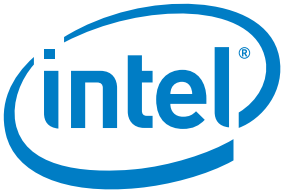


# Intel® Rack Scale Design (Intel® RSD) POD Manager (PODM) Representational State Transfer (REST)

**API Specification**
**Software v2.3.2**

***September 2018***

***Revision 003US***

Document Number: 337204-003US

# Contents



## Figures



## Tables

# Revision History

| Revision | Description | Date |
|---|---|---|
| 003US | Intel® RSD minor software release v2.3.2:<br>• Section 4.9.1.1 inserted missing bracket in the response code<br>• Added Section 6.1 introduction | September 2018 |
| 002US | Intel® RSD interim software release v2.3.1:<br>• Updated Section 4.10.1.2 | July 2018 |
| 001US | Initial release for software release v2.3 | March 2018 |

§

# 1.0 Introduction

## 1.1 Scope

This document contains information about the Intel® Rack Scale Design (Intel® RSD) POD Manager (PODM) Representational State Transfer (REST) API, which was designed and implemented for Intel® Rack Scale Design Software v2.3.2.

The interfaces specified are based on the *Distributed Management Task Force's (DMTF) Redfish* Interface Specification and schema*, v2016.3 and *Swordfish* Scalable Storage Management API Specification* v1.0.4. The schema has been enhanced with the extended common fabric model, which is a work in progress at the *Scalable Platforms Management Forum (SPMF)*; refer to Table 2.

## 1.2 Intended Audience

The intended audience for this document is designers and engineers working with the Intel® Rack Scale Design Software v2.3.2 release.

## 1.3 Conventions

The key words/phrases "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in *Key Words for Use in RFCs to Indicate Requirement Levels, March 1997*, RFC2119, (refer to Table 2).

## 1.4 Notes and Symbol Conventions

Symbol and note conventions are similar to typographical conventions used in the *Cloud Infrastructure Management Interface (CIMI) specification*. Notation used in JavaScript* Object Notation (JSON*) serialization description:

- Values in italics indicate data types instead of literal values.
- Characters are appended to items to indicate cardinality:
  - "?" (0 or 1)
  - "*" (0 or more)
  - "+" (1 or more)
- Vertical bars, "|", denote choice. For example, "a|b" means a choice between "a" and "b"
- Parentheses, "(" and ")", are used to indicate the scope of the operators "?", "*", "+" and "|"
- Ellipses (i.e., "...") indicate points of extensibility

***Note:*** The lack of ellipses doesn't mean no extensibility point exists; rather it's just not explicitly called out.

## 1.5 Terminology

**Table 1 Terminology**

| Term | Definition |
|---|---|
| API | Application Program Interface |

| Term | Definition |
|---|---|
| BMC | Baseboard Management Controller |
| CIMI | Cloud Infrastructure Management Interface |
| CPU | Central Processing Unit |
| HTTP | Hypertext Transfer Protocol |
| iSCSI | Internet Small Computer Systems Interface. Specification available at RFC 3720 and RFC 3721 |
| iQN | iSCSI Qualified Name format defined in RFC 3720 and RFC 3721 |
| JSON* | JavaScript* Object Notation |
| NIC | Network Interface Card |
| NQN | NVMe Qualified Name described in Section 7.9 of the NVMe Base specification, available at *http://www.nvmexpress.org/specifications* |
| NVMe* | Non-volatile Memory Express*. Specification available at *http://www.nvmexpress.org/* |
| OCCI | Open Cloud Computing Interface |
| OEM | Original Equipment Manufacturer |
| OData | Open Data Protocol |
| OVF | Open Virtualization Format |
| PCIe* | Peripheral Component Interconnect express* |
| PODM | POD Manager |
| PSME | Pooled System Management Engine |
| TPM | Trusted Platform Module |
| REST | Representational State Transfer |
| RSD | Rack Scale Design |
| URI | Uniform Resource Identifier |
| UUID | Universally Unique Identifier |
| XML | Extensible Markup Language |

## 1.6 References

**Table 2. Reference Documents and Resources**

| Doc ID | Title | Location |
|---|---|---|
| 337196 | *Intel® Rack Scale Design (Intel® RSD) Pooled System Management Engine (PSME) User Guide Software v2.3.2* | *http://www.intel.com/intelRSD* |
| 337197 | *Intel® Rack Scale Design (Intel® RSD) Conformance and Software Reference Kit Getting Started Guide Software v2.3.2* | |
| 337198 | *Intel® Rack Scale Design (Intel® RSD) POD Manager (PODM) Release Notes Software v2.3.2* | |
| 337199 | *Intel® Rack Scale Design (Intel® RSD) POD Manager (PODM) Representational State Transfer (REST) User Guide Software v2.3.2* | |
| 337200 | *Intel® Rack Scale Design (Intel® RSD) Pooled System Management Engine (PSME) Release Notes Software v2.3.2* | |
| 337201 | *Intel® Rack Scale Design (Intel® RSD) Firmware Extension Specification Software v2.3.2* | |
| 337202 | *Intel® Rack Scale Design (Intel® RSD) Storage Services API Specification Software v2.3.2* | |
| 337203 | *Intel® Rack Scale Design (Intel® RSD) Architecture Specification Software v2.3.2* | |
| 337205 | *Intel® Rack Scale Design (Intel® RSD) Rack Management Module (RMM) Representational State Transfer (REST) API Specification Software v2.3.2* | |

| Doc ID | Title | Location |
|---|---|---|
| 337206 | *Intel® Rack Scale Design (Intel® RSD) Generic Assets Management Interface (GAMI) API Software v2.3.2* | |
| 337206 | *Intel® Rack Scale Design (Intel® RSD) Generic Assets Management Interface (GAMI) API Software v2.3.2* | |
| 337207 | *Intel® Rack Scale Design (Intel® RSD) Pooled System Management Engine (PSME) Representational State Transfer (REST) API Specification Software v2.3.2* | |
| DSP0263 | *Cloud Infrastructure Management Interface (CIMI) specification* | *https://www.dmtf.org/sites/default/files/standards/documents/DSP0263_1.0.1.pdf* |
| DSP0266 | *Redfish* Scalable Platforms Management API Specification v1.1.0* | *https://dmtf.org/sites/default/files/standards/documents/DSP0266_1.1.0.pdf* |
| DSP8010 | *Redfish* API Schema Read Me v2016.3* | https://www.dmtf.org/sites/default/files/standards/documents/DSP8010_2016.3.zip |
| RFC2119 | *Key Words for Use in RFCs to Indicate Requirement Levels, March 1997* | *https://ietf.org/rfc/rfc2119.txt* |
| RFC2616 | *Hypertext Transfer Protocol - HTTP/1.1* | https://tools.ietf.org/html/rfc2616 |
| RFC5789 | *PATCH Method for HTTP* | *https://tools.ietf.org/html/rfc5789* |
| N/A | *Swordfish* Scalable Storage Management API Specification v1.0.4* | *https://www.snia.org/sites/default/files/SMI/swordfish/v104/Swordfish_v1.0.4_Specification.pdf* |
| N/A | *Hypertext Transfer Protocol - HTTP/1.1* | *https://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.1.0.pdf* |
| N/A | *Scalable Platforms Management Forum (SPMF)* | *https://www.dmtf.org/standards/spmf* |

§

# 2.0 Pod Manager (PODM) API

## 2.1 PODM API Structure and Relations

The PODM Representational State Transfer (REST) Application Program Interface (API) provides the REST-based interface that allows full management of the Intel® RSD POD, including asset discovery, configuration, and composed node assembly.

### 2.1.1 PODM API Physical Resource Hierarchy

**Figure 1. PODM REST API Hierarchy**

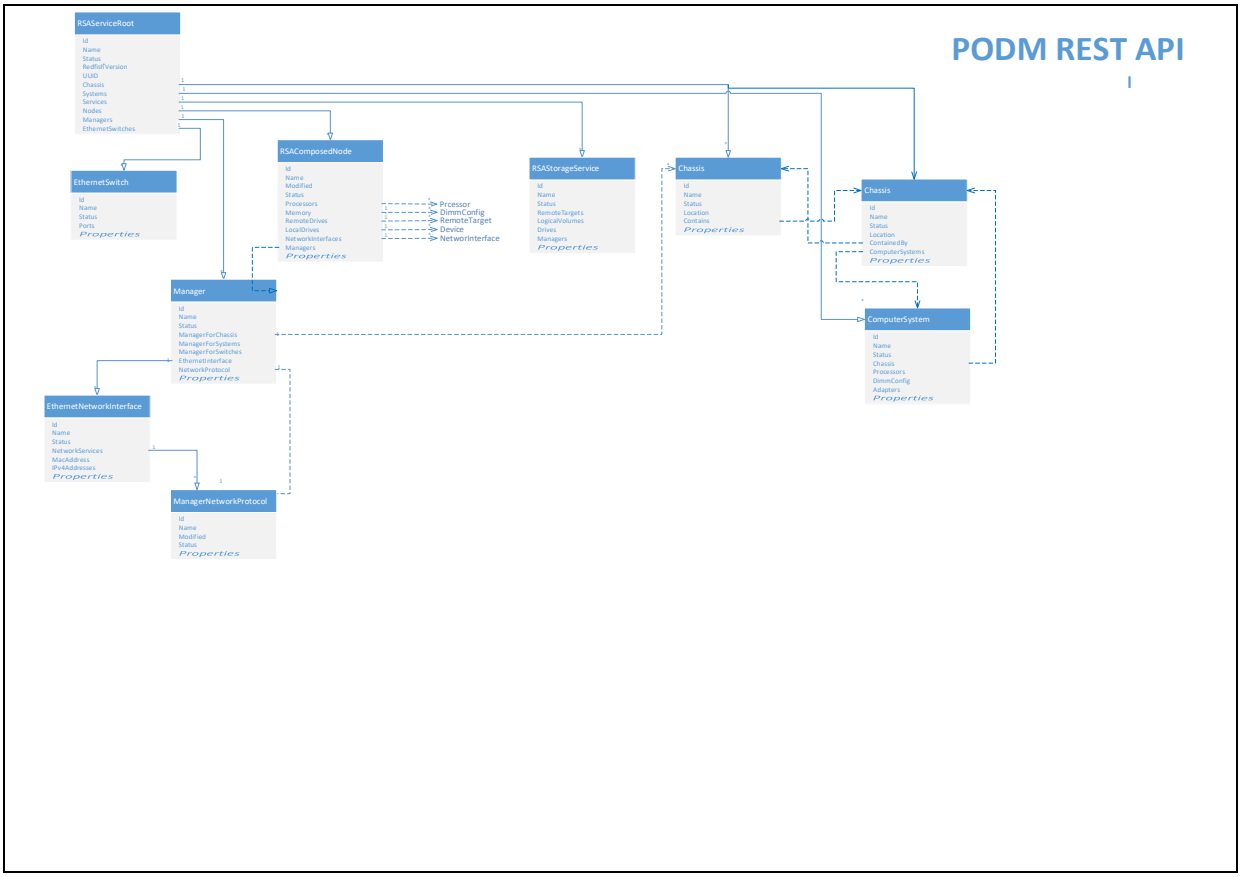



**Table 3. Resources and URI**

| Resource | URI |
|---|---|
| Service Root | /redfish/v1 |
| Chassis Collection | /redfish/v1/Chassis |
| Chassis | /redfish/v1/Chassis/{chassisID} |
| Computer Systems Collection | /redfish/v1/Systems |
| Computer Systems | /redfish/v1/Systems/{systemID} |
| Processors Collection | /redfish/v1/Systems/{systemID}/Processors |
| Processors | /redfish/v1/Systems/{systemID}/Processors/{processorID} |

| Resource | URI |
|---|---|
| Memory Collection | /redfish/v1/Systems/{systemID}/Memory |
| Memory | /redfish/v1/Systems/{systemID}/Memory/{memoryID} |
| Managers Collection | /redfish/v1/Managers |
| Managers | /redfish/v1/Managers/{managerID} |
| NetworkProtocol | /redfish/v1/Managers/{managerID}/NetworkProtocol |
| Network Interface Collection | /redfish/v1/Systems/{systemID}/EthernetInterfaces<br>/redfish/v1/Managers/{managerID}/EthernetInterfaces |
| Network Interface | /redfish/v1/Systems/{systemID}/EthernetInterfaces/{nicID}<br>/redfish/v1/Managers/{managerID}/EthernetInterfaces/{nicID} |
| EthernetSwitches Collection | /redfish/v1/EthernetSwitches |
| EthernetSwitches | /redfish/v1/EthernetSwitches/{switchID} |
| EthernetSwitches Ports Collection | /redfish/v1/EthernetSwitches/{switchID}/Ports |
| EthernetSwitchea Ports | /redfish/v1/EthernetSwitches/{switchID}/Ports/{portID} |
| VLAN Network Interface Collection | /redfish/v1/EthernetSwitches/{switchID}/Ports/{portID}/VLANs<br>/redfish/v1/Systems/{systemID}/EthernetInterfaces/{nicID}/VLANs<br>/redfish/v1/Managers/{managerID}/EthernetInterfaces/{nicID}/VLANs |
| VLAN Network Interface | /redfish/v1/EthernetSwitches/{switchID}/Ports/{portID}/VLANs/{vlanID}<br>/redfish/v1/Systems/{systemID}/EthernetInterfaces/{nicID}/VLANs/{vlanID}<br>/redfish/v1/Managers/{managerID}/EthernetInterfaces/{nicID}/VLANs/{vlanID} |
| Composed Nodes Collection | /redfish/v1/Nodes |
| Composed Nodes | /redfish/v1/Nodes/{nodeID} |
| SimpleStorage Collection | /redfish/v1/Systems/{system1}/SimpleStorage |
| SimpleStorage | /redfish/v1/Systems/{system1}/SimpleStorage/{storageID} |
| PowerZones Collection | /redfish/v1/Chassis/{chassisID}/PowerZones |
| PowerZones | /redfish/v1/Chassis/{chassisID}/PowerZones/{powerzoneID} |
| ThermalZone Collection | /redfish/v1/Chassis/{chassisID}/ThermalZones |
| ThermalZone | /redfish/v1/Chassis/{chassisID}/ThermalZones/{thermalzoneID} |
| Power | /redfish/v1/Chassis/{chassisID}/Power |
| Thermal | /redfish/v1/Chassis/{chassisID}/Thermal |
| Storage Subsystem Collection | /redfish/v1/Systems/{systemID}/Storage |
| Storage Subsystem | /redfish/v1/Systems/{systemID}/Storage/{storageID} |
| Drives | /redfish/v1/Chassis/{chassisID}/Drives/{driveID} |
| Fabrics collection | /redfish/v1/Fabrics |
| Fabrics | /redfish/v1/Fabrics/{fabricID} |
| Fabrics Switches collection | /redfish/v1/Fabrics/{fabricID}/Switches |
| Fabrics Switches | /redfish/v1/Fabrics/{fabricID}/Switches/{switchID} |
| Fabrics Switches Ports collection | /redfish/v1/Fabrics/{fabricID}/Switches/{switchID}/Ports |
| Fabrics Switches Ports | /redfish/v1/Fabrics/{fabricID}/Switches/{switchID}/Ports/{portID} |
| Fabrics Zones collection | /redfish/v1/Fabrics/{fabricID}/Zones |
| Fabrics Zones | /redfish/v1/Fabrics/{fabricID}/Zones/{zoneID} |
| Endpoints Collection | /redfish/v1/Fabrics/{fabricID}/Endpoints |
| Endpoints | /redfish/v1/Fabrics/{fabricID}/Endpoints/{endpointID} |
| PCIeDevices | /redfish/v1/Chassis/{chassisID}/PCIeDevices/{deviceID} |
| PCIeDevices Function | /redfish/v1/Chassis/{chassisID}/PCIeDevices/{deviceID}/Functions/{functionID} |

| Resource | URI |
|---|---|
| StorageServices Collection | /redfish/v1/StorageServices/ |
| StorageService | /redfish/v1/StorageServices/{serviceId} |
| Volumes Collection | /redfish/v1/StorageServices/Volumes |
| Volumes | /redfish/v1/StorageServices/{serviceId}/Volumes/{volumeId} |
| StoragePools Collection | /redfish/v1/StorageServices/{storageId}/StoragePools |
| StoragePools | [/redfish/v1/StorageServices/{serviceId}/{storageId}/StoragePools/{poolId}] |

§

# 3.0 PODM REST API Error Codes

This chapter describes all error codes that may be returned by the REST calls implemented in the PODM REST API of the Intel® RSD Software v2.3.2.

## 3.1 API Error Response

In the case of an error, PODM REST API responds with an Hypertext Transfer Protocol (HTTP) status code, as defined by the *Hypertext Transfer Protocol - HTTP/1.1*, RFC 2616, and constrained by additional requirements defined in this specification (refer to Table 2).

HTTP response status codes alone often don't provide enough information to determine the error cause. The PODM REST API returns extended error information as a JSON* object with a single property named `"error"`. The value of this property is the JSON object with the properties listed in Table 4.

**Table 4. API Error Response Attributes**

| Attribute | Description |
|---|---|
| `MessageId` | String indicating a specific error or message (not to be confused with the HTTP status code). This code can be used to access a detailed message from a message registry. |
| `Message` | A human readable error message indicating the semantics associated with the error. This is the complete message, and it doesn't rely on substitution variables. |
| `MessageArgs` | An optional array of strings representing the substitution parameter values for the message. This is included in the response if a `MessageId` is specified for a parameterized message. |
| `Severity` | An optional string representing the severity of the error. |
| `Resolution` | An optional string describing recommended action(s) to take to resolve the error. |
| `RelatedProperties` | An optional array of JSON Pointers defining the specific properties within a JSON payload described by the message. |

### 3.1.1 Example Error JSON* Object

```
{
    "error": {
        "code": "Base.1.0.GeneralError",
        "message": "A general error has occurred. See ExtendedInfo for more
information.",
        "@Message.ExtendedInfo": [
            {
                "@odata.type" : "/redfish/v1/$metadata#Message.v1_0_0.Message",
                "MessageId": "Base.1.0. MalformedJSON",
                "Message": "The request body submitted was malformed JSON and could
not be parsed by the receiving service",
                "Severity": "Error"
            }
            {
                "@odata.type" : "/redfish/v1/$metadata#Message.v1_0_0.Message",
                "MessageId": "Base.1.0.PropertyNotWriteable",
                "RelatedProperties": [
                    "#/Name"
                ],
                "Message": "The property Name is a read property and cannot be
assigned a value",
                "MessageArgs": [
                    "Name"
                ],
                "Severity": "Warning",
```

```
                "Resolution": "Remove the property from the request body and resubmit
the request if the operation failed"
            }
        ]
    }
}
```

## 3.2 API Error Codes

In general, if an error isn't described in any of the following tables, it is to be mapped into HTTP 500 Internal Server Error code.

### 3.2.1 General Error Codes

For detailed list of error codes, refer to the *Redfish* Scalable Platforms Management API Specification*, Section 6.5.2 (see Table 2).

Client should be prepared to handle the following error codes:

**Table 5. HTTP Error Status Codes**

| HTTP Status Code | Description |
|---|---|
| 400 Bad Request | The request could not be processed because it contains missing or invalid information (such as validation error on an input field, a missing required value, and so on). An extended error is returned in the response body. |
| 404 Not Found | The request specified a Uniform resource identifier (URI) of a resource that does not exist. |
| 405 Method Not Allowed | The HTTP verb specified in the request (for example, DELETE, GET, HEAD, POST, PUT, and PATCH) is not supported for this request URI. The response includes an "Allow" Symbol and note conventions are similar to typographical conventions used in the Cloud Infrastructure Management Interface (CIMI) specification. Notation used in JSON* serialization description: header, which provides a list of methods that are supported by the resource identified by the Request-URI. |
| 409 Conflict | A creation or update request cannot be applied given the state of the resource. |
| 500 Internal Server Error | The server encountered an unexpected condition that prevented it from fulfilling the request. An extended error is returned in the response body. |
| 501 Not Implemented | The server does not (currently) support the functionality required to fulfill the request. This is the appropriate response when the server does not recognize the request method and is not capable of supporting it for any resource. |
| 503 Service Unavailable | The server is currently unable to handle the request due to temporary overloading or maintenance of the server. |

### 3.2.2 PATCH Method Error Codes

For the PATCH method, the Intel® Rack Scale Design service conforms to *PATCH Method for HTTP* (see Table 2).

Service responds with following error codes in cases listed below:

- 400 Bad Request – malformed JavaScript* Object Notation in request (values not in range, unknown property, etc.)
- 405 Method Not Allowed – resource doesn't support PATCH method
- 409 Conflict – update cannot be executed at this moment. User might be able to resolve the conflict and resubmit the request.
- 501 Not Implemented – resource supports PATCH method, but current implementation doesn't (for instance, underlying hardware doesn't support such functionality)

- 500 Internal Server Error – all other situations where any of the above codes does not fit (for instance, underlying hardware does not allow execution of this particular request).

### 3.2.3 POST Method Error Codes

POST method is used to create new resources (POST request is submitted to the resource collection in which the new resource is to belong) or to initiate operation on the object (sending POST method to the URI of the action).

Service responds with following error codes in cases listed below:

- **400 Bad Request** – malformed JSON in request (values not in range, unknown property, etc.).
- **405 Method Not Allowed** – resource does not support POST method.
- **409 Conflict** – update cannot be executed at this moment. User might be able to resolve the conflict and resubmit the request.
- **501 Not Implemented** – resource supports POST method, but current implementation doesn't (for instance, underlying hardware doesn't support such functionality)
- **500 Internal Server Error** – all other situations where any of the above codes does not fit (for instance, underlying hardware does not allow execution of this particular request). Extended error information should provide information on why operation failed.

§

# 4.0 PODM REST API Definition

## 4.1 Odata* Support

Intel® RSD supports Odata* v4.0 as it is defined in *Redfish* Schema v2016.3* (see Table 2).

All resources within this REST API are identified by unique identifier property named "@odata.id". Resource Identifiers are represented in JSON* payloads as Uniform Resource Identifier (URI) paths relative to the Redfish Schema portion of the URI. That is, they always start with "/redfish/". The resource identifier is the canonical URL for the resource and can be used to retrieve or edit the resource, as appropriate.

## 4.2 Protocol Version

The protocol version is separate from the version of the resources or the version of the Redfish Schema supported by them.

Each version of the Redfish protocol is strongly typed. This is accomplished using the URI of the Redfish service in combination with the resource obtained at that URI, called the ServiceRoot.

The root URI for this version of the Redfish protocol is "/redfish/v1/".

While the major version of the protocol is represented in the URI, the major, minor, and errata version of the protocol are represented in the version property of the ServiceRoot resource, as defined in *Redfish* Schema v2016.3* (see Table 2) for that resource. The protocol version is a string of the form:

```
MajorVersion.MinorVersion.Errata
```

**Where:**

- **MajorVersion** = integer: something in the class changed in a backward incompatible way.
- **MinorVersion** = integer: a minor update. New functionality may have been added, but nothing removed. Compatibility is preserved with previous minor versions.
- **Errata** = integer: something in the prior version was broken and needed to be fixed.

Any resource discovered through links found by accessing the root service or any service or resource referenced using references from the root service conforms to the same version of the protocol supported by the root service.

### 4.2.1 Operations

#### 4.2.1.1 GET

**Request:**

```
GET /redfish
Content-Type: application/json
```

**Response:**

```
{
        "v1": "/redfish/v1/"
}
```

## 4.3 Odata Service Document

This service document provides a standard format for enumerating the resources exposed by the service, enabling generic hypermedia-driven Odata* clients to navigate to the resources of the service.

### 4.3.1 Operations

#### 4.3.1.1 GET

**Request:**

```
GET /redfish/v1/odata
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata",
    "value": [
        {
            "name": "Service",
            "kind": "Singleton",
            "url": "/redfish/v1/"
        },
        {
            "name": "Systems",
            "kind": "Singleton",
            "url": "/redfish/v1/Systems"
        },
        {
            "name": "Chassis",
            "kind": "Singleton",
            "url": "/redfish/v1/Chassis"
        },
        {
            "name": "Managers",
            "kind": "Singleton",
            "url": "/redfish/v1/Managers"
        },
        {
            "name": "Nodes",
            "kind": "Singleton",
            "url": "/redfish/v1/Nodes"
        },
        {
            "name": "StorageServices",
            "kind": "Singleton",
            "url": "/redfish/v1/StorageServices"
        },
        {
            "name": "EthernetSwitches",
            "kind": "Singleton",
            "url": "/redfish/v1/EthernetSwitches"
        },
        {
            "name": "Fabrics",
            "kind": "Singleton",
            "url": "/redfish/v1/Fabrics"
        }
    ]
```

```
}
```

## 4.4 Intel® RSD PODM Service Root

Intel® RSD PODM Service Root resource – entry point.

Property details available in `ServiceRoot.xml` metadata file.

### 4.4.1 Operations

#### 4.4.1.1 GET

**Request:**

```
GET /redfish/v1
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#ServiceRoot.ServiceRoot",
    "@odata.id": "/redfish/v1/",
    "@odata.type": "#ServiceRoot.v1_1_1.ServiceRoot",
    "Id": "RootService",
    "Name": "Root Service",
    "Description": "description-as-string",
    "RedfishVersion": "1.1.0",
    "UUID": "92384634-2938-2342-8820-489239905423",
    "Systems": {
        "@odata.id": "/redfish/v1/Systems"
    },
    "Chassis": {
        "@odata.id": "/redfish/v1/Chassis"
    },
    "Managers": {
        "@odata.id": "/redfish/v1/Managers"
    },
    "EventService": {
        "@odata.id": "/redfish/v1/EventService"
    },
    "Fabrics": {
      "@odata.id": "/redfish/v1/Fabrics"
    },
    "StorageServices": {
      "@odata.id": "/redfish/v1/StorageSerices"
    },
    "Oem": {
        "Intel_RackScale": {
            "@odata.type": "#Intel.Oem.ServiceRoot",
            "ApiVersion": "2.3.0",
            "Nodes": {
                "@odata.id": "/redfish/v1/Nodes"
            },
            "EthernetSwitches": {
                "@odata.id": "/redfish/v1/EthernetSwitches"
            }
        }
    },
   "Links": {}
}
```

#### 4.4.1.2 PUT

Operation not allowed on this resource.

#### 4.4.1.3 PATCH

Operation not allowed on this resource.

#### 4.4.1.4 POST

Operation not allowed on this resource.

#### 4.4.1.5 DELETE

Operation not allowed on this resource.

## 4.5 Chassis collection

**Table 6 Chassis Collection Attributes**

| Name | Chassis | | |
|---|---|---|---|
| Type URI | /redfish/v1/Chassis | | |
| Attribute | Type | Required | Description |
| Name | String | Yes | Name of collection |
| Members@odata.count | Number | No | Collection members count |
| Members | Array | No | Contains the members of this collection. |

### 4.5.1 Operations

#### 4.5.1.1 GET

**Request:**

```
GET /redfish/v1/Chassis
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Chassis",
    "@odata.id": "/redfish/v1/Chassis",
    "@odata.type": "#ChassisCollection.ChassisCollection",
    "Name": "Chassis Collection",
    "Members@odata.count": 6,
    "Members": [
        {
            "@odata.id": "/redfish/v1/Chassis/Pod"
        },
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1"
        },
        {
            "@odata.id": "/redfish/v1/Chassis/Drawer1"
        },
        {
            "@odata.id": "/redfish/v1/Chassis/FabricModule1"
        },
```

```
        {
            "@odata.id": "/redfish/v1/Chassis/Sled1"
        },
        {
            "@odata.id": "/redfish/v1/Chassis/Blade1"
        }
    ]
}
```

#### 4.5.1.2 PUT

Operation not allowed on this resource.

#### 4.5.1.3 PATCH

Operation not allowed on this resource.

#### 4.5.1.4 POST

Operation not allowed on this resource.

#### 4.5.1.5 DELETE

Operation not allowed on this resource.

## 4.6 Chassis

This is the Schema Definition for the Chassis resource, which represents properties for physical components for any system. This one object is intended to represent Racks, Rackmount Servers, Blades, Standalone, Modular Systems, Enclosures, and all other containers. The non-C/device centric parts of the schema are all accessed either directly or indirectly through this resource.

Details of this resource are described in metadata file: `Chassis.xml`. Original Equipment Manufacturer (OEM) extensions details are available in `IntelRackScaleOem.xml`.

### 4.6.1 Operations

#### 4.6.1.1 GET

**Request:**

```
GET /redfish/v1/Chassis/1
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Chassis/Members/$entity",
    "@odata.id": "/redfish/v1/Chassis/Rack1",
    "@odata.type": "#Chassis.v1_3_0.Chassis",
    "Id": "Rack1",
    "ChassisType": "RackMount",
    "Name": "name-as-string",
    "Description": "description-as-string",
    "Manufacturer": "Intel Corporation",
    "Model": "model-as-string",
    "SKU": "sku-as-string",
```

```
    "SerialNumber": "serial-number-as-string",
    "PartNumber": "part-number-as-string",
    "AssetTag": null,
    "IndicatorLED": "Unknown",
    "Status": {
        "State": "Enabled",
        "Health": "OK"
    },
    "Oem": {
        "Intel_RackScale": {
            "@odata.type": "#Intel.Oem.RackChassis",
            "Location": {
                "Id": "Rack1",
                "ParentId": "Pod1"
            },
            "RMMPresent": true,
            "RackSupportsDisaggregatedPowerCooling": true,
            "UUID": "Unique ID",
            "GeoTag": "54.348103, 18.645172"
        }
    },
    "Thermal": {
      "@odata.id": "/redfish/v1/Chassis/Rack1/Thermal"
    },
    "Power": {
      "@odata.id": "/redfish/v1/Chassis/Rack1/Power"
    },
    "Links": {
        "@odata.type": "#Chassis.v1_2_0.Links",
        "Contains": [
            {
                "@odata.id": "/redfish/v1/Chassis/Drawer1"
            }
        ],
        "ContainedBy": null,
        "ComputerSystems": [],
        "ManagedBy": [
            {
                "@odata.id": "/redfish/v1/Managers/RMM"
            }
        ],
        "ManagersInChassis": [
            {
                "@odata.id": "/redfish/v1/Managers/RMM"
            }
        ],
        "Oem": {
           "Intel_RackScale": {
            "@odata.type": "#Intel.Oem.ChassisLinks",
             "Switches": []
           }
        }
    }
}
```


#### 4.6.1.2 PUT

Operation not allowed on this resource.

#### 4.6.1.3 PATCH

Following properties can be updated by using the PATCH operation:

| Attribute | Type | Required | Description |
|---|---|---|---|
| `AssetTag` | String | No | The user assigned asset tag for this chassis. |
| `Oem->Intel_RackScale->Location->Id` | String | No | The user assigned location id for this chassis. It can be changed only for Rack Chassis. Support for changing this attribute is not mandatory in this API Specification v2.3.2, but if implemented it needs to be aligned to this specification. |

**Request:**

```
PATCH /redfish/v1/Chassis/Rack1
Content-Type: application/json
{
    "AssetTag": "Rack#1",
    "Oem": {
        "Intel_RackScale": {
            "Location": {
                "Id": "1234",
            },
        }
    },
}
```

**Response:**

```
HTTP/1.1 204 No Content
```

**Or**

```
HTTP/1.1 200 OK
```

With full resource representation.

#### 4.6.1.4 POST

Operation not allowed on this resource.

#### 4.6.1.5 DELETE

Operation not allowed on this resource.

## 4.7 StorageServiceCollection

Intel® Rack Scale Design `StorageServiceCollection` resource – provides a collection of available storage services. Refer to Table 7 for a list of attributes.

**Table 7. Storage Service Collection Attributes**

| Name | Storage Service | | |
|---|---|---|---|
| Type URI | /redfish/v1/StorageServices | | |
| Attribute | Type | Mandatory | Description |
| `Name` | String | Yes | Name of service collection. |
| `Members@odata.count` | Number | No | Collection members count |
| `Members` | Array | No | Contains the members of this collection. |

### 4.7.1 Operations

#### 4.7.1.1 GET

**Request:**

```
GET /redfish/v1/Services
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Services",
    "@odata.id": "/redfish/v1/Services",
    "@odata.type": "#StorageServiceCollection.StorageServiceCollection",
    "Name": "Storage Services Collection",
    "Description": "Collection of Storage Services",
    "Members@odata.count": 1,
    "Members": [
        {
            "@odata.id": "/redfish/v1/Services/RSS1"
        }
    ]
}
```

#### 4.7.1.2 PUT

Operation not allowed on this resource.

#### 4.7.1.3 PATCH

Operation not allowed on this resource.

#### 4.7.1.4 POST

Operation not allowed on this resource.

#### 4.7.1.5 DELETE

Operation not allowed on this resource.

## 4.8 ComposedNodeCollection

Intel® RSD `ComposedNodeCollection` resource – provides collection of all logical nodes. Refer to Table 8 for a list of attributes.

**Table 8. ComposedNodeCollection Attributes**

| **Name** | **`ComposedNodeCollection`** | | |
|---|---|---|---|
| **Type URI** | **/redfish/v1/Nodes** | | |
| **Attribute** | **Type** | **Mandatory** | **Description** |
| `Name` | String | Yes | Name of collection |
| `Members@odata.count` | Number | No | Collection members count |

| Attribute | Type | Mandatory | Description |
|---|---|---|---|
| Members | Array | No | Contains the members of this collection. |
| Actions | Object | No | Actions available:<br>**Allocate** – this action is first mandatory step to create composed node. In response to this action, proper resources are found and allocated for node composition. Node resource is created and URL (link) of this node is returned.<br>To allocate a Composed Node using PODM REST API it is necessary to create a JSON template describing requested resources.<br>The JSON template contains various details concerning resources to be used in Composed Node. All JSON template elements are optional, but should not be mutually exclusive. It is possible to supply the PODM with a JSON template containing no specific requirements (for example, {} – a pair of empty curly braces in HTTP request body) thus allowing PODM to propose a Composed Node containing resources chosen arbitrarily by PODM.<br>Format of JSON template (action payload) is described in Table 11<br>For more information about node allocation and assembly, refer to PODM_Allocation_Guide document. |

**Table 9. ComposedNodeCollection Action Attributes**

| Attribute | Type | Mandatory | Description |
|---|---|---|---|
| Name | String | No | Name of composed node.<br>***Note:*** Because ComposedNode is a Redfish* resource - its Name field is mandatory, so an attempt to directly set a **null** value results in an expected error. PODM sets a default name for newly created ComposedNode resource only upon not supplying the **Name** attribute. |
| Description | String | No | Description of ComposedNode |
| Processors | Array | No | Array of requirements for processor for ComposedNode. Each processor requirement may contain one or more optional attributes: |

| Attribute | Type | Description |
|---|---|---|
| Model | String | Processor model that should be used for ComposedNode (exact model name) |
| TotalCores | Number | Minimum number of processor cores |
| AchievableSpeedMHz | Number | Minimum achievable processor operating frequency |
| InstructionSet | String | Processor supported instruction set.<br>"x86" – x86 32-bit<br>"x86-64" – x86 64-bit<br>"IA-64" – Intel IA-64<br>"ARM-A32" – ARM 32-bit<br>"ARM-A64" – ARM 64-bit<br>"MIPS32" – MIPS 32-bit<br>"MIPS64" – MIPS 64-bit<br>"OEM" – OEM-defined |

| Attribute | Type | Mandatory | Description | | |
|---|---|---|---|---|---|
| | | | `Resource` | Object | Reference to particular processor that should be used in `ComposedNode` |
| | | | `Chassis` | Object | Link to chassis object within this processor should be contained. |
| | | | `Brand` | String | Brand of CPU that should be used to allocate node. Allowable values:<br>New Intel® Xeon® branding: `"Silver", "Gold", "Platinum"`<br>Intel® Xeon® family: E3, E5, E7 SoC/Intel Atom® family: X3 (Avoton), X5 (Broadwell-DE), X7<br>Core family: I3, I5, I7<br>`"Unknown"` – processor doesn't fit to any above categories |
| | | | `Capabilities` | Array | Array of strings describing processor capabilities (like reported in `/proc/cpuinfo flags`), such as:<br>`"sse"` - Streaming SIMD Extensions<br>`"avx"` - Advanced Vector Extensions |
| | | | `ProcessorType` | String | This property contains the string which identifies the type of processor:<br>`"CPU"`<br>`"FPGA"`<br>`"GPU"`<br>`"DSP"`<br>`"Accelerator"`<br>`"OEM"` |

<table>
<tr><th>Attribute</th><th>Type</th><th>Mandatory</th><th>Description</th></tr>
<tr><td>Memory</td><td>Array</td><td>No</td><td>Array of requirements for memory for ComposedNode.
<table>
<tr><th>Attribute</th><th>Type</th><th>Description</th></tr>
<tr><td>CapacityMiB</td><td>Number</td><td>Minimum memory capacity requested for ComposedNode</td></tr>
<tr><td>MemoryDeviceType</td><td>String</td><td>Type details of DIMM:<br>"DDR"<br>"DDR2"<br>"DDR3"<br>"DDR4"<br>"DDR4 SDRAM"<br>"DDR4E SDRAM"<br>"LPDDR4SDRAM"<br>"DDR3 SDRAM"<br>"LPDDR3 SDRAM"<br>"DDR2 SDRAM"<br>"DDR2 SDRAM-FB-DIMM"<br>"DDR2SDRAM-FB-DIMM PROBE"<br>"DDR SGRAM"<br>"DDR SDRAM"<br>"ROM"<br>"SDRAM"<br>"EDO"<br>"FastPageMode"<br>"PipelinedNibble"</td></tr>
<tr><td>SpeedMHz</td><td>Number</td><td>Minimum supported memory speed.</td></tr>
<tr><td>Manufacturer</td><td>String</td><td>Requested memory manufacturer.</td></tr>
<tr><td>DataWidthBits</td><td>Number</td><td>Requested memory data width in bits.</td></tr>
<tr><td>Resource</td><td>Object</td><td>Reference to particular memory module that should be used in ComposedNode</td></tr>
<tr><td>Chassis</td><td>Object</td><td>Link to chassis object within this memory DIMM should be contained.</td></tr>
</table></td></tr>
<tr><td>RemoteDrives</td><td>Array</td><td>No</td><td>Array of requirements for remote drives that should be created/connected to ComposedNode
<table>
<tr><th>Attribute</th><th>Type</th><th>Description</th></tr>
<tr><td>CapacityGiB</td><td>Number</td><td>This property is mutually exclusive with Resource - only one can be provided at time. Minimum drive capacity requested for ComposedNode</td></tr>
<tr><td>Protocol</td><td>String</td><td>Defines which protocol should be use to communicate with attached drive. Currently supported:<br>"iSCSI",<br>"NVMeOverFabrics"</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Attribute</th><th>Type</th><th>Mandatory</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><td>Master</td><td>Object</td><td>Defines master volume that should be taken to create new remote drive. It contains properties described in Table 10.</td></tr>
<tr><td>Resource</td><td>Object</td><td>Reference to particular remote drive (represented by Volume or target Endpoint resource) that should be used for this ComposedNode.</td></tr>
</table>
</td></tr>
<tr><td>LocalDrives</td><td>Array</td><td>No</td><td>
Array of requirements for local drives for ComposedNode.
<table>
<tr><th>Attribute</th><th>Type</th><th>Description</th></tr>
<tr><td>CapacityGiB</td><td>Number</td><td>Minimum drive capacity requested for ComposedNode</td></tr>
<tr><td>Type</td><td>String</td><td>Drive type<br>“HDD”<br>“SSD”</td></tr>
<tr><td>MinRPM</td><td>Number</td><td>Minimum rotation speed of requested drive</td></tr>
<tr><td>SerialNumber</td><td>String</td><td>Serial number of requested drive</td></tr>
<tr><td>Interface</td><td>String</td><td>Interface of requested drive:<br>“SAS”<br>“SATA”<br>“NVMe”</td></tr>
<tr><td>Resource</td><td>Object</td><td>Reference to particular local drive that should be used in ComposedNode</td></tr>
<tr><td>Chassis</td><td>Object</td><td>Link to chassis object within this drive should be contained.</td></tr>
<tr><td>FabricSwitch</td><td>Boolean</td><td>Determine if local drive should be connected using fabric switch or local connected</td></tr>
</table>
</td></tr>
<tr><td>EthernetInterfaces</td><td>Array</td><td>No</td><td>
Array of requirements for Ethernet interfaces of ComposedNode
<table>
<tr><th>Attribute</th><th>Type</th><th>Description</th></tr>
<tr><td>SpeedMbps</td><td>Number</td><td>Minimum speed of Ethernet interface requested for ComposedNode</td></tr>
<tr><td>VLANs</td><td>Array</td><td>Array of VLANs that should be configured on connected switch port for ComposedNode for given Ethernet interface in following format:<br>VLANId – number indicating VLAN Id<br>Tagged – Boolean value describing if given VLAN is tagged.<br>Deprecated: This is going to be removed in future releases of Intel® RSD. Equivalent functionality will be provided via CRUD operation on Ethernet Switch port VLANs.</td></tr>
</table>
</td></tr>
</table>

| Attribute | Type | Mandatory | Description |
|---|---|---|---|
| | | | `PrimaryVLAN` — Number — Primary VLAN ID that should be set for given Ethernet Interface. **Deprecated**: This is going to be removed in future releases of Intel® RSD. Equivalent functionality will be provided via CRUD operation on Ethernet Switch port.<br>`Resource` — Object — Reference to particular Ethernet interface that should be used in `ComposedNode`.<br>`Chassis` — Object — Link to chassis object within this network interface should be contained. |
| `TotalSystemMemory` | Number | No | Minimum total amount of memory in `ComposedNode`. |
| `TotalSystemCoreCount` | Number | No | Minimum total processor core count in `ComposedNode`. |
| `Security` | Object | No | Security requirements for `ComposedNode`<br>`TpmPresent` — Boolean — This is used to specify if `ComposedNode` should have TPM module present.<br>`TpmInterfaceType` — String (enum) — This is the used to specify requested TPM interface type. It overrides TpmPresent attribute.<br>`TxtEnabled` — Boolean — This is used to specify if `ComposedNode` should have TXT mode enabled.<br>`ClearTPMOnDelete` — Boolean — This is used to specify if Trusted Platform Module) TPM module should be cleared on `ComposedNode` DELETE request. |

**Table 10. RemoteDrives Target Properties**

| Attribute | Type | Description |
|---|---|---|
| `Type` | String | Type of replication of master drive:<br>Clone – volume should be cloned<br>Snapshot – Copy on Write should be created from indicated volume (don't work for Protocol `NVMeOverFabrics`)<br><br>***Note:*** For iSCSI protocol, the newly created drive will be marked by PODM as Bootable |
| `Resource` | Object | Reference to volume that should be used as master for replication. |

## 4.8.1 Operation

### 4.8.1.1 GET

**Request:**

```
GET /redfish/v1/Nodes
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Nodes",
    "@odata.id": "/redfish/v1/Nodes",
    "@odata.type": "#ComposedNodeCollection.ComposedNodeCollection",
    "Name": "ComposedNodes Collection",
    "Members@odata.count": 1,
    "Members": [
        {
            "@odata.id": "/redfish/v1/Nodes/Node1"
        }
    ],
    "Actions": {
        "#ComposedNodesCollection.Allocate": {
            "target": "/redfish/v1/Nodes/Actions/Allocate"
        }
    }
}
```

#### 4.8.1.2 PUT

Operation not allowed on this resource.

#### 4.8.1.3 PATCH

Operation not allowed on this resource.

#### 4.8.1.4 POST

***Note:*** Currently, a user can request allocation of a single node with a single request. Node components—CPU, Memory, Local Storage, and Network Interface—must be located on a single physical Blade. Remote storage can be located anywhere in the POD.

***Note:*** Below JSON is just an example. For more details refer to Section 6.0, Composed Node Additional Info.

**Request:**

```
POST /redfish/v1/Nodes/Actions/Allocate
Content-Type: application/json
{
        "Name": "My first composed node",
        "Description": "Test node",
        "Processors": [{
                "Model": "Multi-Core Intel(R) Xeon(R) processor 7xxx Series",
                "TotalCores": 2,
                "AchievableSpeedMHz": 2000,
                "InstructionSet": "x86",
                "Oem": {
                        "Brand": "E5",
                        "Capabilities": [
                                "sse"
                        ],

                },
                "Resource": {
                        "@odata.id": "/redfish/v1/Systems/System1/Processors/CPU1"
                }
        }],
        "Memory": [{
                "CapacityMiB": 16000,
                "MemoryDeviceType": "DDR3",
```

```
                "SpeedMHz": 1600,
                "Manufacturer": "Intel",
                "DataWidthBits": 64,
                "Resource": {
                        "@odata.id": "/redfish/v1/Systems/System1/Memory/Dimm1"
                },
                "Chassis": {
                        "@odata.id": "/redfish/v1/Chassis/Rack1"
                }
        }],
        "RemoteDrives": [{
                "CapacityGiB": 80,
                "Protocol": "iSCSI",
                "Master": {
                        "Type": "Snapshot",
                        "Resource": {
                                "@odata.id":
"/redfish/v1/StorageServices/iSCSI/Volumes/1"
                        }
                }
        }],
        "LocalDrives": [{
                "CapacityGiB": 500,
                "Type": "HDD",
                "MinRPM": 5400,
                "SerialNumber": "12345678",
                "Interface": "SATA",
                "Resource": {
                        "@odata.id": "redfish/v1/Chassis/Blade1/Drives/Disk1"
                },
                "FabricSwitch": false
        }],
        "EthernetInterfaces": [{
                "SpeedMbps": 1000,
                "PrimaryVLAN": 100,
                "VLANs": [{
                        "VLANId": 100,
                        "Tagged": false
                }],
                "Resource": {
                        "@odata.id":
"/redfish/v1/Systems/System1/EthernetInterfaces/LAN1"
                }
        }],
        "Security": {
          "TpmPresent": true,
          "TpmInterfaceType": "TPM2_0",
          "TxtEnabled": false,
          "ClearTPMOnDelete": true
        },
        "Oem": {
        },
        "TotalMemoryCapacityMiB": 32000,
        "TotalProcessorCoreCount": 2
}
```

**Response:**

```
HTTP/1.1 201 Created
Location: http://<IP>:<Port>/redfish/v1/Nodes/2
```

#### 4.8.1.5 DELETE

Operation not allowed on this resource.

## 4.9 Composed Node

Composed Node resource – provides detailed information about an assembled logical node identified by `{nodeID}`. Refer to Table 11 for the lists of the attributes.

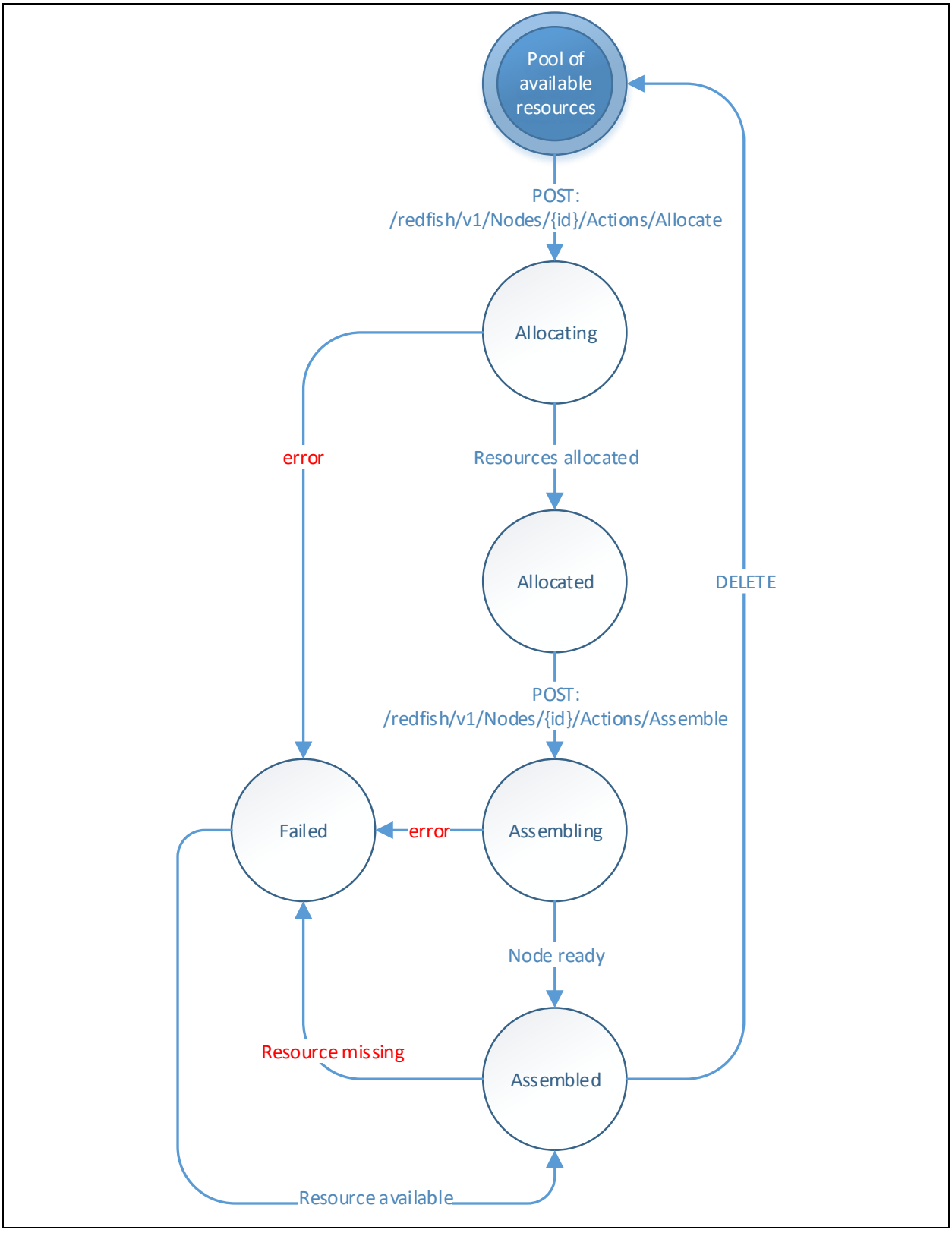
**Table 11 Composed node attributes**

| Name | Composed Node | | |
|---|---|---|---|
| Type URI | /redfish/v1/Nodes/{nodeID} | | |
| Attribute | Type | Mandatory | Description |
| `Id` | String | Yes | Provides a ID of this resource |
| `Name` | String | Yes | Name of composed Node |
| `Description` | String | No | User provided node description |
| `UUID` | String | No | UUID of computer system used as a base for this node. |
| `PowerState` | String (enum) | No | This is the current power state of the node:<br>• `"On"` - The system is powered on<br>• `"Off"` - The system is powered off, although some components may continue to have AUX power such as management controller<br>• `"PoweringOn"` - A temporary state between Off and On. This temporary state can be very short.<br>• `"PoweringOff"` - A temporary state between on and off. The power off action can take time while the OS is in the shutdown process. |
| `Status` | Object, null | No | Refer to Section 5.1 for status of resource. |
| `ComposedNodeState` | String (enum) | Yes | Current state of assembly process for this node.<br>• Allocating: Allocating resources for node is in progress. Next state can be Allocated or Failed.<br>• Allocated: Node resources has been allocated, but assembly not started yet. After `ComposedNode.Assemble` action state will progress to Assembling.<br>• Assembling: Assembly process initiated, but not finished yet. When assembly is done it will change into Assembled.<br>• Assembled: Node successfully assembled.<br>• Failed: Allocation or assembly process failed, or in runtime, one of composing components was removed or transitioned in error state. |

<table>
<tr><td>Name</td><td colspan="3">Composed Node</td></tr>
<tr><td>Type URI</td><td colspan="3">/redfish/v1/Nodes/{nodeID}</td></tr>
<tr><td>Attribute</td><td>Type</td><td>Mandatory</td><td>Description</td></tr>
<tr><td>Boot</td><td>Object</td><td>No</td><td>
<table>
<tr><td>Name</td><td>Type</td><td>Required</td><td>Description</td></tr>
<tr><td>BootSourceOverrideEnabled</td><td>String, Null</td><td>No</td><td>State of the Boot Source Override feature.<br>Proper values:<br>"Disabled"<br>"Once"<br>"Continuous"</td></tr>
<tr><td>BootSourceOverrideTarget</td><td>String, Null</td><td>No</td><td>The current boot source to be used at next boot instead of the normal boot device, if BootSourceOverrideEnabled is true.</td></tr>
<tr><td>BootSourceOverrideTarget@Redfish.AllowableValues</td><td>Array</td><td>No</td><td>Array of supported boot sources.</td></tr>
<tr><td>BootSourceOverrideMode</td><td>String, Null</td><td>No</td><td>The BIOS Boot Mode (either Legacy or UEFI) to be used when BootSourceOverrideTarget boot source is booted from.</td></tr>
<tr><td>BootSourceOverrideMode@Redfish.AllowableValues</td><td>Array</td><td>No</td><td>Array of supported boot modes.</td></tr>
</table>
</td></tr>
<tr><td>Oem</td><td>Object, Null</td><td>No</td><td>OEM defined object</td></tr>
<tr><td>ClearTPMOnDelete</td><td>Boolean</td><td>No</td><td>This is used to specify if TPM module should be cleared on composed node DELETE request. If not specified in allocation request, it is assumed that it is set to "true".</td></tr>
<tr><td>Links</td><td>Object</td><td>No</td><td>Link section:
<table>
<tr><td>Name</td><td>Type</td><td>Required</td><td>Description</td></tr>
<tr><td>ComputerSystem</td><td>Object, null</td><td>Yes</td><td>Reference to ComputerSystem resource used to compose this node.</td></tr>
<tr><td>Processors</td><td>Array</td><td>No</td><td>Array of references to Processor resources.</td></tr>
<tr><td>Memory</td><td>Array</td><td>No</td><td>Array of references to Memory resources.</td></tr>
<tr><td>Storage</td><td>Array</td><td>No</td><td>An array of references to the storage resources used by this composed node.</td></tr>
<tr><td>EthernetInterfaces</td><td>Array</td><td>No</td><td>Array of links to Ethernet Interface collection associated with this Composed Node.</td></tr>
<tr><td>ManagedBy</td><td>Array</td><td>No</td><td>An array of references to Managers responsible for this Composed Node.</td></tr>
</table>
</td></tr>
<tr><td>Actions</td><td>Object</td><td>Yes</td><td>Actions available for this node:<br>• Reset action with following values:</td></tr>
</table>

| Name | Composed Node | | |
|---|---|---|---|
| Type URI | /redfish/v1/Nodes/{nodeID} | | |
| Attribute | Type | Mandatory | Description |
| | | | – "On" - Turn the system on<br>– "ForceOff" - Turn the system off immediately (non-graceful) shutdown<br>– "GracefulRestart" - Perform a graceful system shutdown followed by a restart of the system<br>– "ForceRestart" - Perform an immediate (non-graceful) shutdown, followed by a restart of the system<br>– "Nmi - Generate" a Diagnostic Interrupt (usually an NMI on x86 systems) to cease normal operations, perform diagnostic actions and typically halt the system.<br>– "ForceOn" - Turn the system on immediately<br>– "PushPowerButton" - Simulate the pressing of the physical power button on this system<br>– "GracefulShutdown" – initiate a soft-shutdown of OS via ACPI<br>• Assemble: Does not consume any parameters. Second step of creating a composed node (after Allocate Action on Nodes collection). That action will assembly logical node – initiate ComposedNodeState change from Allocated state into Assembling state<br>• After final assembly, the composed node will stay in Off PowerState. To change its state, one needs to execute Reset action with "On" parameter (make sure that **Boot** is set properly).<br>• **AttachResource** – this action allows attaching poolable resource like remote drive, FPGA, NIC or existing endpoint etc. to this composed node. Action can be performed when Composed Node is Assembled or Failed. Requires below parameter:<br>– Resource – link to device from attachable device pool<br>– Protocol - defines protocol (fabric) used for connecting device if multiple are supported.<br>• **DetachResource** – action used to detach already connected device. In case of storage drive/volume, if EraseOnDetach parameter of drive is set to "**true**", this drive will be Secure Erased before returning to the pool. Action can be performed when Composed Node is Assembled or Failed. It takes one argument:<br>– Resource – link to already connected device or endpoint, which needs to be detached. |

**Figure 2. ComposedNodeState Changes during Assembly Process**



### 4.9.1 Operations

#### 4.9.1.1 GET

**Request:**

```
GET /redfish/v1/Nodes/{nodeID}
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Nodes/Members/$entity",
    "@odata.id": "/redfish/v1/Nodes/Node1",
    "@odata.type": "#ComposedNode.v1_0_0.ComposedNode",
    "Id": "Node1",
    "Name": "Composed Node",
    "Description": "Node #1",
    "UUID": "00000000-0000-0000-0000-000000000000 - the same as Computer System",
    "PowerState": "On",
    "Status": {
        "State": "Enabled",
        "Health": "OK",
        "HealthRollup": "OK"
    },
    "ComposedNodeState": "Allocated",
    "Boot": {
        "BootSourceOverrideEnabled": "Disabled",
        "BootSourceOverrideTarget": "None",
        "BootSourceOverrideTarget@Redfish.AllowableValues": [
            "None",
            "Pxe",
            "Hdd",
            "RemoteDrive"
        ] ,
    "BootSourceOverrideMode": "Legacy",
    "BootSourceOverrideMode@Redfish.AllowableValues": ["Legacy",”UEFI”]
    },
    "Oem": {},
    "ClearTPMOnDelete": true,
    "Links": {
        "ComputerSystem": {
            "@odata.id": "/redfish/v1/Systems/System1"
        },
        "Processors": [
            {
                "@odata.id": "/redfish/v1/Systems/System1/Processors/CPU1"
            }
        ],
        "Memory": [
            {
                "@odata.id": "/redfish/v1/Systems/System1/Memory/Dimm1"
            }
        ],
        "EthernetInterfaces": [
            {
                "@odata.id": "/redfish/v1/Systems/System1/EthernetInterfaces/LAN1"
            }
        ],
        "Storage": [
            {
                "@odata.id": "/redfish/v1/Chassis/Blade1/Drives/1"
```

```
            },
            {
                "@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/1"
            }
        ],
        "ManagedBy": [
            {
                "@odata.id": "/redfish/v1/Managers/PODM"
            }
        ],
        "Oem": {}
    },
    "Actions": {
        "#ComposedNode.Reset": {
            "target": "/redfish/v1/Nodes/Node1/Actions/ComposedNode.Reset",
            " @Redfish.ActionInfo":
"/redfish/v1/Nodes/Node1/Actions/ComposedNodeResetActionInfo"
        },
        "#ComposedNode.Assemble": {
            "target": "/redfish/v1/Nodes/Node1/Actions/ComposedNode.Assemble"
        },
    "#ComposedNode.AttachResource": {
      "target": "/redfish/v1/Nodes/Node1/Actions/ComposedNode.AttachResource",
      "@Redfish.ActionInfo":
"@odata.id":"/redfish/v1/Nodes/Node1/Actions/AttachResourceActionInfo"
    },
    "#ComposedNode.DetachResource": {
      "target": "/redfish/v1/Nodes/Node1/Actions/ComposedNode.DetachResource",
      "@Redfish.ActionInfo":
"@odata.id":"/redfish/v1/Nodes/Node1/Actions/DetachResourceActionInfo"
    }
  }
}
```

### 4.9.1.2 PUT

Operation not allowed on this resource.

### 4.9.1.3 PATCH

Table 12 lists the properties that can be updated by the PATCH operation.

**Table 12. Properties Updated by PATCH Operation**

| Attribute | Type | Required | Description |
|---|---|---|---|
| `Boot` | Object | No | Boot override properties, details in Table 13 |
| `ClearTPMOnDelete` | Boolean | No | This is used to specify if TPM module should be cleared on composed node `DELETE` request. |

Table 13 describes `"Boot"` properties that can be patched:

**Table 13 Boot Override Update Properties**

| Attribute | Type | Required | Description |
|---|---|---|---|
| `BootSourceOverrideEnabled` | String | No | Describes the state of the Boot Source Override feature. Allowed values:<br>`"Disabled"` - The system will boot as normal<br>`"Once"` - On its next boot cycle, the system will boot (one time) to the `BootSourceOverrideTarget` |

| Attribute | Type | Required | Description |
|---|---|---|---|
| | | | "Continuous" - The system will boot to the target specified in the BootSourceOverrideTarget until this property is set to **Disabled** |
| BootSourceOverrideTarget | String | No | The current boot source to be used at next boot instead of the normal boot device, if BootSourceOverrideEnabled is true. Supported values:<br>"None" - Boot from the normal boot device<br>"Pxe" - Boot from the Pre-Boot EXecution (PXE) environment<br>"Hdd" - Boot from a hard drive<br>"RemoteDrive" – Boot from remotely attached drive (iSCSI) |
| BootSourceOverrideMode | String | No | The BIOS Boot Mode (either Legacy or UEFI) to be used when BootSourceOverrideTarget boot source is booted from:<br>"Legacy" - The system will boot in non-UEFI boot mode to the Boot Source Override Target<br>"UEFI" - The system will boot in UEFI boot mode to the Boot Source Override Target |

***Note:*** Boot property is representing only override values, not current boot source configured on system. To make sure that correct boot source/mode will be applied, it is recommended to send PATCH to boot property after node assembly, before powering it on.

**Request:**

```
PATCH /redfish/v1/Nodes/Node1
Content-Type: application/json
{
        "Boot": {
                 "BootSourceOverrideEnabled": "Once",
                 "BootSourceOverrideTarget": "Pxe",
                 "BootSourceOverrideMode": "Legacy"
        }
}
```

**Response:**

```
HTTP/1.1 204 No Content
```

**Or:**

```
HTTP/1.1 200 OK
```

with full resource representation.

### 4.9.1.4 POST

#### 4.9.1.4.1 Reset Action

**Request:**

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.Reset
Content-Type: application/json
{
        "ResetType": "On"
}
```

**Response:**

```
HTTP/1.1 204 No Content
```

#### 4.9.1.4.2 Assemble action

**Request:**

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.Assemble
```

**Response:**

```
HTTP/1.1 204 No Content
```

#### 4.9.1.4.3 AttachResource action – specific device

Attaching specific device to existing `ComposedNode`.

**Request:**

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.AttachResource
Content-Type: application/json
{
    "Resource": {
        "@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/1"
    }
}
```

**Response:**

```
HTTP/1.1 204 No Content
```

#### 4.9.1.4.4 AttachResource action – specifing Protocol

Attaching the specific device to the existing `ComposedNode` with protocol specified.

**Request:**

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.AttachResource
Content-Type: application/json
{
    "Resource": {
        "@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/1"
    },
    "Protocol": "NVMeOverFabrics"
}
```

**Response:**

```
HTTP/1.1 204 No Content
```

#### 4.9.1.4.5 DetachResource Action

Action used to detach specified resource from `ComposedNode`. Resource attribute can point to the Drive, Volume, Endpoint etc.

This action will trigger following operations:

- If resource is Peripheral Component Interconnect Express (PCIe*) Drive connected via PCIe switch it is removed from `RootComplex` of `ComposedNode`
- If resource is Volume attached remotely (`iSCSI, NVMeOverFabrics`) associated endpoint is being deleted
- If resource were Drive attached via PCIe Switch or Volumes attached remotely (`iSCSI`, `NVMeOverFabrics`) with property `EraseOnDetach` set to ***true***, it is securely erased.

**Request:**

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.DetachResource
Content-Type: application/json
{
    "Resource": {
       "@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/1"
    }
}
```

**Response:**

```
HTTP/1.1 204 No Content
```

#### 4.9.1.5 DELETE

Upon deletion (disassembly) of `ComposedNode` several actions are performed:

1. Force off request is sent to Computer System
2. All VLANs (except for reserved ones – refer to Section 6.1.8.1) are removed from Ethernet switch ports associated with Computer System's Ethernet interfaces
3. All PCIe devices connected to Node via PCIe switch are detached
4. All Volumes attached remotely are disconnected
5. All Drives attached via PCIe Switch or Volumes attached remotely (iSCSI, NVMe* over Fabrics) with property `EraseOnDetach` set to ***true*** are securely erased
6. All Volumes that were erased are deleted
7. If a Computer System used by a *Composed Node* has any TPM module in the Enabled state and `ClearTPMOnDelete` property is set to **true**, the PODM will issue a `ChangeTPMState` action with the `ClearOwnership` parameter set to **true** to clear the TPM ownership

***Note:*** If the TPM is not present, or all of the TPM modules are in a Disabled state, the TPM won't be cleared.

`DELETE` action removes components reservation (deallocation) and put them back to the resource pool.

When any resource cannot be removed from composed node, `ComposedNodeState` will be changed to *Failed*. All remaining resources will return to proper pools. Resending `DELETE` operation will remove node, without performing the actions described above.

**Request:**

```
DELETE /redfish/v1/Nodes/1
```

**Response:**

```
HTTP/1.1 204 No Content
```

## 4.10 ActionInfo

`ActionInfo` describes the parameters and other information necessary to perform a Redfish Action to a particular Action target. As parameter support may differ between implementations, and even among instances of a resource, this data can be used to ensure Action requests from applications contain supported parameters.

## 4.10.1 Operations

### 4.10.1.1 GET (ComposedNode::AttachResource ActionInfo)

**Request:**

```
GET /redfish/v1/Nodes/Node1/Action/AttachResourceActionInfo
Content-Type: application/json
```

**Response:**

```
{
  "@odata.type": "#ActionInfo.v1_0_0.ActionInfo",
  "@odata.context": "/redfish/v1/$metadata#ActionInfo.ActionInfo",
  "@odata.id": "/redfish/v1/Nodes/Node1/Actions/AttachResourceActionInfo"
  "Id": "AttachResourceActionInfo"
  "Name": "Attach Resource ActionInfo"
  "Description": null

  "Parameters": [
    {
      "Name": "Resource",
      "Required": true,
      "DataType": "Object",
      "ObjectDataType": "#Resource.Resource",
      "AllowableValues": [
        {"@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/10"},
        {"@odata.id": "/redfish/v1/Fabrics/iSCSI/Endpoints/5"},
        {"@odata.id": "/redfish/v1/Fabrics/NVMe/Endpoints/8"}
      ]
    },
    {
      "Name": "Protocol",
      "Required": false,
      "DataType": "String",
      "AllowableValues": [ "iSCSI", "NVMeOverFabrics"]
    }
  ],
  "Oem": {},
}
```

### 4.10.1.2 GET (ComposedNode::DetachResource ActionInfo)

**Request:**

```
GET /redfish/v1/Nodes/Node1/Actions/DetachResourceActionInfo
Content-Type: application/json
```

**Response:**

```
{
  "@odata.type": "#ActionInfo.v1_0_0.ActionInfo",
  "@odata.context": "/redfish/v1/$metadata#ActionInfo.ActionInfo",
  "@odata.id": "/redfish/v1/Nodes/Node1/Actions/DetachResourceActionInfo"
  "Id": "DetachResourceActionInfo"
  "Name": "Detach Resource ActionInfo"
  "Description": null

  "Parameters": [
    {
      "Name": "Resource",
      "Required": true,
      "DataType": "Object",
```

```
      "ObjectDataType": "#Resource.Resource",
      "AllowableValues": [
        {"@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/1"}
      ]
    }
  ],
  "Oem": {},
}
```

### 4.10.1.3 GET (ComposedNode::Reset ActionInfo)

**Request:**

```
GET /redfish/v1/Nodes/Node1/Actions/ResetActionInfo
Content-Type: application/json
```

**Response:**

```
{
  "@odata.type": "#ActionInfo.v1_0_0.ActionInfo",
  "@odata.context": "/redfish/v1/$metadata#ActionInfo.ActionInfo",
  "@odata.id": "/redfish/v1/Nodes/Node1/Actions/ResetActionInfo"
  "Id": "ResetActionInfo"
  "Name": "Reset ActionInfo"
  "Description": null

  "Parameters": [
    {
      "Name": "Resource",
      "Required": true,
      "DataType": "String",
      "AllowableValues": [
                          "On",
                          "ForceOff",
                          "GracefulShutdown",
                          "GracefulRestart",
                          "ForceRestart",
                          "Nmi",
                          "ForceOn",
                          "PushPowerButton"
                  ]
    }
  ],
  "Oem": {},
}
```

### 4.10.1.4 PUT

Operation not allowed on this resource.

### 4.10.1.5 PATCH

Operation not allowed on this resource.

### 4.10.1.6 POST

Operation not allowed on this resource.

#### 4.10.1.7 DELETE

Operation not allowed on this resource.

## 4.11 PSME and Storage Services Resources

PODM supports PSME and Storage Services resources. Table 14 describes which resources and their operations are included as a part of PODM API. For additional details, refer to *Intel® Rack Scale Design (Intel® RSD) Pooled System Management Engine (PSME) Representational State Transfer (REST) API Specification Software v2.3.2* and *Intel® Rack Scale Design (Intel® RSD) Pooled System Management Engine (PSME) Representational State Transfer (REST) API Specification Software v2.3.2*, and *Intel® Rack Scale Design (Intel® RSD) Storage Services API Specification Software v2.3.2* (see Table 2).

**Table 14. PODM API—Resources and Supported Operations**

| Resource Name | Supported Operations | | | | |
|---|---|---|---|---|---|
| | GET | PATCH | POST | DELETE | Actions |
| Computer System | X | X | | | X |
| ComputerSystemMetrics | | | | | |
| Processor | X | | | | |
| ProcessorMetrics | X | | | | |
| Memory | X | | | | |
| MemoryMetrics | X | | | | |
| Storage | X | | | | |
| Drive | X | X | | | X |
| VLAN | X | | X | X | |
| Manager | X | | | | |
| Network Protocol | X | | | | |
| Ethernet Switch | X | X | | | |
| Ethernet Switch Port | X | X | X | X | |
| Ethernet Switch ACL | X | X | X | X | X |
| Ethernet Switch ACL rules | X | X | X | X | |
| Ethernet Switch Port static MACs | X | X | X | X | |
| Fabric | X | | | | |
| Zone | X | X | X | X | |
| PCIe Device | X | X | | | |
| Fabric Switch | X | | | | |
| PCIe Port | X | | | | |
| Fabric Port Metrics | | | | | |
| PCIe Function | X | | | | |
| Endpoint | X | X | X | X | |
| Storage Service | X | | | | |
| Volume | X | X | X | X | X |
| Volume Metrices | X | | | | |
| EventService | X | | | | |
| EventSubscription | X | | X | X | |

| Resource Name | Supported Operations | | | | |
|---|---|---|---|---|---|
| | GET | PATCH | POST | DELETE | Actions |
| NetworkInterface | X | | | | |
| NetworkDeviceFunction | X | X | | | |
| MetricDefinition | X | | | | |
| Thermal | X | | | | |
| Power | X | | | | |

## 4.12 Simple Storage Collection

**Table 15 Simple Storage Collection Attributes**

| Name | Simple storage | |
|---|---|---|
| Type URI | /redfish/v1/Systems/System1/SimpleStorage | |
| Attribute | Type | Description |
| Name | String | Name of collection |
| Members@odata.count | Number | Collection members count |
| Members | Array | Contains the members of this collection. |

### 4.12.1 Operations

#### 4.12.1.1 GET

**Request:**

```
GET /redfish/v1/Systems/System1/SimpleStorage
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context":
"/redfish/v1/$metadata#SimpleStorageCollection.SimpleStorageCollection",
    "@odata.id": "/redfish/v1/Systems/System1/SimpleStorage",
    "@odata.type": "#SimpleStorageCollection.SimpleStorageCollection",
    "Name": "Simple Storage Collection",
    "Members@odata.count": 1,
    "Members": [
        {
            "@odata.id": "/redfish/v1/Systems/System1/SimpleStorage/Storage1"
        }
    ]}
```

#### 4.12.1.2 PUT

Operation not allowed on this resource.

#### 4.12.1.3 PATCH

Operation not allowed on this resource.

#### 4.12.1.4 POST

Operation not allowed on this resource.

#### 4.12.1.5 DELETE

Operation not allowed on this resource.

## 4.13 SimpleStorage

`SimpleStorage` devices associated with this system.

Details of this resource are described in metadata file: `SimpleStorage.xml` .

### 4.13.1 Operations

#### 4.13.1.1 GET

**Request:**

```
GET /redfish/v1/Systems/System1/SimpleStorage/Storage1
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#SimpleStorage.SimpleStorage",
    "@odata.id": "/redfish/v1/Systems/System1/SimpleStorage/Storage1",
    "@odata.type": "#SimpleStorage.v1_1_0.SimpleStorage",
    "Id": "Storage1",
    "Name": "Simple Storage Controller",
    "Description": "System SATA",
    "UefiDevicePath": "UEFI Device Path",
    "Status": {
        "State": "Enabled",
        "Health": "OK",
        "HealthRollup": "OK"
    },
    "Devices": [
        {
             "@odata.type": "#SimpleStorage.v1_1_0.Device",
            "Name": "Drive 1",
            "Manufacturer": "ACME",
            "Model": "Drive Model string",
            "CapacityBytes": 322122547200,
            "Status": {
                "State": "Enabled",
                "Health": "OK"
            }
        },
        {
             "@odata.type": "#SimpleStorage.v1_1_0.Device",
            "Name": "Drive 2",
            "Manufacturer": "SuperDuperSSD",
            "Model": "Drive Model string",
            "CapacityBytes": 68719476736,
            "Status": {
                "State": "Enabled",
                "Health": "OK"
            }
        },
        {
            "Name": "Drive 3",
            "Status": {
```

```
                "State": "Absent"
            }
        },
        {
            "Name": "Drive 4",
            "Status": {
                "State": "Absent"
            }
        }
    ]
}
```

#### 4.13.1.2 PUT

Operation not allowed on this resource.

#### 4.13.1.3 PATCH

Operation not allowed on this resource.

#### 4.13.1.4 POST

Operation not allowed on this resource.

#### 4.13.1.5 DELETE

Operation not allowed on this resource.

## 4.14 Power

Power metrics resource represents the properties for Power Consumption and Power Limiting.

Obtain detailed information about the properties of this resource from the metadata file: `Power.xml`

### 4.14.1 Operations

#### 4.14.1.1 GET

**Request:**

```
GET /redfish/v1/Chassis/Rack1/Power
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Power.Power",
    "@odata.id": "/redfish/v1/Chassis/Rack1/Power",
    "@odata.type": "#Power.v1_1_0.Power",
    "Id": "Power",
    "Name": "Power",
    "Description": "PowerSubsystem",
    "PowerControl": [
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1/Power#/PowerControl/0",
            "MemberId": "0",
                "Name": "System Power Control",
            "PowerConsumedWatts": 8000,
```


```
            "PowerRequestedWatts": 8500,
            "PowerAvailableWatts": 8500,
            "PowerCapacityWatts": 10000,
            "PowerAllocatedWatts": 8500,
            "PowerMetrics": {
                "IntervalInMin": 30,
                "MinConsumedWatts": 7500,
                "MaxConsumedWatts": 8200,
                "AverageConsumedWatts": 8000
            },
            "PowerLimit": {
                "LimitInWatts": 9000,
                "LimitException": "LogEventOnly",
                "CorrectionInMs": 42
                },
            "RelatedItem": [
                {"@odata.id": "/redfish/v1/Chassis/Drawer1"},
                {"@odata.id": "/redfish/v1/Systems/System1"}
            ],
              "Status": {
                    "State": "Enabled",
                    "Health": "OK",
                "HealthRollup": "OK"
                },
                "Oem": {}
        }
    ],
    "Voltages": [
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1/Power#/Voltages/0",
            "MemberId": "0",
            "Name": "VRM1 Voltage",
            "SensorNumber": 11,
            "Status": {
                "State": "Enabled",
                "Health": "OK"
            },
            "ReadingVolts": 12,
            "UpperThresholdNonCritical": 12.5,
            "UpperThresholdCritical": 13,
            "UpperThresholdFatal": 15,
            "LowerThresholdNonCritical": 11.5,
            "LowerThresholdCritical": 11,
            "LowerThresholdFatal": 10,
            "MinReadingRange": 0,
            "MaxReadingRange": 20,
            "PhysicalContext": "VoltageRegulator",
            "RelatedItem": [
                {"@odata.id": "/redfish/v1/Systems/System1" }
            ]
        }
    ],
    "PowerSupplies": [
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1/Power#/PowerSupplies/0",
            "MemberId": "0",
            "Name": "Power Supply Bay 1",
            "Status": {
                "State": "Enabled",
                "Health": "Warning"
            },
            "Oem": {},
```

```
            "PowerSupplyType": "DC",
            "LineInputVoltageType": "DCNeg48V",
            "LineInputVoltage": -48,
            "PowerCapacityWatts": 400,
            "LastPowerOutputWatts": 192,
            "Model": "499253-B21",
            "Manufacturer": "ManufacturerName",
            "FirmwareVersion": "1.00",
            "SerialNumber": "1z0000001",
            "PartNumber": "1z0000001A3a",
            "SparePartNumber": "0000001A3a",
                "InputRanges": [
               {
                   "InputType": "DC",
                   "MinimumVoltage": -47,
                   "MaximumVoltage": -49,
                   "OutputWattage": 400,
                   "MinimumFrequencyHz": 50,
                   "MaximumFrequencyHz": 60,
                   "Oem": {}
               }
            ],
            "IndicatorLED": "Off",
            "RelatedItem": [
              { "@odata.id": "/redfish/v1/Chassis/Rack1" }
            ],
            "Redundancy": [
              { "@odata.id": "/redfish/v1/Chassis/1/Power#/Redundancy/0" }
            ]
        }
    ],
    "Redundancy": [
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1/Power#/Redundancy/0",
            "MemberId": "0",
            "Name": "PowerSupply Redundancy Group 1",
            "Mode": "Failover",
            "MaxNumSupported": 2,
            "MinNumNeeded": 1,
            "RedundancySet": [
                { "@odata.id": "/redfish/v1/Chassis/1/Power#/PowerSupplies/0" }
            ],
            "Status": {
                "State": "Offline",
                "Health": "OK"
            }
        }
    ],
    "Oem": {}
}
```


### 4.14.1.2 PUT

Operation not allowed on this resource.

### 4.14.1.3 PATCH

Operation not allowed on this resource.

#### 4.14.1.4 POST

Operation not allowed on this resource.

#### 4.14.1.5 DELETE

Operation not allowed on this resource.

## 4.15 Thermal

Thermal metrics resource represents the properties for Temperature and Cooling.

Obtain detailed information about the properties of this resource from the metadata file: `Thermal.xml`

### 4.15.1 Operations

#### 4.15.1.1 GET

**Request:**

```
GET /redfish/v1/Chassis/Rack1/Thermal
Content-Type: application/json
```

**Response:**

```
{
    "@odata.context": "/redfish/v1/$metadata#Thermal.Thermal",
    "@odata.id": "/redfish/v1/Chassis/Rack1/Thermal",
    "@odata.type": "#Thermal.v1_1_0.Thermal",
    "Id": "Thermal",
    "Name": "Thermal",
    "Description": "Thermal Subsystem",
    "Temperatures": [
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1/Thermal#/Temperatures/0",
            "MemberId": "0",
            "Name": "Drawer inlet Temp",
            "SensorNumber": 42,
            "Status": {
                "State": "Enabled",
                "Health": "OK"
            },
            "ReadingCelsius": 21,
            "UpperThresholdNonCritical": 42,
            "UpperThresholdCritical": 42,
            "UpperThresholdFatal": 42,
            "LowerThresholdNonCritical": 42,
            "LowerThresholdCritical": 5,
            "LowerThresholdFatal": 42,
            "MinReadingRange": 0,
            "MaxReadingRange": 200,
            "PhysicalContext": "Intake",
            "RelatedItem": [
                {"@odata.id": "/redfish/v1/Chassis/Drawer1" }
            ]
        }
    ],
    "Fans": [
        {
```

```
            "@odata.id": "/redfish/v1/Chassis/Rack1/Thermal#/Fans/0",
            "MemberId": "0",
            "Name": "BaseBoard System Fan",
            "PhysicalContext": "Backplane",
            "Status": {
                "State": "Enabled",
                "Health": "OK"
            },
            "Reading": 2100,
            "ReadingUnits": "RPM",
            "UpperThresholdNonCritical": 42,
            "UpperThresholdCritical": 4200,
            "UpperThresholdFatal": 42,
            "LowerThresholdNonCritical": 42,
            "LowerThresholdCritical": 5,
            "LowerThresholdFatal": 42,
            "MinReadingRange": 0,
            "MaxReadingRange": 5000,
            "Redundancy" : [
                {"@odata.id": "/redfish/v1/Chassis/Rack1/Thermal#/Redundancy/0"}
            ],
            "RelatedItem" : [
                {"@odata.id": "/redfish/v1/Chassis/Rack1" }
            ]
        }
    ],
    "Redundancy": [
        {
            "@odata.id": "/redfish/v1/Chassis/Rack1/Thermal#/Redundancy/0",
            "MemberId": "0",
            "Name": "BaseBoard System Fans",
            "RedundancyEnabled": false,
            "RedundancySet": [
                { "@odata.id": "/redfish/v1/Chassis/1/Thermal#/Fans/0" }
            ],
            "Mode": "N+m",
            "Status": {
                "State": "Disabled",
                "Health": "OK"
            },
            "MinNumNeeded": 1,
            "MaxNumSupported": 2
        }
    ]
}
```


#### 4.15.1.2 PUT

Operation not allowed on this resource.

#### 4.15.1.3 PATCH

Operation not allowed on this resource.

#### 4.15.1.4 POST

Operation not allowed on this resource.

#### 4.15.1.5 DELETE

Operation not allowed on this resource.

§

# 5.0 Common Property Description

## 5.1 Status

| Attribute | Type | Nullable | Description |
|---|---|---|---|
| `State` | String | Yes | Indicates the known state of the resource, as if is enabled. Allowed values (refer to section 5.2). |
| `Health` | String | Yes | Represents the health state of this resource in the absence of its dependent resources. Allowed values: Refer to section 5.3. |
| `HealthRollup` | String | Yes | Represents the overall health state from the view of this resource. Allowed values: Refer to section 5.3. |

## 5.2 Status -> State

- Enabled: This function or resource has been enabled
- Disabled: This function or resource has been disabled
- StandbyOffline: This function or resource is enabled, but awaiting an external action to activate it
- StandbySpare: This function or resource is part of a redundancy set and is awaiting a failover or other external action to activate it
- InTest: This function or resource is under test
- Starting: This function or resource is starting
- Absent: This function or resource is not installed
- UnavailableOffline: This function or resource is present but cannot be used
- Deferring: The element will not process any commands but will queue new requests
- Quiesced: The element is enabled but only processes a restricted set of commands
- Updating: The element is updating and may be unavailable or degraded

## 5.3 Status -> Health

- OK: Normal
- Warning: A condition exists that requires attention
- Critical: A critical condition exists that requires immediate attention

## 5.4 ComputerSystem.Reset

- On: Turn the system on
- ForceOff: Turn the system off immediately (nongraceful) shutdown
- GracefulRestart: Perform a graceful system shutdown followed by a restart of the system
- ForceRestart: Perform an immediate (non-graceful) shutdown, followed by a restart of the system
- Nmi: Generate a nonmaskable interrupt to cause an immediate system halt
- ForceOn: Turn the system on immediately
- PushPowerButton: Simulate the pressing of the physical power button on this system
- GracefulShutdown: Perform a graceful system shutdown and power off

## 5.5 BootSourceOverrideTarget/Supported

- None: Boot from the normal boot device
- Pxe: Boot from the preboot execution (PXE) environment
- Floppy: Boot from the floppy disk drive
- Cd: Boot from the CD/DVD disc
- Usb: Boot from a USB device as specified by the system BIOS
- Hdd: Boot from a hard drive
- BiosSetup - Boot to the BIOS Setup Utility
- Utilities: Boot the manufacturer's Utilities programs
- Diags: Boot the manufacturer's Diagnostics program
- UefiShell: Boot to the UEFI Shell
- UefiTarget: Boot to the UEFI Device specified in the UefiTargetBootSourceOverride property
- SDCard: Boot from an SD Card
- UefiHttp: Boot from a UEFI HTTP network location
- RemoteDrive: Boot from a remote drive (for example, iSCSI)

§

# 6.0 Composed Node Additional Info

## 6.1 Creating New Composed Node – Explanation

This section explains how to create  a Composed Node that satisfies a particular set of requirements using the PODM REST API.

### 6.1.1 Creating Composed Node using JSON Template

To create a Composed Node using PODM REST API it is necessary to create a JSON* template that describes the requested resources. It needs to be supplied to the PODM by performing a HTTP POST request using the Composed Node Collection Action URI located at `"/redfish/v1/Nodes/Actions/Allocate"` on the PODM service.

The JSON template may contain various details of resources used in the Composed Node. All JSON template elements are optional, but each requirement should be coherent itself. It is possible to supply the PODM with a JSON template containing no specific requirements (for example, `{}` – a pair of empty curly braces in HTTP request body) thus allowing the PODM to propose a Composed Node containing resources chosen arbitrarily by the PODM.

### 6.1.2 Specifying Requirements for a Composed Node

The JSON template contains requirements for a single Composed Node. Basic customization covers setting a `"Name"` and `"Description"` of such System (both being of type *String*). As `"Name"` parameter is required by Redfish for all resources if it is not supported the PODM will use the default name. The example shown below will allocate a single Composed Node with requested name and description:

```
{
        "Name": "Customized Composed Node name",
        "Description": "Description of a customized Composed Node"
}
```

The JSON template may contain requirements for Processors, Memory, Remote Drives, Local Drives, and Ethernet Interfaces. To specify requirements for those resources, a proper section must appear in JSON template.

### 6.1.3 General Assumptions for Allocation

Requirements are treated as minimal required value, so the resulting Composed Node may have better parameters than requested. Composed Node customization and resource customization sections described below can be used jointly.

Each resource type description has an associated table, which contains details about specific requirements. **Key** is the JSON object field. **JSON type** contains data type as defined at *https://www.json.org/*. **Allowed values** contains additional restrictions to JSON type or hints (for example, for enumerations or *Boolean* values). **Nullable** indicates if a *null* value can be passed for specified key. The **Notes, limitations** column, where present, provides additional hints about specific requirements.

#### 6.1.3.1 Location Requirements

Processor, Memory, Local Drive, and Ethernet interface secctions may contain Resource and Chassis objects. The Resource must contain the PODM URI (presented as `"@odata.id"`) of the discovered resource (Processor's URI in

Processor section, URI to Memory resource in Memory section and so on). The Chassis must contain the PODM URI of the discovered Chassis in which applicable resources will be looked for.

### 6.1.4 Specifying Processor Requirements

The JSON template may contain requirements for multiple Processors. The example below specifies requirements for a single Processor to be used in the Composed Node.

```
{
        "Processors": [{
                "Model": "Multi-Core Intel(R) Xeon(R) processor 7xxx Series",
                "TotalCores": 2,
                "AchievableSpeedMHz": 3700,
                "InstructionSet": "x86-64",
                "Oem": {
                        "Brand": "X7"",
                        "Capabilities": [
                                "sse"
                        ]
                },
                "Resource": {
                        "@odata.id": "/redfish/v1/Systems/1/Processors/1"
                },
                "Chassis": {
                        "@odata.id": "/redfish/v1/Chassis/1"
                },
               "ProcessorType": "CPU"
        }]
}
```

**Table 16. Processor Requirements**

| Attribute | Type | Allowed values | Nullable | Description |
|---|---|---|---|---|
| Model | String | | Yes | String representing Processor model. |
| TotalCores | Number | | Yes | Positive integer value expected |
| AchievableSpeedMHz | Number | | Yes | Positive integer value expected |
| InstructionSet | String | "x86", "x86-64", "IA-64", "ARM-A32", "ARM-A64", "MIPS32", "MIPS64", "OEM" | Yes | One of allowed, enumerated values |
| Oem | Object | | Yes | |
| Oem → Brand | String | "E3", "E5", "E7", "X3", "X5", "X7", "I3", "I5", "I7", "Silver", "Gold", "Platinum", "Unknown" | Yes | One of allowed, enumerated values |
| Oem->Capabilities | Array | CPU capabilities string | Yes | List of processor capabilities (like "sse") |
| Resource | Object | Exact location of a single Processor. | Yes | Refer to section 6.1.3.1. |
| Chassis | Object | Exact location of a single chassis. | Yes | Refer to section 6.1.3.1. |

| Attribute | Type | Allowed values | Nullable | Description |
|---|---|---|---|---|
| ProcessorType | String | "CPU", "FPGA", "GPU", "DSP", "Accelerator", "OEM" | Yes | One of allowed enumerated values |

The template can also provide the requirement for total processor cores available in the composed node:

```
        "TotalSystemCoreCount": 32
```

Allocation assumptions:

- Which Processors will meet supplied requirements?
  - located on the same computer system as other resources
  - with exact match on Model
  - with exact match on Brand
  - with at least TotalCores
  - with at least AchievableSpeedMHz
  - with exact match on InstructionSet
  - with exact match on ProcessorType
  - with superset of processor capabilities specified Capabilities array

If a computer system contains processors with cores that number at least TotalSystemCoreCount, it meets the requirements.

### 6.1.5 Specifying Memory Requirements

The JSON template may contain requirements for multiple Memory Modules. The example below specifies requirements for a single Memory Module used in the Composed Node.

```
{
        "Memory": [{
                "CapacityMiB": 16000,
                "MemoryDeviceType": "DDR3",
                "SpeedMHz": 1600,
                "Manufacturer": "Intel",
                "DataWidthBits": 64,
                "Resource": {
                        "@odata.id": "/redfish/v1/Systems/1/Memory/1"
                },
                "Chassis": {
                        "@odata.id": "/redfish/v1/Chassis/1"
                }
        }]
}
```

**Table 17. Memory Module Requirements**

| Attribute | Type | Allowed values | Nullable | Notes, limitations |
|---|---|---|---|---|
| CapacityMiB | Number | | Yes | Positive value expected |
| MemoryDeviceType | String | "DDR", "DDR2", "DDR3", "DDR4", "DDR4_SDRAM", "DDR4E_SDRAM", "LPDDR4_SDRAM", "DDR3_SDRAM", "LPDDR3_SDRAM", "DDR2_SDRAM", "DDR2_SDRAM_FB_DIMM", "DDR2_SDRAM_FB_DIMM_PROBE", "DDR_SGRAM", "DDR_SDRAM", "ROM", "SDRAM", "EDO", "FastPageMode", " " | Yes | One of allowed, enumerated values |
| SpeedMHz | Number | | Yes | Positive integer value expected |
| Manufacturer | String | | Yes | String representing Memory Module manufacturer name |
| DataWidthBits | Number | | Yes | Positive integer value expected. |
| Resource | Object | Exact location of a single Memory Module. | Yes | Refer to section 6.1.3.1. |
| Chassis | Object | Exact location of a single chassis. | Yes | Refer to section 6.1.3.1. |

Template can also provide requirement for total memory available in composed node, without dividing it into memory modules:

```
"TotalSystemMemoryMiB": 32000
```

Allocation assumptions:

- Which Memory Modules will meet supplied requirements?
  - With at least CapacityMiB
  - located on the same computer system as other resources
  - with exact match on MemoryDeviceType
  - with at least SpeedMHz
  - with exact match on Manufacturer
  - with at least DataWidthBits

If a computer system contains Memory Modules of least the same size as the TotalSystemMemory, it will meet requirements.

## 6.1.6 Specifying Remote Drive Requirements

The JSON template may contain requirements for multiple Remote Drives, but currently supports only one set of requirements. The example below specifies requirements for a single Remote Drive used in the Composed Node.

```
{
        "RemoteDrives": [{
                "CapacityGiB": 80,
                "Protocol": "iSCSI",
                "Master": {
                        "Type": "Snapshot",
                        "Resource": {
                                "@odata.id":
"/redfish/v1/StorageServices/iscsi1/Volumes/1"
                        }
                },
```

```
                "Resource": null
        }]
}
```

**Table 18. Requirements for a Single Remote Drive**

| Attribute | Type | Allowed Values | Nullable | Description |
|---|---|---|---|---|
| `CapacityGiB` | Number | | Yes | Positive value expected, required if Master Drive supplied. Should be at least the size of Logical Drive used as Master Drive. |
| `Protocol` | String | "iSCSI"<br>"NVMeOverFabrics" | Yes | Defines protocol used to communicate with attached drive. Currently supported:<br>`"iSCSI"`<br>`"NVMeOverFabrics"` |
| `Master` | Object | | Yes | |
| `Master →Type` | String | `"Snapshot"`,<br>`"Clone"` | No | One of allowed, enumerated values. Required if Master Drive supplied |
| `Master → Resource` | Object | | No | URI of Master volume that should be used for clone or snapshot operation |
| `Resource` | Object | | Yes | Reference to particular volume or already created endpoint. |

### 6.1.6.1 Using existing iSCSI Remote Drive

To use an existing Volume it is necessary to point to the resource (Volume) directly:

```
{
        "RemoteDrives": [{
                "Protocol": "iSCSI",
                "Resource": {
                   "@odata.id": "/redfish/v1/StorageServices/iSCSI/Volumes/1"
                }
        }]
}
```

### 6.1.6.2 Using existing NVMe* over Fabrics Remote Drive

To use an existing Volume it is necessary to point to the resource (Volume) directly:

```
{
        "RemoteDrives": [{
                "Protocol": "NVMeOverFabrics",
                "Resource": {
                   "@odata.id": "/redfish/v1/StorageServices/NVMeoE1/Volumes/1"
                }
        }]
}
```

### 6.1.6.3 Using a Master Drive for fresh Remote Drive Creation

To use a fresh Drive created from Master Drive it is necessary to:

- Set `CapacityGiB` to define capacity of the new Remote Drive that is at least of Master Drive's size,
- Set Master → Type to "Snapshot" or "Clone"
- Set Master → Resource to valid PODM URI of Volume resource used as source Drive.
- Set Protocol to the desired protocol that should be used.

***Note:*** The Snapshot does not work for the `NVMeOverFabric`s Protocol.

```
{
        "RemoteDrives": [{
                "CapacityGiB": 80,
         "Protocol": "NVMeOverFabrics",
                "Master": {
                        "Type": "Snapshot",
                        "Resource": {
                                "@odata.id":
"/redfish/v1/StorageServices/NVMeoE1/Volumes/102"
                        }
                }
        }]
}
```

### 6.1.6.4 Creating a New Empty Drive

To create a new empty Drive without any data:

- Set `CapacityGiB` to define capacity of the new Remote Drive,
- Set Protocol to the desired protocol that should be used.

***Note:*** Current iSCSI implementation requires the remote drive to be bootable. For this reason, this functionality is intended to work with the NVMe-oF Volumes only.

```
{
        "RemoteDrives": [{
                "CapacityGiB": 80,
                "Protocol": "NVMeOverFabrics",
        }]
}
```

## 6.1.7 Specifying Local Drive Requirements

The JSON template may contain requirements for multiple Local Drives (represented by Drive resource). The example below specifies requirements for a single Local Drive to be used in the Composed Node.

```
{
        "LocalDrives": [{
                "CapacityGiB": 100,
                "Type": "HDD",
                "MinRPM": 5400,
                "SerialNumber": "12345678",
                "Interface": "SATA",
                "Resource": {
                        "@odata.id": "redfish/v1/Chassis/Blade1/Drives/Disk1"
                },
                "Chassis": {
                        "@odata.id": "/redfish/v1/Chassis/Blade1"
                },
                "FabricSwitch": false
        }]
}
```

**Table 19. Local Drive Requirements**

| Attribute | Type | Allowed values | Nullable | Description |
|---|---|---|---|---|
| `CapacityGiB` | Number | | Yes | Positive value expected |
| `Type` | String | `"HDD"`, `"SSD"` | Yes | One of allowed, enumerated values |

| Attribute | Type | Allowed values | Nullable | Description |
|---|---|---|---|---|
| MinRPM | Number | | Yes | Positive integer value expected |
| SerialNumber | String | | Yes | |
| Interface | String | "SAS", "SATA", "NVMe" | Yes | One of allowed, enumerated values |
| Resource | Object | Exact location of a single Device. | Yes | Refer to section 6.1.3.1. |
| Chassis | Object | Exact location of a single Chassis. | Yes | Refer to section 6.1.3.1. |
| FabricSwitch | Boolean | "True", "false" | Yes | Determines if the drive should be connected using the fabric switch (PNC) or directly attached to the computer system. |

Allocation assumptions:

- Which Local Drives will meet supplied requirements?
  - located on the same computer system as other resources
  - with at least CapacityGiB
  - with exact match on Type
  - with at least MinRPM
  - with exact SerialNumber
  - with exact Interface

### 6.1.7.1 Pooled NVMe* Controller (PNC) Drives

If the PNC is available in the PODM, and there is no system fulfilling Local Drive requirements, PNC drives are attached to the Composed Node from the pool of available the PNC drives.

***Note:*** PNC drives that were detached from Composed Node resource (via Detach action or via DELETE operation on the node) and have the property "EraseOnDetach" set to **false** (or is set to **null)** are not erased. Their property "DriveErased" is changed to **true** and because of that, they are not available in the pool of the PNC drives ready for node composition. The PNC drives need to be selected using the Resource property, or erased by using the action "SecureErase" on drive resource.

Drives with property "DriveErased" set to **true** or **null** are available for composition without the need to specify their URI in the Resource property.

Example request that allocates the node with the PNC drive:

```
{
        "LocalDrives": [{
                "CapacityGiB": 100,
                "Type": "SSD",
                "Interface": "NVMe",
                "Chassis": {
                        "@odata.id": "/redfish/v1/Chassis/PCIeSwitchChassis"
                }
        }]
}
```

After node allocation and assembly (in "Assembled" and "Failed" ComposedNodeState), the user is able to attach and remove PNC devices (drives) using the AttachEndpoint and DetachEndpoint actions:

Example attaching drive to existing node:

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.AttachResource
Content-Type: application/json
{
    "Resource": {
       "@odata.id":"/redfish/v1/Chassis/PCIeSwitchChassis/Drives/Disk.Bay.2"
```

```
    }
}
```

Example removing drive from existing node:

```
POST /redfish/v1/Nodes/1/Actions/ComposedNode.DetachResource
Content-Type: application/json
{
    "Resource": {
        "@odata.id":"/redfish/v1/Chassis/PCIeSwitchChassis/Drives/Disk.Bay.3"
    }
}
```

### 6.1.8 Specifying Ethernet Interface Requirements

The JSON template may contain requirements for multiple Ethernet Interfaces. The example below specifies requirements for a single Ethernet Interface used in the Composed Node.

```
{
        "EthernetInterfaces": [{
                "SpeedMbps": 1000,
                "PrimaryVLAN": 100,
                "VLANs": [{
                        "VLANId": 100,
                        "Tagged": false
                }],
                "Resource": {
                        "@odata.id": "/redfish/v1/Systems/1/EthernetInterfaces/1"
                },
                "Chassis": {
                        "@odata.id": "/redfish/v1/Chassis/1"
                }
        }]
}
```

**Table 20. Ethernet Interface Requirements**

| Attribute | Type | Allowed values | Nullable | Description |
|---|---|---|---|---|
| `SpeedMbps` | Number | | Yes | Positive integer value expected |
| `PrimaryVLAN` | Number | | Yes | Positive integer value expected |
| `VLANs` | Array[Object] | | Yes | Null value will be interpreted as absence of this key.<br>Empty array [ ] will clear all existing vlans, excluding Reserved VLANs. |
| `VLANs` → `VLANId` | Number | | No | Positive integer value expected |
| `VLANs` → `Tagged` | Boolean | `"true", "false"` | No | Boolean value |
| `Resource` | Object | Exact location of a single Ethernet Interface. | Yes | Refer to section 6.1.3.1. |
| `Chassis` | Object | Exact location of a single Chassis. | Yes | Refer to section 6.1.3.1. |

Allocation assumptions:

- Which Ethernet Interfaces will meet supplied requirements?
  - located on the same Computer System as other resources
  - with at least `SpeedMbps`
  - ones that are connected with `SwitchPorts` (when VLANs section is provided)

#### 6.1.8.1 Reserved VLANs

There is a possibility to restrict usage of some VLANs by changing the configuration file located in **/etc/pod-manager/`allocation.json`.**

Example file looks like:

```
{
  "ReservedVlanIds": [1, 170, 4088, 4091, 4094]
}
```

Where 1, 170, 4088, 4091, 4094 are VLANs, which are reserved. Reserved VLANs have following implications:

- Allocation JSON cannot contain such VLANs and such requests result in an error
- Reserved VLANs are not deleted during allocation
- Reserved VLANs are not deleted during disassembly

### 6.1.9 Specifying Security Requirements

The JSON template may contain requirements for the security of Composed Node. The example below specifies the requirements for a Trusted Platform Module (TPM) v2.0 present in the Composed Node.

```
{
        "Security": {
          "TpmPresent": true,
          "TpmInterfaceType": "TPM2_0"
        "ClearTPMOnDelete": true
        }
}
```

**Table 21. Security Requirements**

| Attribute | Type | Allowed values | Nullable | Description |
|---|---|---|---|---|
| `TpmPresent` | Boolean | | Yes | Boolean expected |
| `TpmInterfaceType` | String | | Yes | String value (enum) expected. Possible values for this field are defined in metadata file `ComputerSystem.xml` as a `InterfaceType` |
| `TxtEnabled` | Boolean | | Yes | Boolean expected |
| `ClearTPMOnDelete` | Boolean | | Yes | Boolean expected |

The PODM will allocate the Composed Node that meet security requirements listed in Section 6.1.9, Specifying Security Requirements.

- `TpmPresent` – determine if Composed Node should be equipped with TPM module
- `TpmInterfaceType` – overrides `TpmPresent` parameter (if specified TPM module expected). System must be equipped with only defined TPM interface type.
- `TxtEnabled` – determine if Composed Node should have Trusted Execution Technology (TXT) mode enabled.
- `ClearTPMOnDelete` - used to specify if TPM module should be cleared on composed node `DELETE` request.

### 6.1.10 Allocation Algorithm

Node composition starts with HTTP POST request of JSON template on "`/redfish/v1/Nodes/Actions/Allocate`" Composed Node Collection Action URI on PODM Service. If the JSON template is well formed, and contains a supported set of requirements, the allocation process begins. Four major scenarios are currently supported:

- Allocating resources for Composed Node to be booted from Local Drive.

- Allocating resources for Composed Node to be booted from existing Remote Drive.
- Allocating resources for Composed Node to be booted from Remote Drive that need to be created.
- Allocating resources for Composed Node with VLAN requirements specified. This scenario is used with one of other three.

The allocation process is preceded by a general verification of JSON template that checks if requested node can be realized by available resources and consists of:

- Selecting and allocating a Computer System that contains resources matching template requirements for Processors, Memory, Local Drives, and Ethernet Interfaces.
- Selecting or creating Remote Drive to be used with previously selected Computer System and allocating it.

#### 6.1.10.1 Detailed Process of Selecting and Allocating a Computer System for a Composed Node:

- Find all Computer Systems that are not yet allocated (not used by any other allocated Composed Node) with Status Enabled and Health OK.
- Filter Computer Systems by specified Resource and Chassis (if supplied in template)
- **Filter Computer Systems by Processors**: return all Computer Systems that contain at least requested quantity of Processors that meet requirements (if supplied in template):
  - Exactly matching requested model,
  - Exactly matching requested brand,
  - With at least requested number of cores,
  - With at least requested frequency,
  - Exactly matching requested instruction set.
- **Filter Computer Systems by Memory:** return all Computer Systems with at least total requested size of memory located on Memory Modules that each of them meet requirements (if supplied in template):
  - Memory of exactly requested DIMM device type
  - With at least requested speed MHz
  - With exact requested manufacturer
  - With at least requested data width bits
- **Filter Computer Systems by Local Drives:** return all Computer Systems that contain for each requested Drive one distinct Device meeting requirements (if supplied in template):
  - With at least requested capacity specified
  - Exactly matching requested Drive type
  - With at least requested min RPM
  - With exact requested serial number
  - With exact Interface
- **Filter Computer Systems by Ethernet Interfaces:** return all Computer Systems that contain for each requested Ethernet Interface one distinct Ethernet Interface meeting requirements (if supplied in template):
  - With at least requested speed.
  - If VLANs section is provided then Computer Systems with Ethernet Interfaces which are not connected with `EthernetSwitchPorts` are filtered out (as described below)
- A first Computer System from resulting filtered collection is then allocated to be used in the Composed Node.

#### 6.1.10.2 Connection between Computer System Ethernet Interface and Ethernet Switch Port

To enable particular VLAN usage on the Composed Node, there is a need to map the Ethernet Switch Port and Computer System's Ethernet interface. This mapping is done using the MAC address as an identifier. Fields used for this mapping are:

- `NeighborMAC` on `EthernetSwitchPort` resource
- `MacAddress` on `EthernetInterface` resource

If those two properties contain the same value, the Computer System's Ethernet Interface and Ethernet Switch Port are treated as connected. Only Computer Systems with the Ethernet Interface, connected to Ethernet Switch Ports, could be used in allocation with specified VLANs requirement.

### 6.1.10.3 Detailed Process of Selecting Remote Drives:

1. Determine what type of Remote Drive is requested.
2. When requesting an existing Remote Drive:
   - Find all Targets that are not yet allocated (not used by any allocated Composed Node),
   - Find the first Target that exactly matches the requested IQN and allocate it to be used in the Composed Node.
3. When requesting a new Remote Drive:
   a. Check if Target exists with requested IQN to be set for the newly created target,
   b. Check if the Logical Drive requested as Master Drive exists on the Storage Service handled by the PODM, and select the Storage Service to handle a new Target creation.
   c. Find all Logical Volume Groups meeting requirements:
      - Located on selected Storage Service
      - Having free space of at least requested capacity for a new Remote Drive
        – A first Logical Volume Group from resulting filtered collection is selected as a placement for new Logical Volume, which will be exposed as a new Target (Remote Drive)
        – A new Logical Volume is created on selected Logical Volume Group (as a snapshot or as a clone)
        – A new Target is created on top of newly created Logical Volume
        – Newly created Target is allocated to be used in Composed Node

### 6.1.10.4 Post-allocation Scenarios

A Composed Node is created as a new REST resource at `/redfish/v1/Nodes/ {NodeId}` when a proper Computer System was found and was successfully allocated. The state of Composed Node is set to `"Allocated"`. An `"Allocated"` Composed Node is a PODM proposition that can be either accepted or rejected.

- If accepted, the user has to send a HTTP POST request on `ComposedNode.Assemble` action of the proposed Composed Node to assemble it:
  – If no Remote Drive was requested, a Composed Node's state is set to `"Assembled"`.
  – When the Remote Drive is requested, Composed Node remains `"Assembling"` until Target creation finishes. When the Target is successfully assembled to be used with the Composed Node, the node's state is set to `"Assembled"`.
  – The assembly process does not end with sending power on request, so it is necessary to perform a `ComposedNode.Reset` action to power on a Composed Node after assembly.
- If rejected, a user can continue sending HTTP POST requests of the JSON template on `/redfish/v1/Nodes/Actions/Allocate` to create more proposals from which to pick. When finding the right pick, it is recommended to send HTTP DELETE on all rejected proposals of the Composed Nodes to free resources allocated by them.

## 1.1 Remote Drive lifecycle

Remote drives using `NVMeOverFabrics` or iSCSI protocols can be attached and detached to/from Composed Node during allocation and removal. In addition, it is possible in runtime using `Attach/DetachResource` actions.

To make sure user data are safe and not provided to other users, all volumes by default are erased on detach. If data should be kept, user shall change `EraseOnDetach` property on each volume. Below table summarizes what actions are made on each scenario.

| Action | EraseOnDetach=false | EraseOnDetach=true |
|---|---|---|
| POST `Node.DetachResource` | 1. Volume **not** erased<br>2. Volume **not** deleted | 1. Volume erased<br>2. Volume **not** deleted |
| DELETE `node` | 1. Volume **not** erased<br>2. Volume **not** deleted | 1. Volume erased<br>2. Volume deleted |

***Note:*** When the remote drive is detached using the `Node.DetachResource` action pointing to the Endpoint, the associated Endpoint will not be deleted by the PODM and will prevent the Volume from deletion.

§