

# Device Management over REST for Resource Configuration

Ericsson Dynamic Activation 1

---

## INTERFACE DESCRIPTION

**Copyright**

© Ericsson AB 2017. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

**Disclaimer**

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

**Trademark List**

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Purpose and Scope	1
1.2	Target Group	1
1.3	Typographic Conventions	1
1.4	Prerequisites	1
1.5	Operations	2
1.6	Web Service Interface	3
<b>2</b>	<b>Authentication</b>	<b>5</b>
2.1	Example	5
<b>3</b>	<b>Device Management Data</b>	<b>7</b>
<b>4</b>	<b>Device Management Operations</b>	<b>11</b>
4.1	/scm-rest/device-repository/admin-states	11
4.2	/scm-rest/device-repository/device-categories	11
4.3	/scm-rest/device-repository/devices	12
4.4	/scm-rest/device-repository/devices/{identifier}	16
4.5	/scm-rest/device-repository/devices/{identifier}/config-sync	19
<b>5</b>	<b>Faults or Errors</b>	<b>21</b>
	<b>Reference List</b>	<b>23</b>





# 1 Introduction

This document describes the Device Management Representational State Transfer (REST) API. The API offers the possibility to:

- Add, remove, update, and get devices in the Device Repository.
- Management of device information like login credentials, device type and the use of Resource Configuration features like Candidate Store and Configuration Validation.
- Trigger Device Discovery and Reconciliation.

For more information about the Device Repository and device related functionality, refer to *Function Specification Resource Configuration*, Reference [1] and *User Guide for Resource Configuration*, Reference [2].

## 1.1 Purpose and Scope

All functionality exposed over the REST API for Device Management is also available in the Resource Configuration GUI. For smaller solutions of Resource Configuration it is convenient to add and remove devices manually in the system, using this GUI. When the number of devices is increasing, it is recommended to automate the introduction of new devices in the solution. The REST API described in this document is then needed to manage the devices in Resource Configuration from another system.

## 1.2 Target Group

The target group for this document is as follows:

- System Integrator

For more information about the different target groups, see *Library Overview*, Reference [3].

## 1.3 Typographic Conventions

Typographic conventions are described in *Library Overview*<sup>3</sup>, Reference [3].

## 1.4 Prerequisites

To use this document fully, users must meet the following prerequisites:



- Basic knowledge about the Dynamic Activation product
- Knowledge about REST.
- Knowledge about json.

For information about Resource Configuration in Ericsson Dynamic Activation (EDA), refer to *Function Specification Resource Configuration*, Reference [1].

## 1.5 Operations

The following HTTP request methods are used:

- GET

The GET method requests a representation of the specified resource. Requests using GET only retrieves data and have no other effect.

- POST

The POST method requests that the server accept the entity enclosed in the request as a new subordinate of the web resource identified by the URI, or an item to add to a database.

- PUT

The PUT method requests that the enclosed entity be stored under the supplied URI. If the URI refers to an already existing resource, it is modified.

- DELETE

The DELETE method deletes the specified resource.

See the following table for all available URLs and corresponding operations.

*Table 1 URLs and Corresponding Operations*

URL	Operations			
	GET	POST	PUT	DELETE
/scm-rest/device-repository/admin-states	X			
/scm-rest/device-repository/device-categories	X	X		
/scm-rest/device-repository/devices	X	X		
/scm-rest/device-repository/devices/{identifier}	X		X	X
/scm-rest/device-repository/devices/{identifier}/config-sync		X		

All requests and responses use the Content-Type application/json.



## 1.6 Web Service Interface

Resource Configuration provides Web Application Description Language (WADL) file to use as a base when creating REST interface.

The interface.zip file containing the WADL file can be found in /home/dveinstaller/da/. It is also possible to download the zip file and store it in an appropriate area by following below instruction:

1. Save the zip file, [Dynamic Activation WSDL and XSD files.zip](#), to a local folder.
2. Unpack the zip file.
3. Go to \EDAinterface\webservice\_provisioning\_rest\wadl\Applications\Resource\_Configuration.







## 2 Authentication

All the requests in this document must include the HTTP Basic authentication header to provide username and password. The basic authentication string uses Base64 encoding.

### 2.1 Example

This section contains a complete example of an authorization header that can be reused in all operations mentioned later in this document.

```
GET /scm-rest/device-repository/devices/device1/ HTTP/1.1
Host: 10.0.0.2:8383
Authorization: Basic YWRtaW46YWRtaW4=
Content-Type: application/json
```

*Example 1 Complete Header Example*





### 3 Device Management Data

This chapter describes the data attributes related to the device. For more information about each operation, see Section 4 on page 11.

*Table 2 Data Attributes*

Property	Type	Occurrence		Description
		POST	PUT	
masterIdentifier	String	M	N/A	The unique identifier of the device (not possible to update).
identifiers	Array	O	O	Additional unique identifiers of each device.
adminState	Array	M	M	Specifies the availability of the device, if set to the ACTIVE it is possible to provision the device otherwise the device cannot be provisioned.  This parameter can be in OPERATION FAILED if an operation failed, for example a restore, then the user has to set it back to ACTIVE manually once the problem has been resolved.
modes	Array	O	O	Device modes.
CONFIGURATION_VALIDATION	-	O	O	Specifies whether the Configuration validation is enabled or not for the specific device and what action to take if there is an inconsistency between configuration.  If enabled, a check is made that the actual configuration on the device matches the configuration last provisioned to the device. It must also be specified what action to take if the configurations do not match.
CANDIDATE_STORE	-	O	O	Specifies whether the Candidate store function is enabled or not for the specific device. If Candidate Store is enabled, the configuration for the selected device is stored in Candidate store before sent out to the device.



Property	Type	Occurrence		Description
		POST	PUT	
subModes	Array	O	O	<p>Sub-modes property is only applicable for CONFIGURATION_VALIDATION. The following operations are available:</p> <ul style="list-style-type: none"><li>• DISPLAY ERROR: The user will be notified with details of the discrepancies between the configurations.</li><li>• UPDATE DEVICE: Dynamic Activation will update the device with the stored device configuration.</li><li>• UPDATE REPOSITORY: Dynamic Activation will update the Resource Configuration device repository with the device configuration. If the configurations match, the device is provisioned with the new or updated configuration. If the configurations do not match, the action that was previously selected will occur.</li></ul>
types	Array	O	O	<p>Specifies what type the device belong to. A device can belong to many types. This information is also used to decide which template to use when generating the southbound commands to the device.</p>
accessPoint	Array	M	M	<p>Access points for the device.</p>
protocol	String	M	M	<p>Specifies what protocol to use for communication with the device.</p> <p>Available protocols are SSH_CLI, SSH_NETCONF, and SNMP.</p>
addresses	String	M	M	<p>The address to the device according to format <code>&lt;IP_address&gt;:&lt;port&gt;</code>, for example 10.0.0.0:22.</p>
parameters	key/value	O	O	<p>Specifies a list of key value objects to set in the device. For example, providing a default parameter value.</p>



Property	Type	Occurrence		Description
		POST	PUT	
credentials	key/value	M	M	<p>Specifies the authentication credentials to use for communication with the device. These are provided as key value objects. The credentials must be recognized by the device and the keys "username" and "password" must be present.</p> <p>Optional Additional Parameter - Name can be added. For example, providing a default parameter value.</p>
info	key/value	O	O	<p>This is a field used by the system to inform about incidents or progress of long running tasks. For example a restore that failed, or a reconciliation action that changed status.</p> <p>The valid keys are:</p> <ul style="list-style-type: none"> <li>• errorMsg: Contains information about a current incident related to this device. The field is cleared if Admin state is changed from OPERATION FAILED to ACTIVE.</li> <li>• RECONCILIATION: Shows the progress of a reconciliation action. Predefined values are: STARTED, ONGOING, FINISHED, and FAILED.</li> <li>• RECONCILIATION_ERROR: Shows a detailed error message related to the FAILED state of the RECONCILIATION.</li> </ul>
description	String	O	O	Free text where to enter additional information about the specific device.
category	String	O	O	Specifies what category the device belongs to.





## 4 Device Management Operations

### 4.1 /scm-rest/device-repository/admin-states

#### 4.1.1 GET

This operation fetches all valid admin states.

**Parameters:**

N/A

**Request example:**

```
https://10.0.0.2:8383/scm-rest/device-repository/admin-states
```

**Response body example:**

```
[  
  "IDLE",  
  "INSTALLED",  
  "UNKNOWN",  
  "NOT INSTALLED",  
  "INACTIVE",  
  "ACTIVE"  
]
```

### 4.2 /scm-rest/device-repository/device-categories

#### 4.2.1 GET

This method gets all valid device categories.

**Parameters:**

This operation contains no parameters.

**Request example:**

```
https://10.0.0.2:8383/scm-rest/device-repository/device-categories
```

**Response body example:**



```
[
  "OTHER",
  "CE",
  "Proxy",
  "Jump Server",
  "PE",
  "FIREWALL"
]
```

### 4.2.2 POST

This method adds new device category.

**Parameters:**

N/A

**Request example:**

The request body `{"category": "MY_CATEGORY"}` is sent to the URL `https://10.0.0.2:8383/scm-rest/device-repository/device-categories`.

**Response body example:**

```
{"category": "MY_CATEGORY"}
```

## 4.3 /scm-rest/device-repository/devices

This operation consists of the methods described below.

### 4.3.1 GET

Get a page of devices by filtering criteria.

**Parameters:**

The following parameters can be used when filtering:

*Table 3 Device List Filter Parameters*

Name	Type	Occurrence	Description
pageIndex	query	Mandatory	The page to retrieve.





Name	Type	Occurrence	Description
pageSize	query	Mandatory	Specifies how many device entries the page should contain.
searchPattern	query	Optional	Devices with matching identifiers will be returned.
categories	query	Optional	Devices with matching categories will be returned.
types	query	Optional	Devices with matching types will be returned.
protocols	query	Optional	Devices with matching protocol will be returned.
modes	query	Optional	Devices with matching modes will be returned.
adminStates	query	Optional	Devices with matching admin states will be returned.

**Request body:**

N/A

**Request example:**

```
https://10.0.0.2:8383/scm-rest/device-repository/devices?pageIndex=1&pageSize=50&adminStates[]=ACTIVE&categories[]=CE&types[]=Junos_Template_CLI&protocols[]=SSH_CLI&modes[]=CANDIDATE+STORE&searchPattern=device
```

**Response body example:**

```
{
  "totalNrOfPages": 1,
  "currentPage": 1,
  "listItems": [
    {
      "masterIdentifier": "device1",
```



```
    "identifiers": [],
    "adminState": "ACTIVE",
    "modes": [
      "CANDIDATE STORE",
      "CONFIGURATION VALIDATION"
    ],
    "subModes": [
      "UPDATE REPOSITORY"
    ],
    "types": [
      "Junos_Template_CLI"
    ],
    "accessPoints": [
      {
        "protocol": "SSH_CLI",
        "address": "10.0.0.3",
        "parameters": {
          "parameterKey1": "parameterValue1"
        },
        "credentials": {
          "username": "admin",
          "password": "admin"
        }
      }
    ],
    "info": {
      "RECONCILIATION": "FAILED"
      "RECONCILIATION_ERROR": "Unexpected error - detailed"
    },
    "description": "This is the description of the device",
    "candidateConfigCount": 0,
    "category": "CE"
  }
]
```

### 4.3.2 POST

This method adds a new device to the system.

**Parameters:**

N/A

**Request body example:**

```
{
  "masterIdentifier": "device1",
  "adminState": "ACTIVE",
  "types": ["Junos_Template_CLI"],
```



```

        "parameters": {"parameterKey1": "parameterValue1"},
        "category": "CE",
        "modes": [
            "CANDIDATE STORE",
            "CONFIGURATION VALIDATION"
        ],
        "subModes": ["UPDATE REPOSITORY"],
        "description": "This is the description of the device",
        "accessPoints": [
            {
                "credentials": {
                    "username": "admin",
                    "password": "admin"
                },
                "parameters": {},
                "address": "10.0.0.3:8000",
                "protocol": "SSH_CLI"
            }
        ]
    }
}

```

#### Request example:

`https://10.0.0.2:8383/scm-rest/device-repository/devices`

#### Response body example:

```

{
    "identifiers": [],
    "modes": [
        "CANDIDATE STORE",
        "CONFIGURATION VALIDATION"
    ],
    "subModes": [
        "UPDATE REPOSITORY"
    ],
    "types": [
        "Junos_Template_CLI"
    ],
    "parameters": {
        "parameterKey1": "parameterValue1"
    },
    "accessPoints": [
        {
            "parameters": {},
            "credentials": {
                "password": "admin",
                "username": "admin"
            }
        }
    ]
}

```



```
        "protocolParameters": {},
        "protocol": "SSH_CLI",
        "address": "10.0.0.3:8000"
    },
    "info": {},
    "masterIdentifier": "device1",
    "adminState": "ACTIVE",
    "description": "This is the description of the device",
    "candidateConfigCount": null,
    "category": "CE"
}
```

## 4.4 /scm-rest/device-repository/devices/{identifier}

This operation consists of the methods described below.

### 4.4.1 GET

This method gets the specific device by specifying deviceId.

#### Parameters:

This operation contains a path parameter which sends the request as a string Identifier.

Table 4 Device Parameter

Name	Type	Occurrence	Description
identifier	path	Mandatory	The identity of the device to retrieve

#### Request example:

```
https://10.0.0.2:8383/scm-rest/device-repository/devices/device1
```

#### Response body example:

```
{
  "identifiers": [],
  "modes": [
    "CANDIDATE STORE",
    "CONFIGURATION VALIDATION"
  ],
  "subModes": [
    "UPDATE REPOSITORY"
  ],
}
```



```

"types": [
  "Junos_Template_CLI", "IOS_Template"
],
"parameters": {"parameterKey1": "parameterValue1"},
"accessPoints": [
  {
    "parameters": {
      "parameterKey1": "parameterValue1"
    },
    "credentials": {
      "password": "admin",
      "username": "admin"
    },
    "protocolParameters": {},
    "protocol": "SSH_CLI",
    "address": "10.0.0.3:8000"
  }
],
"info": {
  "RECONCILIATION": "FINISHED"
},
"masterIdentifier": "device1",
"adminState": "ACTIVE",
"description": "This is the description of the device",
"candidateConfigCount": null,
"category": "CE"
}

```

For a more detailed description of the different parameters, see Section 3 on page 7.

#### 4.4.2 PUT

This method updates information about a specific device with deviceId.

##### Parameters:

This operation contains a path parameter which sends the request as a string Identifier.

Table 5 Device Parameter

Name	Type	Occurrence	Description
Identifier	path	Mandatory	The identity of the device to modify

##### Request example:



https://10.0.0.2:8383/scm-rest/device-repository/devices/device1

**Request body example:**

```
{
  "identifiers": [],
  "adminState": "ACTIVE",
  "types": [
    "Junos_Template_CLI",
    "IOS_Template"
  ],
  "category": "CE",
  "modes": ["CONFIGURATION VALIDATION"],
  "subModes": ["UPDATE REPOSITORY"],
  "description": "This is the description of the device",
  "parameters": {"parameterKey1": "parameterValue1"},
  "accessPoints": [
    {
      "protocol": "SSH_CLI",
      "address": "10.0.0.3",
      "parameters": {},
      "credentials": {
        "username": "admin",
        "password": "admin"
      }
    }
  ]
}
```

**Response body:**

N/A

### 4.4.3 DELETE

This method removes a specific device.

**Parameters:**

This operation contains a path parameter which sends the request as a string Identifier.

Table 6 Device Parameter

Name	Type	Occurrence	Description
Identifier	path	Mandatory	The identity of the device to delete.

**Request example:**

```
https://10.0.0.2:8383/scm-rest/device-repository/devices/device1
```

**Response body:**

N/A

## 4.5 /scm-rest/device-repository/devices/{identifier}/config-sync

This operation consists of the methods described below.

### 4.5.1 POST

This method triggers a device reconciliation to align the Device Repository with the provisioned devices.

**Parameters:**

This following parameters can be used when reconciling a device.

*Table 7 Device Parameter*

Name		Type	Occurrence	Description
Identifier		path	Mandatory	The identity of the device to reconcile.
syncType		string	Mandatory	
	UPDATE DEVICE	-	-	Specifies that the Device should be aligned with the Repository
	UPDATE REPOSITORY	-	-	Specifies that the Repository should be aligned with the Device.
cached		boolean	Mandatory	Specifies that the system should rebuild the cache of the network data. Must be "false".

**Request example:**

```
https://10.0.0.2:8383/scm-rest/device-repository/devices/device1/config-sync
```

**Request body example:**

```
{
```



```
"syncType": "UPDATE_DEVICE",  
"cached": "false"  
}
```

### **Response body**

N/A





## 5 Faults or Errors

A REST error message consists of an error code and a message describing the error.

The following table covers Device Management REST error codes. They can appear in any REST response.

*Table 8 Device Management REST Error Codes*

Error Code	Error Code Description
400	Bad request, the request was invalid.
401	Unauthorized, due to authentication header is missing, or username or password was invalid.
404	Not found, the requested resource was not found on the server.
500	Internal server error, something went wrong when processing the request.

The following is an example of a REST error message:

```
{
  "errorCode": 404,
  "description": "Not found."
}
```

*Example 2 REST Error Message*





## Reference List

### Ericsson Documents

- [1] *Function Specification Resource Configuration*, 19/155 17-CSH 109 628Uen
- [2] *User Guide for Resource Configuration*, 11/1553-CSH 109 62 Uen
- [3] *Library Overview*, 18/1553-CSH 109 62 Uen