Node management

- Node creation
- Node changing
- Node deletion
- Obtaining node information
- Obtaining node current state

Nodes can be managed using Websocket API orREST API.

Node creation

New node is created automatically when WCS server, which is not in the database, connects to backend server.

Node changing

Node parameters can be changed with /api/node/update request

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/node/update
	content-length:121
	
	{
	"requestId":"70ed9798-6463-4741-818f-b0b9d7299b6f",
	"realm":"/api/node/update",
	"payload":
	{
	"id":"2",
	"profiles":["1","2"]
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;
		charset=UTF-8
		subscription:sub-1
		message-id:3-25
		content-length:84
		
		{
		"requestId":"70ed9798-6463-4741-818f-b0b9d7299b6f",
		"status":200,
		"reason":"SUCCESS"}
			200 OK
			
			404 Object not found
			
			500 Persist exception

REST API	POST: /api/node/update "application/json; charset=utf-8"
	{
	"id":"2",
	"profiles":["1","2"]
	}
		{
		"status":200,
		"reason":"SUCCESS"
		}
			200 OK
			
			404 Object not found
			
			500 Persist exception

Where

- id – node identifier
- profiles –list of identifiers of profiles defining which metrics are acquired from this node

Node deletion

A node can be deleted with /api/node/delete request

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/node/delete
	content-length:100
	
	{
	"requestId":"a2d3fd81-4598-41cf-8840-e40770d597b4",
	"realm":"/api/node/delete",
	"payload":
	{
	"id":"2"
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;
		charset=UTF-8
		subscription:sub-1
		message-id:3-25
		content-length:84
		
		{
		"requestId":"a2d3fd81-4598-41cf-8840-e40770d597b4",
		"status":200,
		"reason":"SUCCESS"}
			200 OK
			
			404 Object not found
			
			500 Persist exception

REST API	POST: /api/node/delete "application/json; charset=utf-8"
	{
	"id":"2"
	}
		{
		"status":200,
		"reason":"SUCCESS"
		}
			200 OK
			
			404 Object not found
			
			500 Persist exception

Where

- id –node identifier

The node monitoring stops when it is deleted.

Obtaining node information

A node information can be obtained with /api/node/list request

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/node/list
	content-length:98
	
	{
	"requestId":"5c809c02-1082-418e-8423-db7dbff9605c",
	"realm":"/api/node/list",
	"payload":
	{
	"id":"3"
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;charset=UTF-8
		subscription:sub-1
		message-id:3-28
		content-length:143
		
		{
		"requestId":"5c809c02-1082-418e-8423-db7dbff9605c",
		"status":200,
		"reason":"SUCCESS",
		"payload":[
		{
		"id":3,
		"host":"192.168.1.64",
		"profiles":[1]
		}
		]
		}
			200 OK
			
			404 Object not found
			
			500 Persist exception

REST API	POST: /api/node/list "application/json; charset=utf-8"
	{
	"id":"3"
	}
		{
		"status": 200,
		"reason": "SUCCESS",
		"payload": [
		{
		"id": 3,
		"host": "192.168.1.64",
		"profiles": [
		1
		]
		}
		]
		}
			200 OK
			404 Object not found
			500 Persist exception

Where

- id –node identifier
- host - WCS server address
- profiles - profiles list

If node Id is set, the response will contain only that node information.If Id is not set,the response will contain list with all the nodes on backend server.

Obtaining node current state

A node current state can be obtained with /api/node_status/list query

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/node_status/list
	content-length:105
	
	{
	"requestId":"404ae542-b729-4b6e-bf28-ef85dc1f8630",
	"realm":"/api/node_status/list",
	"payload":
	{
	"id":"3"
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json; charset=UTF-8
		subscription:sub-1
		message-id:3-31
		content-length:171
		
		{
		"requestId":"404ae542-b729-4b6e-bf28-ef85dc1f8630",
		"status":200,
		"reason":"SUCCESS",
		"payload":[
		{
		"id":3,
		"rate":160
		}
		]
		}
			200 OK
			404 Object not found
			500 Persist exception

REST API	POST: /api/node_status/list "application/json; charset=utf-8"
	{
	"id":"3"
	}
		{
		"status": 200,
		"reason": "SUCCESS",
		"payload": [
		{
		"id": 3,
		"rate": 160
		}
		]
		}
			200 OK
			404 Object not found
			500 Persist exception

Where

- id – node identifier
- rate – metrics acquisition rate

If node Id is set, the response will contain only that node state.If Id is not set,the response will contain current state of all the nodes on backend server.

If the node is not connected to backend server, or if this node never be connected, the response will be 404 OBJECT_NOT_FOUND