	Reference to the Resource Function that needs to be scaled
type	Type of scale requested i.e. Scale out or Scale up etc.
aspectId	Scaling aspect is the dimension along which the Resource Function needs to be scaled. The id of the aspect is provided here.
numberOfSteps	Number of scaling steps in the direction indicated by type of scale
schedule	Schedule for the scale. If not provided then needs to be actioned immediately
additionalParms	Various parameters needed as input to the scale request.
scaleStatus	Status of the scale operation.

ResourceFunction/Migrate – Task Resource

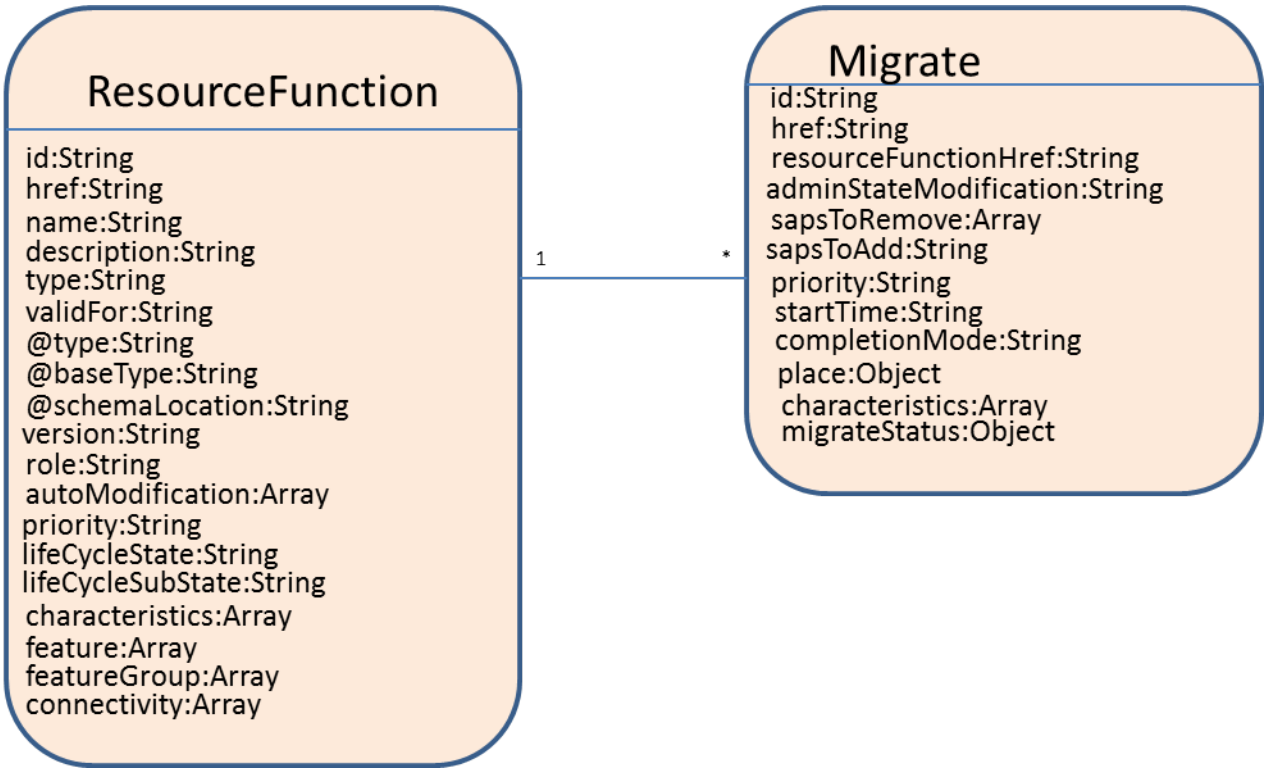


Figure 7: Migrate task resource model

Task resource used to request migration of the Resource Function.

```
{
  "id": "80987",
  "href": "http://serverlocation:port/resourceFunction/migrate/80987",
  "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "adminStateModification": "locked",
  "sapsToRemove": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "sapsToAdd": [
```

```
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "priority": "1",
  "startTime": "2017:11:7:15:53:10Z",
  "completionMode": "bestEffort",
  "place": {
    "href": "http://serverlocation:port/place/4980",
    "id": "4980"
  },
  "characteristics": [
    {
      "name": "string",
      "valueType": "string",
      "value": "string"
    }
  ],
  "migrateStatus": "In Progress"
              }
```


Field	Description
Id	Identifier of the Migrate task resource. Required to be unique. Used in URIs as the identifier of the Migrate task resource
href	Reference to the migrate resource
resourceFunctionHref	Reference to the Resource Function that needs to be migrated
adminStateModification	Sub-state required before migrating is carried out
sapsToRemove	Service Access Points that need to be removed when service is migrated
sapsToAdd	Service Access Points that need to be added when service is migrated
priority	Priority of the migrate operation.

startTime	Start time for the migrate process. Starts immediately if not populated.
completionMode	In what mode is the migrate operation to be performed.
place	Target Location at which Resource Function is to be migrated
characteristics	Additional attributes to pass to the migrate operation
migrateStatus	Status of the migrate operation

Monitor

The Monitor resource is used to monitor the execution of async requests on specific resource.

```
{
  "id": "12",
  "state": "MonitorState",
  "type": "monitor",
  "request": {
    "method": "",
    "to": "",
    "body": "",
    "header": [
       {
      "name": "string",
      "value": "string"
      }
    ]
  },
  "response": {
    "statusCode": "",
    "body": "",
    "header": [
       {
```

```
      "name": "string",
      "value": "string"
      }
    ]
 },

 "href": " http://serverlocation:port/place/4980",
 "sourceHref": " http://serverlocation:port/place/80"
 }
```


Field	Description
id	Identifier of an instance of the monitor. Required to be unique within the resource type. Used in URIs as the identifier for specific instances of a type
type	Name of resource type i.e. monitor
request	Represents the request
response	Represent the response
state	The monitor state of the resource. InProgress, InError, Completed
href	The reference to this monitor
sourceHref	The monitored resource href

Notification Resource Models

RESOURCEFUNCTIONCREATIONNOTIFICATION

Used to notify that a resource function has just been newly created.

```
{
  "event": {
        "resourceFunction":
        {
           "id": "4564",
          //Following a whole representation of the Resource Function
with all its attributes. See ResourceFunction resource.
        }
  },
  "eventType": "resourceFunctionCreationNotification",
}
```

RESOURCEFUNCTIONMODIFICATIONNOTIFICATION

Used to notify that a resource function has just been modified.

```
{
  "event": {
        "resourceFunction":
        {
           "id": "4564",
          //Following a whole representation of the modified Resource
Function with all its attributes. See ResourceFunction resource.
        }
```

```
  },
  "eventType": "resourceFunctionModificationNotification",
}
```

RESOURCEFUNCTIONDELETIONNOTIFICATION

Used to notify that a resource function has just been deleted.

```
{
  "event": {
        "resourceFunction":
        {
           "id": "4564",
          //Following a whole representation of the deleted Resource Function with all its attributes. See ResourceFunction resource.
        }
  },
  "eventType": "resourceFunctionDeletionNotification",
}
```

API OPERATION TEMPLATES

For every single of operation on the entities use the following templates and provide sample REST requests and responses.

Remember that the following Uniform Contract rules must be used:

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource.
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute Task Resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods.

Filtering and attribute selection rules are described in the TMF REST Design Guidelines.

Notifications are also described in a subsequent section.

GET /API/RESOURCEFUNCTION/{ID}

This Uniform Contract operation is used to retrieve the representation a Resource Function.

Note that collections can be retrieved via GET /API/ RESOURCEFUNCTION with no {ID}

Description:

- This operation is used to retrieve the service information including the ID
- Attribute selection is enabled.

Behavior:

- Status code 200 – if the request was successful
- Status code 404 Not found – supplied ID does not match a known service

REQUEST

```
GET /api/resourceFunction/17898?fields=relatedParty,field2 ,field3&name=value
Accept: application/json
```

RESPONSE

```
200
Content-Type: application/json
```


```
{
  "id": "17898",
  "href": "http://serverlocation:port/resourceFunction/17898",
  "name": "CDN Cluster",
  "description": "CDN capability spread across multiple geographies",
  "type": "Content Delivery",
  "version": "1.2",
  "role": "Backup Media Store",
  "place": {
    "href": "http://serverlocation:port/place/4980",
```



```
      "id": "4980"
    },
    "autoModification": "scaleStorage",
    "priority": "2",
    "lifeCycleState": "planning",
    "lifeCycleSubState": "unknown",
    "schedule": [
      {
        "id": "SCH-78906",
        "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
      }
    ],
    "sap": [
      {
        "id": "SAP-49876",
        "href": "http://serverlocation:port/resourceSpecification/SAP-49876"
      }
    ],
    "resourceSpecification": {
      "id": "RS-6789",
      "href": "http://serverlocation:port/resourceSpecification/RS-6789"
    },
    "characteristic": [
      {
        "name": "bandwidth",
        "value": "100MB"
      }
    ],
    "resourceRelationship": [
      (Details of supporting resource functions or references to resource functi
ons depending of depth of information required)
       "oneOf":[{
             "$ref": "#/definitions/ResourceFunction”},
              {
               "$ref": "#/definitions/ResourceFunctionRef”
              }]

    ],
```

```
    "connectivity": [
      {
        "source": "http://serverlocation:port/resourceFunction/6789",
        "target": "http://serverlocation:port/resourceFunction/1234",
        "relationship": "AdjacentTo"
      }
    ],
    "relatedParty": [
      {
        "id": "1234",
        "href": "http://serverlocation:port/partyManagement/partyRole/1234",
        "role": "Admin"
      }
    ]
  }
```


POST API/RESOURCEFUNCTION

This Uniform Contract operation is used to create a Resource Function. Resource Function is a managed entity.

This operation can create the Resource Functions that this composite Resource Function is composed of or include links to existing Resource Functions.

Example 1: In many cases, the response of the operation cannot be sent back synchronously, a “monitor” resource hyperlink is provided in the response.

See TM Forum Rest Design Guidelines for more information on asynchronous pattern and monitor resources.

REQUEST

POST api/resourceFunction
Content-type: application/json


```
{

    "role": "Backup Media Store",
```



```
    "place": {
      "href": "http://serverlocation:port/place/4980",
      "id": "4980"
    },
    "priority": "2",

    "resourceSpecification": {
      "id": "RS-6789",
      "href": "http://serverlocation:port/resourceSpecification/RS-6789"
    },
    "characteristic": [
      {
        "name": "bandwidth",
        "value": "100MB"
      }
    ],
    "resourceRelationship": [
      (Details of supporting resource functions or references to resource functi
ons depending of depth of information required)
       "oneOf":[{
             "$ref": "#/definitions/ResourceFunction”},
              {
               "$ref": "#/definitions/ResourceFunctionRef”
              }]

    ],
    "connectivity": [
      {
        "source": "http://serverlocation:port/resourceFunction/6789",
        "target": "http://serverlocation:port/resourceFunction/1234",
        "relationship": "AdjacentTo"
      }
    ],
    "relatedParty": [
      {
        "id": "1234",
        "href": "http://serverlocation:port/partyManagement/partyRole/1234",
        "role": "Admin"
```

```
      }
    ]

  }
```

RESPONSE

202 Accepted
Content-Type: application/json
Location: http://api/resourceFunction/17898

{
//same as in request
}
Link: <http:///api/resourceFunction/monitor/38>;rel-related;title=monitor
<http://api/resourceFunction/14>;rel=self,
< api/resourceFunction/14>;rel=canonical

Example see TMF REST Design Guidelines.

Example 2: The response can be sent back synchronously.

REQUEST

POST api/resourceFunction
Content-type: application/json

```
{


    "role": "Backup Media Store",
    "place": {
      "href": "http://serverlocation:port/place/4980",
      "id": "4980"
    },
    "priority": "2",


    "resourceSpecification": {
      "id": "RS-6789",
      "href": "http://serverlocation:port/resourceSpecification/RS-6789"
    },
```

```
    "characteristic": [
      {
        "name": "bandwidth",
        "value": "100MB"
      }
    ],
    "resourceRelationship": [
      (Details of supporting resource functions or references to resource functi
ons depending of depth of information required)
       "oneOf":[{
             "$ref": "#/definitions/ResourceFunction”},
              {
               "$ref": "#/definitions/ResourceFunctionRef”
              }]

    ],
    "connectivity": [
      {
        "source": "http://serverlocation:port/resourceFunction/6789",
        "target": "http://serverlocation:port/resourceFunction/1234",
        "relationship": "AdjacentTo"
      }
    ],
    "relatedParty": [
      {
        "id": "1234",
        "href": "http://serverlocation:port/partyManagement/partyRole/1234",
        "role": "Admin"
      }
    ]
  }
```

RESPONSE

201 Created
Content-Type: application/json
Location: http://api/resourceFunction/17898

```
{
```

```
    "id": "17898",
    "href": "http://serverlocation:port/resourceFunction/17898",
    "name": "CDN Cluster",
    "description": "CDN capability spread across multiple geographies",
    "type": "Content Delivery",
    "version": "1.2",
    "role": "Backup Media Store",
    "place": {
      "href": "http://serverlocation:port/place/4980",
      "id": "4980"
    },
    "autoModification": "scaleStorage",
    "priority": "2",
    "lifeCycleState": "planning",
    "lifeCycleSubState": "unknown",
    "schedule": [
      {
        "id": "SCH-78906",
        "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
      }
    ],
    "sap": [
      {
        "id": "SAP-49876",
        "href": "http://serverlocation:port/resourceSpecification/SAP-49876"
      }
    ],
    "resourceSpecification": {
      "id": "RS-6789",
      "href": "http://serverlocation:port/resourceSpecification/RS-6789"
    },
    "characteristic": [
      {
        "name": "bandwidth",
        "value": "100MB"
      }
    ],
```



```
    "resourceRelationship": [
      (Array of supporting resource functions that need to be created or referen
ces to resource functions that have already been created)
       "oneOf":[{
             "$ref": "#/definitions/ResourceFunction”},
              {
               "$ref": "#/definitions/ResourceFunctionRef”
              }]

    ],
    "connectivity": [
      {
        "source": "http://serverlocation:port/resourceFunction/6789",
        "target": "http://serverlocation:port/resourceFunction/1234",
        "relationship": "AdjacentTo"
      }
    ],
    "relatedParty": [
      {
        "id": "1234",
        "href": "http://serverlocation:port/partyManagement/partyRole/1234",
        "role": "Admin"
      }
    ]
  }
```

Link: http:// /api/resourceFunction/17898;rel-related;title=monitor

Example see TMF REST Design Guidelines.

PATCH API/RESOURCEFUNCTION/{ID}

This Uniform Contract operation is used to partially update the representation of a Resource Function.

The response of the operation can be sent back synchronously or not, in which case a “monitor” resource hyperlink is given in the response.

REQUEST

PATCH API/resourceFunction /{ID}
Content-type: application/json

```
{
                  "schedule": [
                    {
                      "id": "SC43891",
                      "href": "http://.."
                    }
                  ]
}
```

RESPONSE

200 OK
Content-Type: application/json

```
{
    "id": "17898",
    "href": "http://serverlocation:port/resourceFunction/17898",
    "name": "CDN Cluster",
    "description": "CDN capability spread across multiple geographies",
    "type": "Content Delivery",
    "version": "1.2",
    "role": "Backup Media Store",
    "place": {
      "href": "http://serverlocation:port/place/4980",
      "id": "4980"
    },
    "autoModification": {
    “name”:"scaleStorage",
    “value”:”scaleIn”},
    "priority": "2",
    "lifeCycleState": "planning",
    "lifeCycleSubState": "unknown",
    "schedule": [
      {
        "id": "SCH-78906",
        "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
```

```
      }
    ],
    "sap": [
      {
        "id": "SAP-49876",
        "href": "http://serverlocation:port/resourceSpecification/SAP-49876"
      }
    ],
    "resourceSpecification": {
      "id": "RS-6789",
      "href": "http://serverlocation:port/resourceSpecification/RS-6789"
    },
    "characteristic": [
      {
        "name": "bandwidth",
        "value": "100MB"
      }
    ],
    "resourceRelationship": [
      (Array of supporting resource functions that need to be created or referen
ces to resource functions that have already been created)
       "oneOf":[{
             "$ref": "#/definitions/ResourceFunction”},
              {
               "$ref": "#/definitions/ResourceFunctionRef”
              }]

    ],
    "connectivity": [
      {
        "source": "http://serverlocation:port/resourceFunction/6789",
        "target": "http://serverlocation:port/resourceFunction/1234",
        "relationship": "AdjacentTo"
      }
    ],
    "relatedParty": [
      {
        "id": "1234",
```

```
        "href": "http://serverlocation:port/partyManagement/partyRole/1234",
        "role": "Admin"
      }
    ]
}
```


Example see TMF REST Design Guidelines.

DELETE API/ RESOURCEFUNCTION /{ID}

This Uniform Contract operation is used to delete a Resource Function.

- The response of the operation can be sent back synchronously or not in case a “monitor” resource hyperlink is given in the response.
- Client can request some functions to be retained post the delete. This can be provided in the URL.

Behavior:

- Returns HTTP/1.1 status code 200(OK) or 202(Accepted) accepted if the request was successful.
- Returns a monitor object in case of 202(Accepted) that can be queried to get back the status of the object.

REQUEST
DELETE API/RESOURCEFUNCTION/{ID}?retainFunctions=”129087,39098”
RESPONSE

202 Accepted
Content-Type: application/json
 	
{
//same as in request
}
Link: http:// /api/resourceFunction/monitor/38;rel-related;title=monitor

Example see TMF REST Design Guidelines.

GET /api/ RESOURCEFUNCTION /HEAL{ID}

This Uniform Contract operation is used to retrieve the representation of the “HEAL” task resource. The resource can be created against a composite of atomic Resource Function.

Description:

- This operation is used to retrieve the Heal task resource information including the ID
- Attribute selection is enabled.

Behavior:

- Status code 200 – if the request was successful
- Status code 404 Not found – supplied ID does not match a known Heal resource

REQUEST

```
GET /API/resourceFunction/heal/490987
Accept: application/json
```

RESPONSE

```
200
Content-Type: application/json
```


```
{
  "id": "490987",
  "href": "http://serverlocation:port/resourceFunction/490987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "cause": "SLA violation",
  "degreeOfHealing": "Complete - Restore to state before failure",
  "healAction": "/conf/lab/healfast.sh",
  "healPolicy": {
    "id": "Pol-3490",
    "href": "http://serverlocation:port/policy/Pol-3490"
  },
  "startTime": "2017:11:7:15:53:10Z",
  "additionalParms": [
    {
      "name": "string",
      "value": "string"
```



```
    }
  ],
  "healStatus": "In Progress"
}
```


Example see TMF REST Design Guidelines.

POST API/ RESOURCEFUNCTION /HEAL

This Uniform Contract operation is used to create a Heal task resource. This is an operation to request heal of a Resource Function.

Behavior:

- Returns HTTP/1.1 status code 202 accepted if the request was successful.
- A Heal object will be returned that can be queried to get the latest status of the operation.

ID Management:

ID is generated by the operation.

REQUEST

POST API/resourceFunction/heal
Content-type: application/json


```
{
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "cause": "SLA violation",
  "degreeOfHealing": "Complete - Restore to state before failure",
  "healAction": "/conf/lab/healfast.sh",
  "healPolicy": {
    "id": "Pol-3490",
    "href": "http://serverlocation:port/policy/Pol-3490"
  },
```



```
  "startTime": "2017:11:7:15:53:10Z",
  "additionalParms": [
    {
      "name": "string",
      "value": "string"
    }
  ],
  }
```


RESPONSE

202 Accepted
Content-Type: application/json


```
{
  "id": "490987",
  "href": "http://serverlocation:port/resourceFunction/heal/490987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "cause": "SLA violation",
  "degreeOfHealing": "Complete - Restore to state before failure",
  "healAction": "/conf/lab/healfast.sh",
  "healPolicy": {
    "id": "Pol-3490",
    "href": "http://serverlocation:port/policy/Pol-3490"
  },
  "startTime": "2017:11:7:15:53:10Z",
  "additionalParms": [
    {
      "name": "string",
      "value": "string"
    }
  ],
  "healStatus": "In Progress"
}
```


Example see TMF REST Design Guidelines.

PATCH API/ RESOURCEFUNCTION/HEAL/{ID}

This Uniform Contract operation is used to partially update the representation of a Heal task resource.

The response of the operation can be sent back synchronously or not. As the heal action may already be in progress then the patch can be applied on a best effort basis. It may be unsuccessful if the heal action has progressed beyond a certain point.

REQUEST

PATCH API/resourceFunction/heal/{ID}
Content-type: application/json

```
{
  "id": "490987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "healAction": "/conf/lab/healfastnew.sh",
  }
```

RESPONSE

200 OK
Content-Type: application/json

```
{
  "id": "490987",
  "href": "http://serverlocation:port/resourceFunction/heal/490987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "cause": "SLA violation",
  "degreeOfHealing": "Complete - Restore to state before failure",
  "healAction": "/conf/lab/healfastnew.sh",
  "healPolicy": {
    "id": "Pol-3490",
    "href": "http://serverlocation:port/policy/Pol-3490"
  },
  "startTime": "2017:11:7:15:53:10Z",
  "additionalParms": [
    {
      "name": "string",
```

```
      "value": "string"
    }
  ],
  "healStatus": "In Progress"
}
```


Example see TMF REST Design Guidelines.

DELETE API/ RESOURCEFUNCTION /HEAL/{ID} -

This Uniform Contract operation is used to delete a Heal task resource. This is where a heal may no longer be required on a Resource Function because the problem may have been resolved. This operation is performed on a best effort basis and will fail if the heal action has progressed beyond a certain point.

Behavior:

- Returns HTTP/1.1 status code 200 if the request was successful.

REQUEST
DELETE API/resourceFunction/heal/490987
RESPONSE
204 No Content Content-Type: application/json

Example see TMF REST Design Guidelines.

GET /api/ RESOURCEFUNCTION /SCALE{ID}

This Uniform Contract operation is used to retrieve the representation of the “SCALE” task resource.

Description:

- This operation is used to retrieve the Scale task resource information including the ID
- Attribute selection is enabled.

Behavior:

- Status code 200 – if the request was successful
- Status code 404 Not found – supplied ID does not match a known Scale resource.

REQUEST

```
GET /API/resourceFunction/scale/180987
Accept: application/json
```

RESPONSE

```
200
Content-Type: application/json
```


```
{
  "id": "180987",
  "href": "http://serverlocation:port/resourceFunction/scale/180987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "type": "Scale Out",
  "aspectId": "Quick Access Memory",
  "numberOfSteps": "2",
  "schedule": [
    {
      "id": "SCH-78906",
      "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
    }
  ],
  "scaleStatus": "In Progress"
}
```


Example see TMF REST Design Guidelines.

POST API/ RESOURCEFUNCTION /SCALE

This Uniform Contract operation is used to create a Scale task resource. This is an operation to request Scale of a Resource Function.

Behavior:

- Returns HTTP/1.1 status code 202 accepted if the request was successful.
- A Scale object will be returned that can be queried to get the latest status of the operation.

ID Management:

ID is generated by the operation.

REQUEST

POST API/resourceFunction/scale
Content-type: application/json

```
{
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "type": "Scale Out",
  "aspectId": "Quick Access Memory",
  "numberOfSteps": "2",
  "schedule": [
    {
      "id": "SCH-78906",
      "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
    }
  ],
}
```

RESPONSE

202 Accepted
Content-Type: application/json

```
{
  "id": "180987",
  "href": "http://serverlocation:port/resourceFunction/scale/180987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "type": "Scale Out",
  "aspectId": "Quick Access Memory",
  "numberOfSteps": "2",
  "schedule": [
    {
      "id": "SCH-78906",
      "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
    }
  ],
  "scaleStatus": "In Progress"
}
```


Example see TMF REST Design Guidelines.

PATCH API/ RESOURCEFUNCTION /SCALE/{ID}

This Uniform Contract operation is used to partially update the representation of a Scale task resource.

The response of the operation can be sent back synchronously or not. As the Scale action may already be in progress then the patch can be applied on a best effort basis. It may be unsuccessful if the Scale action has progressed beyond a certain point.

REQUEST

PATCH API/resourceFunction/scale/{ID}
Content-type: application/json


```
{
  "id": "180987",
  "href": "http://serverlocation:port/resourceFunction/17898",
```



```
  "numberOfSteps": "4",
  }
```

RESPONSE

201
Content-Type: application/json


```
{
  "id": "180987",
  "href": "http://serverlocation:port/resourceFunction/scale/180987",
  "resourceFunctionHref": "http://serverlocation:port/resourceFunction/17898",
  "type": "Scale Out",
  "aspectId": "Quick Access Memory",
  "numberOfSteps": "4",
  "schedule": [
    {
      "id": "SCH-78906",
      "href": "http://serverlocation:port/resourceSpecification/SCH-78906"
    }
  ],
  "scaleStatus": "In Progress"
}
```


Example see TMF REST Design Guidelines.

DELETE API/ RESOURCEFUNCTION /SCALE/{ID}

This Uniform Contract operation is used to delete a Scale task resource. This is where a Scale may no longer be required because the problem may have been resolved. This operation is performed on a best effort basis and will fail if the Scale action has progressed beyond a certain point.

Behavior:

- Returns HTTP/1.1 status code 200 if the request was successful.

REQUEST
DELETE API/resourceFunction/scale/{ID}
RESPONSE
204 No Content Content-Type: application/json

Example see TMF REST Design Guidelines.

GET /api/ RESOURCEFUNCTION /MIGRATE{ID}

This Uniform Contract operation is used to retrieve the representation of the "MIGRATE" task resource.

Description:

- This operation is used to retrieve the Migrate task resource information including the ID
- Attribute selection is enabled.

Behavior:

- Status code 200 – if the request was successful
- Status code 404 Not found – supplied ID does not match a known Migrate task resource

REQUEST
GET /API/ resourceFunction/migrate/80987 Accept: application/json
RESPONSE
202 Accepted Content-Type: application/json { "id": "80987",

```
  "href": "http://serverlocation:port/resourceFunction/migrate/80987",
  "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "adminStateModification": "locked",
  "sapsToRemove": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "sapsToAdd": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "priority": "1",
  "startTime": "2017:11:7:15:53:10Z",
  "completionMode": "bestEffort",
  "place": {
    "href": "http://serverlocation:port/place/4980",
    "id": "4980"
  },
  "characteristics": [
    {
      "name": "string",
      "valueType": "string",
      "value": "string"
    }
```



```
  ],
  "migrateStatus": "In Progress"
}
```

Example see TMF REST Design Guidelines.

POST API/ RESOURCEFUNCTION /MIGRATE

This Uniform Contract operation is used to create a Migrate task resource. This is an operation to request Migration of a Resource Function.

Behavior:

- Returns HTTP/1.1 status code 202 accepted if the request was successful.
- A Migrate object will be returned that can be queried to get the latest status of the operation.

ID Management:

ID is generated by the operation.

REQUEST

POST API/resourceFunction/migrate
Content-type: application/json

```
{
   "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "adminStateModification": "locked",
  "sapsToRemove": [
    {
      "id": "SAP-49876",
```

```
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "sapsToAdd": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "priority": "1",
  "startTime": "2017:11:7:15:53:10Z",
  "completionMode": "bestEffort",
  "place": {
    "href": "http://serverlocation:port/place/4980",
    "id": "4980"
  },
  "characteristics": [
    {
      "name": "string",
      "valueType": "string",
      "value": "string"
    }
  ],
  "migrateStatus": "In Progress"
              }
```

RESPONSE

202 Accepted
Content-Type: application/json

```
{
```



```
  "id": "80987",
  "href": "http://serverlocation:port/resourceFunction/migrate/80987",
  "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "adminStateModification": "locked",
  "sapsToRemove": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "sapsToAdd": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "priority": "1",
  "startTime": "2017:11:7:15:53:10Z",
  "completionMode": "bestEffort",
  "place": {
    "href": "http://serverlocation:port/place/4980",
    "id": "4980"
  },
  "characteristics": [
    {
      "name": "string",
      "valueType": "string",
      "value": "string"
```



```
    }
  ],
  "migrateStatus": "In Progress"
              }
```

Example see TMF REST Design Guidelines.

PATCH API/ RESOURCEFUNCTION /MIGRATE/{ID}

This Uniform Contract operation is used to partially update the representation of a Migrate task resource.

The response of the operation can be sent back synchronously or not. As the Migrate action may already be in progress then the patch can be applied on a best effort basis. It may be unsuccessful if the Migrate action has progressed beyond a certain point.

REQUEST

PATCH API/resourceFunction/migrate/{ID}
Content-type: application/json

```
{
  "id": "80987",
  "href": "http://serverlocation:port/resourceFunction/17898",

  "place": {
    "href": "http://serverlocation:port/place/5670",
    "id": "5670"
  }
}
```

RESPONSE

201 OK
Content-Type: application/json

```
{
  "id": "80987",
  "href": "http://serverlocation:port/resourceFunction/migrate/80987",
  "resourceFunctionHref":
"http://serverlocation:port/resourceFunction/17898",
  "adminStateModification": "locked",
  "sapsToRemove": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "sapsToAdd": [
    {
      "id": "SAP-49876",
      "href": "http://serverlocation:port/resourceSpecification/SAP-
49876"
    }
  ],
  "priority": "1",
  "startTime": "2017:11:7:15:53:10Z",
  "completionMode": "bestEffort",
  "place": {
    "href": "http://serverlocation:port/place/5670",
    "id": "5670"
  },
  "characteristic": [
    {
      "name": "string",
      "value": "string"
```



```
    }
  ],
  "migrateStatus": "In Progress"
}
```

Example see TMF REST Design Guidelines.

DELETE API/ RESOURCEFUNCTION /MIGRATE/{ID} -

This Uniform Contract operation is used to delete a Migrate task resource. This is where a Migrate may no longer be required on a Resource Function. This operation is performed on a best effort basis and will fail if the Migrate action has progressed beyond a certain point.

Behavior:

- Returns HTTP/1.1 status code 200 if the request was successful.

REQUEST
DELETE API/resourceFunction/migrate/{ID}
RESPONSE
200 OK Content-Type: application/json

Example see TMF REST Design Guidelines.

API NOTIFICATION TEMPLATES

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines reproduced below.

REGISTER LISTENER POST /HUB

Description:

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

Behavior:

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

REQUEST
POST /api/hub Accept: application/json {"callback": "http://in.listener.com"}
RESPONSE
201 Content-Type: application/json Location: /api/hub/42 {"id":"42","callback":"http://in.listener.com","query":null}

UNREGISTER LISTENER DELETE HUB/{ID}

Description:

Clears the communication endpoint address that was set by creating the Hub.

Behavior:

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

REQUEST
DELETE /api/hub/{id} Accept: application/json
RESPONSE
204

PUBLISH {EVENTTYPE} POST /LISTENER

Description:

Provide the Event description

Behavior:

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

REQUEST

```
POST /client/listener
Accept: application/json

{

   "event": {

   EVENT BODY
         },
   "eventType": "eventType"
}
```

RESPONSE
201 Content-Type: application/json

Example see TMF REST Design Guidelines.

RELEASE HISTORY

Release Number	Date	Release led by:	Description
Release 17.0.0 Version 0.2.0	25-May-17	Milind Bhagwat BT plc milind.2.bhagwat@bt.com	Draft Version 0.2 of the Document produced after the initial review
Release 17.0.0 Version 0.2.1	05-Jul-17	Alan Pope TM Forum apope@tmforum.org	Draft Version 0.2.1 Document Title changed from 'Resource Function Configuration and Activation' to 'Resource Function Activation and Configuration'
Release 17.0.1 Version 0.2.2	20-Nov-17	Adrienne Walcott	Updated to reflect TM Forum Approved Status
Release 17.5.0 Version 0.3.0	17-Nov-17	Milind Bhagwat BT plc milind.2.bhagwat@bt.com	Draft Version 0.3 of the document. Added 2 sample use cases to include requirements AP-971 and AP-972 Added valueType attribute to characteristics
Release 17.5.0 Version 0.3.1	11-Dec-17	Milind Bhagwat BT plc milind.2.bhagwat@bt.com	Updated document based on review comments received
Release 17.5.0 Version 0.3.2	16-Jan-18	Alan Pope TM Forum apope@tmforum.org	Minor edits and formatting for release.