opCharts Node Resource API

Node Resource API

Nodes are gathered from the NMIS installation on the existing server. Each node has a list of resources. Each resource can either be indexed or non-indexed. If non-indexed each resource contains a list of datasets (which are individual data points over time, eg. avgbusy5). If indexed, the resource contains an array, each element in that array contains datasets.

Authentication is required to access all methods below.

API Endpoint

Some of these API endpoints will return HTML or JSON data, to ensure you get JSON data, use the HTTP Header "accept:application/json". Alternatively you can append ".json" onto some requests.

All requests are made under the following base URL:

```
http[s]://server/omk/opCharts/
```

For clarity the methods below include the base of the URL in the HTTP request line.

GET nodes (fetch list of nodes)

`GET /omk/opCharts/nodes/`

Retrieves the nodes available to opCharts. No parameters are accepted.

This endpoint requires the use of the "accept:application/json" header, or you can use "GET /omk/opCharts/nodes.json".

Successful Response

Returns an array of strings containing the node names

```
[
  "odin",
  "sparkle",
  "maverick",
]
```

GET node resources (fetch resources for a single node)

`GET /omk/opCharts/nodes/[node]/resources/`

Retrieves all resources for the specified node. A resource is required when requesting specific datasets or indexed datasets.

Successful Response

Returns an array of resource objects. Each resource object has a value it can be identified by(which can be used in later requests) and a url that the resource value can be appended to in order to fetch information about this resource and it's name which can be displayed to users.

The resource block also contains an entry called "active_indexes", if this section is null the resource is not indexed, if it is an empty array the resource is indexed but no indexes exist. If the entry contains array entries then these indexes are actively collected (to get a full list use the resource indicies below)

```
[
  {
    "active_indexes": null,
    "value": "nodehealth",
    "url": "/omk/opCharts/nodes/asgard/resources",
    "name": "nodehealth"
  },
  {
    "active_indexes": [ "192.168.90.17" ],
    "value": "bgpPeer",
    "url": "/omk/opCharts/nodes/asgard/resources",
    "name": "bgpPeer"
  },
  {
    "active_indexes": null,
    "value": "mib2ip",
    "url": "/omk/opCharts/nodes/asgard/resources",
    "name": "mib2ip"
  },
  {
    "active_indexes": [ "2"],
    "value": "cbqos-out",
    "url": "/omk/opCharts/nodes/asgard/resources",
    "name": "cbqos-out"
  },
  {
    "active_indexes": [ "2", "1" ],
    "value": "pkts_hc",
    "url": "/omk/opCharts/nodes/asgard/resources",
    "name": "pkts_hc"
  },
  {
    "active_indexes": [ "2", "1" ],
    "value": "interface",
    "url": "/omk/opCharts/nodes/asgard/resources",
    "name": "interface"
  }
]
```

GET node resource datasets (fetch datasets for a single node & resource)

`GET /omk/opCharts/nodes/[node]/resources/[resource]/datasets.json`

Retrieves all datasets for the specified node & resource.

Successful Response

An array of dataset objects. Each dataset object lists it's resource, the value it can be identified by, a url that the value can be appended to (if not ending in .json), tokens used for searching and it's name which can be used for displaying to users.

Non-indexed:

```
[
  {
    indexed: "",
    resource_id: "mib2ip",
    value: "ipInAddrErrors",
    url: "/omk/opCharts/nodes/asgard/resources/mib2ip/datasets",
    tokens: [
      "ipInAddrErrors",
      "mib2ip"
    ],
    name: "ipInAddrErrors"
  },
  ...
]
```

If a dataset is indexed it must be accessed through one of the indexes provided by the resource. This list can be found using the url listed in the dataset object.

Indexed:

```
[
  {
    indexed: "1",
    resource_id: "interface",
    value: "ifInOctets",
    url: "/omk/opCharts/nodes/asgard/resources/interface/indicies.json",
    tokens: [
      "ifInOctets",
      "interface"
    ],
    name: "ifInOctets"
  },
  ...
]
```

GET node resource indexes (fetch indexes for a node resource)

`GET /omk/opCharts/nodes/[node]/resources/[resource]/indicies.json`

Retrieve all indexes available from a resource. This is only required if the dataset you wish to load is indexed.

Successful Response

Returns a hash, with the name of the resource. A datum array of index objects, these hold the data you are likely looking for. The value attribute holds the identifier for this object. Name is the attribute which can be displayed to users. The "active" property tells you if this index is being collected.

The list of datasets available for the specified resource along with the list of indexes create an N x M matrix of possible datasets.

```
{
  name: "interface",
  datum: [
    {
      resource_id: "interface",
      value: "2",
      url: "/omk/opCharts/nodes/asgard/resources/interface/datasets",
      "active": 1,
      tokens: [
        "WAN/DSL",
        "FastEthernet0/1",
        "2"
      ],
      name: "WAN/DSL -- FastEthernet0/1 -- 2"
    }
  ],
  header: "interface"
}
```

If the resource is CBQoS and the index holds CBQoS data there will be an array of classes available from this index.

Another example, with CBQoS data:

```
{
  "name": "cbqos-out",
  "datum": [
    {
      "classes": [
        "InternetControl",
        "class-default",
        "Interactive",
        "Voice",
        "VPN",
        "WebTraffic"
      ],
      "resource_id": "cbqos-out",
      "value": "2",
      "active": 1,
      "url": "/omk/opCharts/nodes/asgard/resources/cbqos-out/datasets",
      "tokens": [
        "WAN/DSL",
        "FastEthernet0/1",
        2
      ],
      "name": "WAN/DSL -- FastEthernet0/1 -- 2"
    }
  ],
  "header": "cbqos-out"
}
```


Performance Data

GET/POST node resource dataset (fetch data for a single dataset from a node resource)

`GET /omk/opCharts/data_model_view/new?requestData=[JSON object]`

`POST /omk/opCharts/data_model_view/new`

`  BODY WITH JSON OBJECT = { contents listed below }`

Request data from a dataset. There are 2 basic ways to request data.

1. Request NMIS dataset information for a specific dataset from an RRD file (nmis_rrd)
2. Request NMIS graph information, this provides the same data that is displayed in NMIS graphs and are defined by nmis Graph-*.nmis files (nmis_graph)

The request structure is not simple. The required inputs are resource, dataset, possibly index.

Required Parameters

Parameters		
{		

<table>
<tr><td></td><td>requestData (struct)</td><td>
<table>
<tr><td>model</td><td>string</td><td>"nmis_rrd" or "nmis_graph". nmis_rrd will provide data directly as it is in the RRD files. nmis_graph will provide data after running it through the manipulations done by the NMIS graph</td></tr>
<tr><td>model_view</td><td>string</td><td>"graph", no other options supported at this time</td></tr>
<tr><td>data_source</td><td>string</td><td>"local_nmis", no other options supported at this time</td></tr>
<tr><td>options</td><td>struct</td><td>* can be empty for now, so just {}</td></tr>
<tr><td>parameters</td><td>struct</td><td>
<table>
<tr><td>start_date_raw</td><td>number</td><td>unix time for data to start</td></tr>
<tr><td>end_date_raw</td><td>number</td><td>unix time for data to end</td></tr>
<tr><td>node</td><td>string</td><td>node name, found using GET node</td></tr>
<tr><td>graph_type</td><td>string</td><td>If using nmis_rrd: maps to a resource found using Get Node Resources (value attribute, which is the ID).

If using nmis_graph: maps to the name of the graph if not indexed, maps to the same as nmis_rrd if indexed</td></tr>
<tr><td>field</td><td>string</td><td>Only used for nmis_rrd: maps to the dataset required found using GET node resource datasets (value attribute of desired dataset)</td></tr>
<tr><td>resource_index</td><td>string/number</td><td>If the resource is indexed, the index of the resource required, found using GET node resource indexes</td></tr>
<tr><td>index_graph_type</td><td>string</td><td>Only used if the resource is indexed:

nmis_rrd: same as graph_type

nmis_graph: the name of the graph</td></tr>
<tr><td>item</td><td>string</td><td>CBQoS class name, found by using "Get node resource index"</td></tr>
<tr><td>axis</td><td>number</td><td>0 or 1, the axis the dataset will group itself into, usually 0</td></tr>
<tr><td>mode</td><td>string</td><td>If using nmis_rrd: Defines the "Consolidation Function" that is used to on the data when it needs to change resolution. Defaults to AVERAGE, other options: MAX, MIN. If changes have been made to default settings not all of these may be available for all time periods

If using nmis_graph: mode/CF functions are built into the graph definitions, after changing the definition reloading the dataset the data will use the new definition.</td></tr>
</table>
</td></tr>
</table>
</td></tr>
<tr><td>}</td><td></td><td></td></tr>
</table>

Request Example

```
"requestData": { # remove "requestData" here if posting
  "model": "nmis_rrd",
  "model_view": "graph",
  "data_source": "local_nmis",
  "parameters": {
    "start_date_raw": 1401346748,
    "end_date_raw": 1401951548,
    "node": "asgard",
    "graph_type": "nodehealth",
    "field": "avgBusy5",
    "resource_index": "",
    "index_graph_type": "",
    "axis": "0"
  },
  "options": {
  }
}
```

Successful Response

A successful response will be a hash with 2 objects. The requestObject that was submitted along with a responseObject :

```
{
  "requestData": {
    ... <snip, same as above >...
  },
  "replyData": {
    "options": {
      "subtitleText": null,
      "titleText": null,
      "yAxis1TitleText": null,
      "plotBands": [],
      "yAxis0TitleText": null
    },
    "min": null,
    "stacking": null,
    "max": null,
    "meta_data": {
      "time_start": 1400206204,
      "start_date_input": "2014-05-16 12:10:04",
      "end_date_input": "2014-05-23 12:10:04"
    },
    "data": [
      {
        "yAxis": 0,
        "color": null,
        "stack": null,
        "name": "avgBusy5",
        "type": null,
        "data": [
          [
            1400206500000,
            null
          ],
          [
            1400206800000,
            null
          ],
          [
            1400571000000,
            6
          ],
          [
            1400571300000,
            6.99308580333333
          ],
          [
            1400571600000,
            9.98057514
          ],
          [
            1400571900000,
            8.00908836666667
          ],
          [
            1400572200000,
            6.01230244666667
          ],
          [
            1400572500000,
            3.02188843
          ],
          [
            1400572800000,
            2.00550849666667
          ],
        ]
      }
    ]
  }
}
```

For nmis_rrd, the important data is held in replyData->data->[0]->data, this is an array holding [time, value]. Time is in Javascript time format (and not unix epoch, to get this simply divide by 1000).

For nmis_graph, the data is almost the same but there may be more than 1 entry in replyData->data. There will be one per dataset that is on the graph, for example avgbusy1, avgbusy5.

If there is no data for that time in NMIS null will be returned.