developer.skatelescope.org Documentation

Release 0.1.0-beta

Marco Bartolini

Sep 20, 2022

Table of Contents

CHAPTER 1

Quickstart

This project uses Docker containers for development and testing, and `make` to provide a consistent UI.

Build a new Docker image and execute the test suite with:

```
make test
```

Launch an interactive shell inside a container, with your workspace visible inside the container, with:

```
make interactive
```

To list all available targets, execute `make` without any arguments, e.g.,

```
equuleus:ska-tmc-cdm $ make
build                          build the application image
down                           stop develop/test environment and any interactive␣
↪session
help                           show this help.
interactive                    start an interactive session using the project image␣
↪(caution: R/W mounts source directory to /app)
lint                           lint the application (static code analysis)
piplock                        overwrite Pipfile.lock with the image version
pull                           download the application image
push                           push the image to the Docker registry
test                           test the application
up                             start develop/test environment
```

CHAPTER 2

Background

SKA Tango devices have commands that accept structured arguments and/or return structured responses. These structured data are often expressed as JSON-formatted strings.

The Configuration Data Model (CDM) is a data model used to describe subarray resource allocations and the subsequent configuration of those resources. It is effectively the superset of the configurations used by receptors, correlators, and data processing systems. The CDM is one such example of structured data delivered to TMC Tango devices.

This project defines object representations of the structured data passed to and from Tango devices, and serialisation schema used to convert the structured data to and from JSON. This project defines:

1. a Python object model of the CDM;
2. a Python object model for the structured arguments sent to TMC Tango devices and the structured responses received in return;
3. serialisation schema to convert the Python object model instances to and from JSON.
4. validation of the JSON strings sent between devices are compliant with the agreed interfaces.

The primary users of this shared library are the OET, SubArrayNode, and CentralNode. The OET uses this library to construct object representations of telescope configurations and resource allocation instructions, to convert those object representations to JSON-formatted payloads for TMC devices, and finally to convert the JSON responses returned by TMC devices back into Python objects.

It is intended that TMC devices also use this library to guarantee correct data exchange with the OET. TMC can also use this library to marshall and unmarshall its arguments to CSP and SDP Tango devices, which accept the appropriate subset of the JSON.

CHAPTER 3

Project layout

The CDM project contains three top-level packages, `ska_tmc_cdm.messages`, `ska_tmc_cdm.schemas` and `ska_tmc_cdm.jsonschema` as shown in the figure below. The `ska_tmc_cdm.messages` package contains Python object models for the JSON command arguments agreed in the ICDs. The `ska_tmc_cdm.schemas` package contains code to transform the classes defined in `ska_tmc_cdm.messages` to and from JSON. The `ska_tmc_cdm.jsonschema` package contains code to verify that the JSON strings sent between devices are compliant with the agreed interfaces.

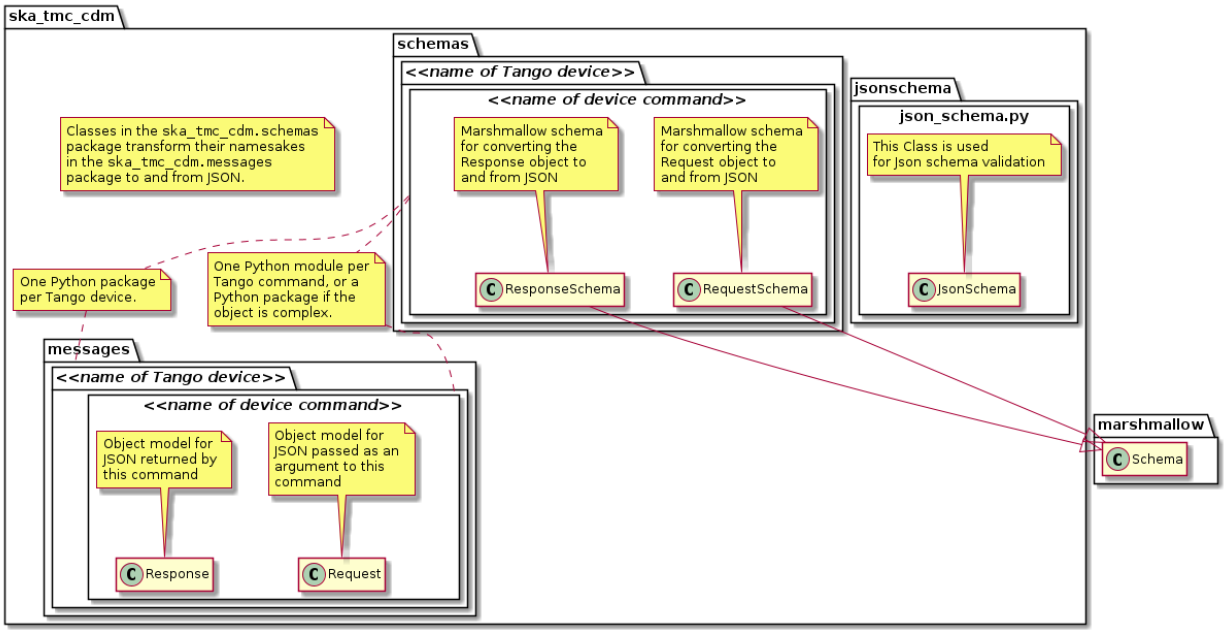


Fig. 1: Project layout and naming conventions.

The project layout and naming conventions are:

- Each Tango device has a corresponding Python sub-package in `ska_tmc_cdm.messages` and `ska_tmc_cdm.schemas`.
- Code and schema for each Tango device command are located in Python modules inside their respective package.
- Structured input for the Tango command is modelled by a `Request` object.
- Structured output from the command is modelled by a `Response` object.
- Marshmallow schema are created to transform Python `Request` and `Response` instances to an from JSON, along with any other content they contain.

3.1 Messages

The Python object model for the JSON defined in the ICD is located in the `ska_tmc_cdm.messages` package. In general, each CDM JSON entity is represented as a Python class and each CDM attribute presented as a class property.

CDM attributes can be typed as plain Python data types (strings, floats, etc.) or, where appropriate, represented by rich objects if this provides additional value to the client. For example, while astronomical coordinates are represented by floats and strings in the JSON schema, in the object model they are defined as Astropy SkyCoord instances to ensure ensure correct coordinate handling and permit easier manipulation downstream. Similarly, quantities with units could be defined as instances of Astropy Quantity to provide additional functionality.

For details on the device messages modelled by this library, see:

- *TMC CentralNode*
- *TMC SubArrayNode*
- *MCCSController*
- *MCCSSubarray*

3.2 Marshmallow Schemas

Classes to marshall the `ska_tmc_cdm.messages` objects to and from JSON are defined in the `ska_tmc_cdm.schemas` package. The ska-tmc-cdm project uses Marshmallow for JSON serialisation. Classes in the `ska_tmc_cdm.schemas` define Marshmallow schemas which are used by Marshmallow during JSON conversion.

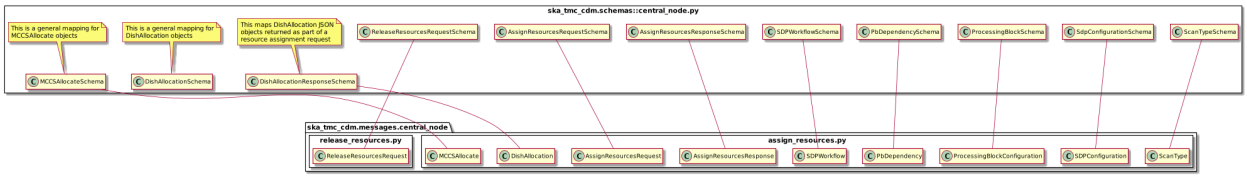


Fig. 2: Schema mapping for objects used to communicate with TMC CentralNode device.

3.3 JSON Schemas

The CDM library uses the SKA Telescope Model to ensure the JSON accepted and JSON generated by the library are compliant with the schema declared by the data.

The entry points for code handling JSON schema validation is located in the `ska_tmc_cdm.jsonschema` module. This module contains methods for fetching version-specific JSON schemas using interface URI and validating the

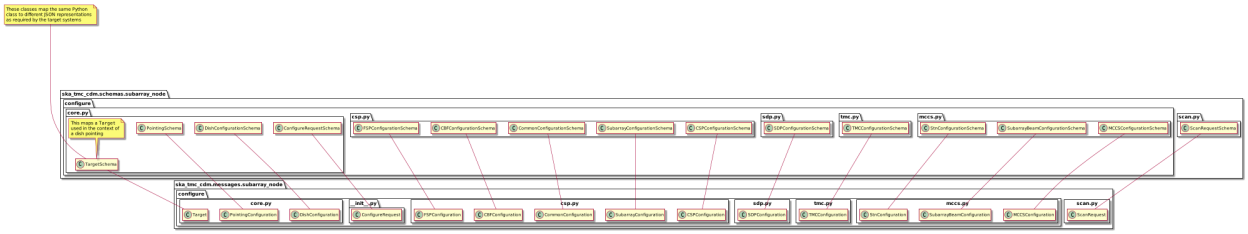


Fig. 3: Schema mapping for objects used to communicate with TMC SubArrayNode device.

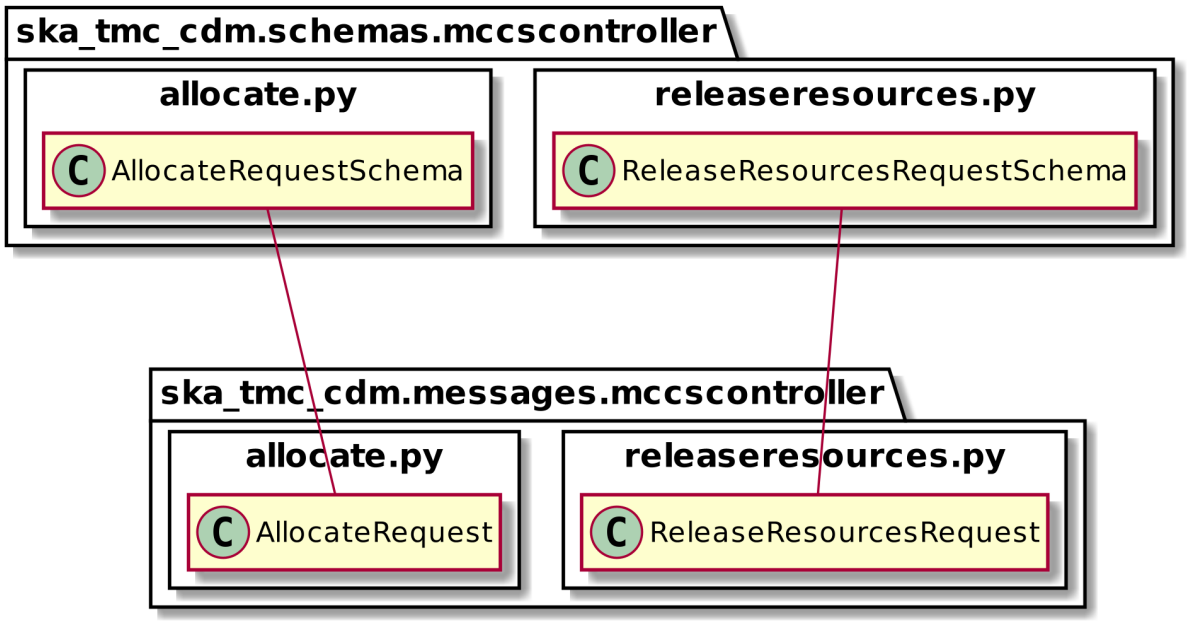


Fig. 4: Schema mapping for objects used to communicate with MCCSController device.

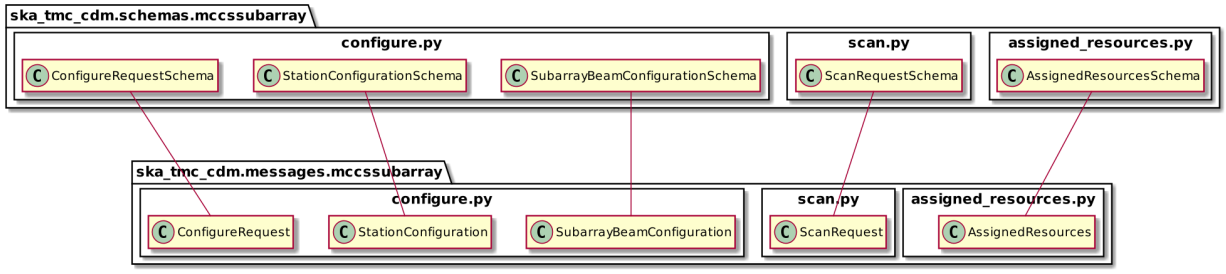
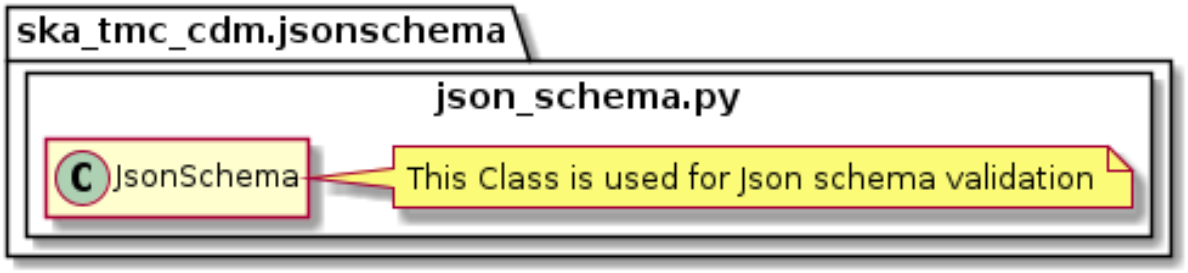


Fig. 5: Schema mapping for objects used to communicate with MCCSSubarray device.

structure of JSON against these schemas. Json Schema validation functionality is enabled by default with the parameter `validate=True` when converting a JSON string to CDM using `ska_tmc_cdm.schemas.CODEC.loads()` and when converting CDM to a JSON string using `ska_tmc_cdm.schemas.CODEC.dumps()`.



CHAPTER 4

Extending the CDM

Additional devices and applications cay use this library to communicate CDM elements wherever useful. Developers are encouraged to extend the ska-tmc-cdm project, adding object models and schemas for the structured arguments for their Tango devices.

The steps to extend the CDM are:

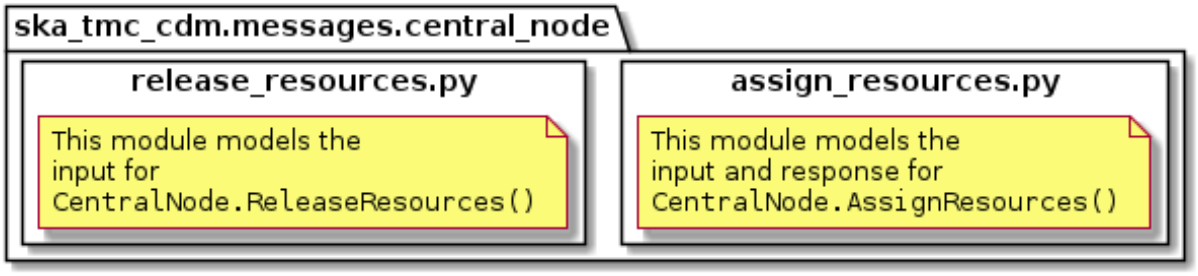
1. Create a new package for the Tango device in `ska_tmc_cdm.messages`.
2. For each device command, create a new module in the new package.
3. If the command accepts structured input, define a `Request` class in the module.
4. If the command returns a structured response, define a `Response` class in the module.
5. With the Python object model defined, create a corresponding package and module structure in `ska_tmc_cdm.schemas`.
6. In the schema module, define Marshmallow schemas to convert the object model classes and any structure to JSON.
7. If this is a major entity, register the schema with the `ska_tmc_cdm.schemas.CODEC` object using the `@CODEC.register_mapping` decorator.

CHAPTER 5

TMC CentralNode

5.1 Overview

Sub-array resource allocation is achieved via communication with a TMC CentralNode device. The `centralnode` package models the JSON input and responses for TMC CentralNode commands. The contents of this package are shown in the figure below.



Classes in the *assign_resources.py* module model the arguments for the `CentralNode.AssignResources()` command.

Classes in the *release_resources.py* module model the arguments for the `CentralNode.ReleaseResources()` command.

5.2 assign_resources.py

The `assign_resources.py` module models the the JSON input and response for a `CentralNode.AssignResources()` command.

Example JSON input modelled by `AssignResourcesRequest` for MID:

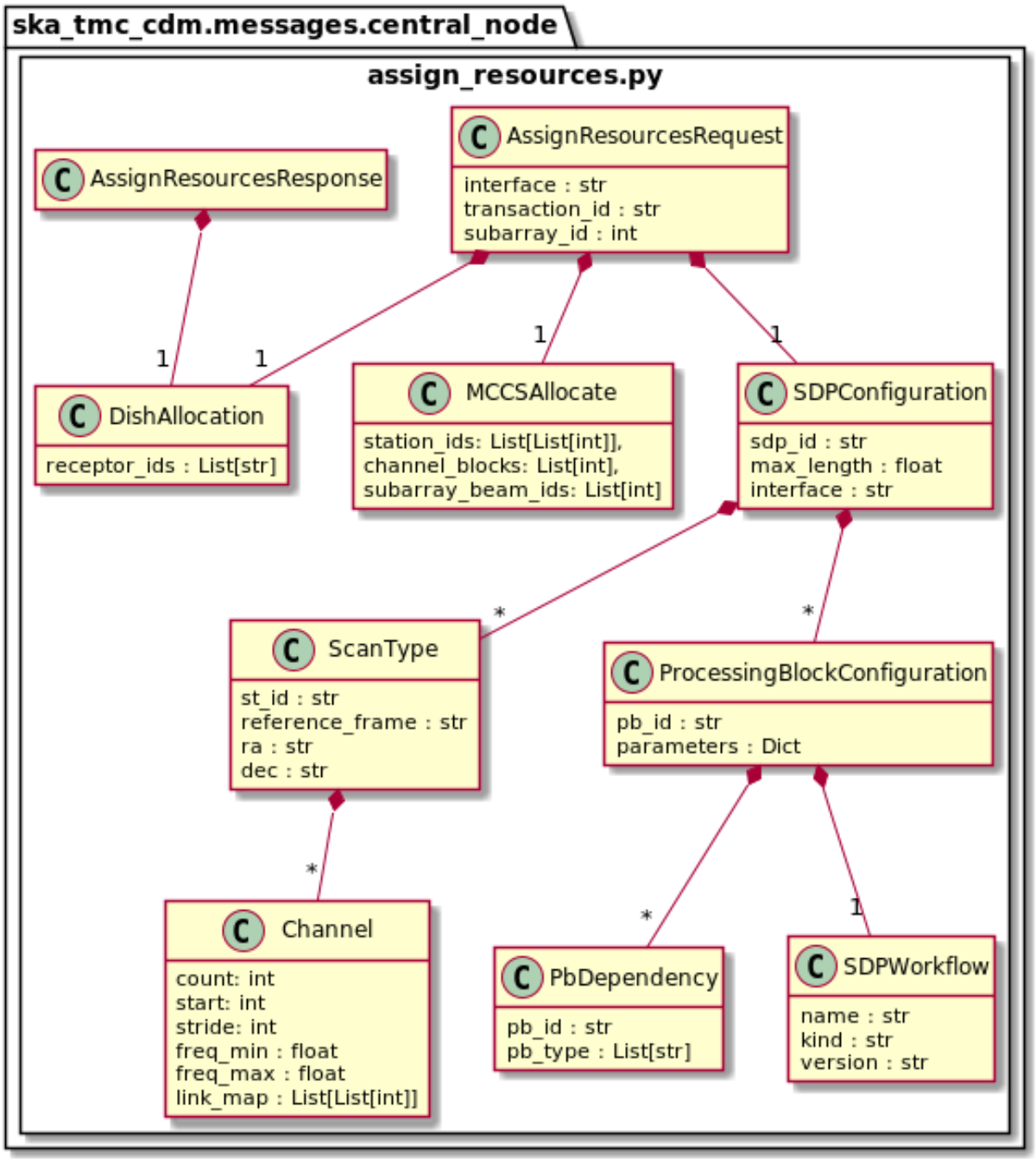


Fig. 1: assign_resources.py object model

```
{
  "interface": "https://schema.skao.int/ska-tmc-assignresources/2.0",
  "transaction_id": "txn-mvp01-20200325-00001",
  "subarray_id": 1,
  "dish": {
    "receptor_ids": ["0001", "0002"]
  },
   "sdp": {
      "interface": "https://schema.skao.int/ska-sdp-assignresources/2.0",
      "eb_id": "eb-mvp01-20200325-00001",
      "max_length": 100.0,
      "scan_types": [
        {
          "scan_type_id": "science_A",
          "reference_frame": "ICRS", "ra": "02:42:40.771", "dec": "-00:00:47.84",
          "channels": [{
             "count": 744, "start": 0, "stride": 2, "freq_min": 0.35e9, "freq_max": 1.
↪05e9,
             "link_map": [[1,0], [101,1]]
          }]
        },
        {
          "scan_type_id": "calibration_B",
          "reference_frame": "ICRS", "ra": "12:29:06.699", "dec": "02:03:08.598",
          "channels": [{
            "count": 744, "start": 0, "stride": 2, "freq_min": 0.35e9, "freq_max": 1.
↪05e9,
            "link_map": [[1,0], [101,1]]
          }]
        }
      ],
      "processing_blocks": [
        {
          "pb_id": "pb-mvp01-20200325-00001",
          "workflow": {"kind": "realtime", "name": "vis_receive", "version": "0.1.0"},
          "parameters": {}
        },
        {
          "pb_id": "pb-mvp01-20200325-00002",
          "workflow": {"kind": "realtime", "name": "test_realtime", "version": "0.1.0
↪"},
          "parameters": {}
        },
        {
          "pb_id": "pb-mvp01-20200325-00003",
          "workflow": {"kind": "batch", "name": "ical", "version": "0.1.0"},
          "parameters": {},
          "dependencies": [
            {"pb_id": "pb-mvp01-20200325-00001", "kind": ["visibilities"]}
          ]
        },
        {
          "pb_id": "pb-mvp01-20200325-00004",
          "workflow": {"kind": "batch", "name": "dpreb", "version": "0.1.0"},
          "parameters": {},
          "dependencies": [
            {"pb_id": "pb-mvp01-20200325-00003", "kind": ["calibration"]}
```

(continues on next page)

(continued from previous page)

```
            ]
          }
        ]
      }
}
```

Example JSON response modelled by `AssignResourcesResponse` for MID:

```
{
  "dish": {
    "receptor_ids_allocated": ["0001", "0002"]
  }
}
```

Example JSON input modelled by `AssignResourcesRequest` for LOW:

```
{
  "interface": "https://schema.skao.int/ska-low-tmc-assignresources/2.0",
  "subarray_id": 1,
  "mccs": {
      "subarray_beam_ids": [1],
      "station_ids": [[1,2]],
      "channel_blocks": [3]
   }
}
```

5.3 release_resources.py

The `release_resources.py` module models the input JSON for a `CentralNode.ReleaseResources()` command.

Example ReleaseResourcesRequest JSON that requests specific dishes be released from a sub-array:

```
{
  "interface": "https://schema.skao.int/ska-tmc-releaseresources/2.0",
  "transaction_id": "txn-mvp01-20200325-00001",
  "subarray_id": 1,
  "receptor_ids": ["0001", "0002"]
}
```

Example JSON that requests all sub-array resources be released:

```
{
  "interface": "https://schema.skao.int/ska-tmc-releaseresources/2.0",
  "transaction_id": "txn-mvp01-20200325-00001",
  "subarray_id": 1,
  "release_all": true
}
```

Example JSON that requests all sub-array resources be released for LOW:

```
{
  "interface": "https://schema.skao.int/ska-low-tmc-releaseresources/2.0",
  "subarray_id": 1,
```

(continues on next page)

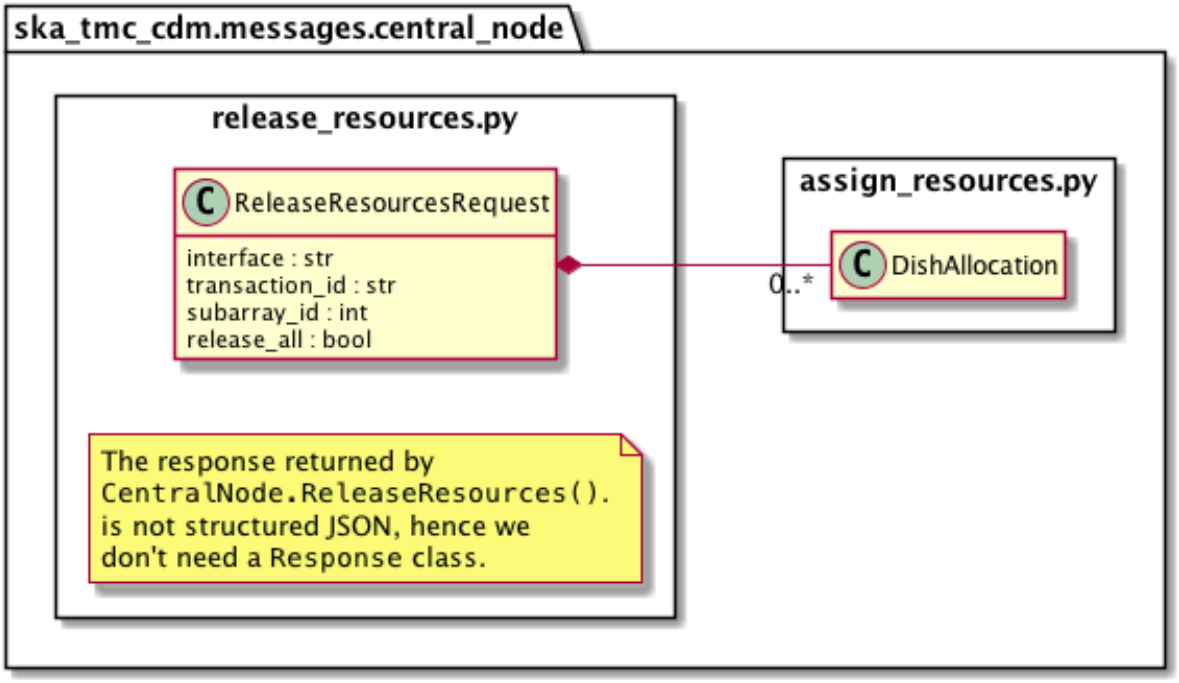


Fig. 2: release_resources.py object model

(continued from previous page)

```
  "release_all": true
}
```

CHAPTER 6

TMC SubArrayNode

6.1 Overview

Sub-array configuration and scan control is achieved via communication with a TMC SubArrayNode Tango device. The diagram below shows the packages and high-level object model used for telescope configuration and control.

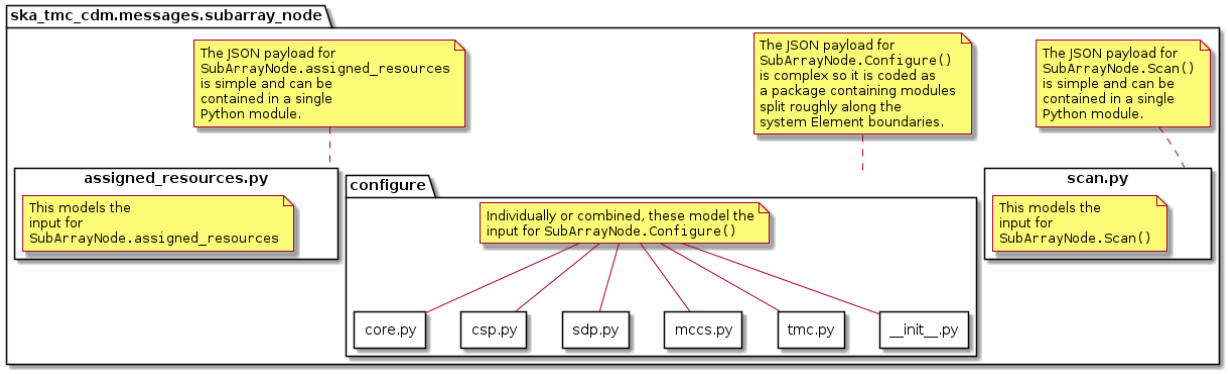


Fig. 1: High-level object model for communication with a TMC SubArrayNode device.

Classes in the *configure* package model the arguments for the `SubArrayNode.Configure()` command.

Classes in the *scan.py* module model the arguments for the `SubArrayNode.Scan()` command.

6.2 configure

The configuration JSON is complex, the module is split between several modules. The `configure` package contains five modules:

- __init__.py

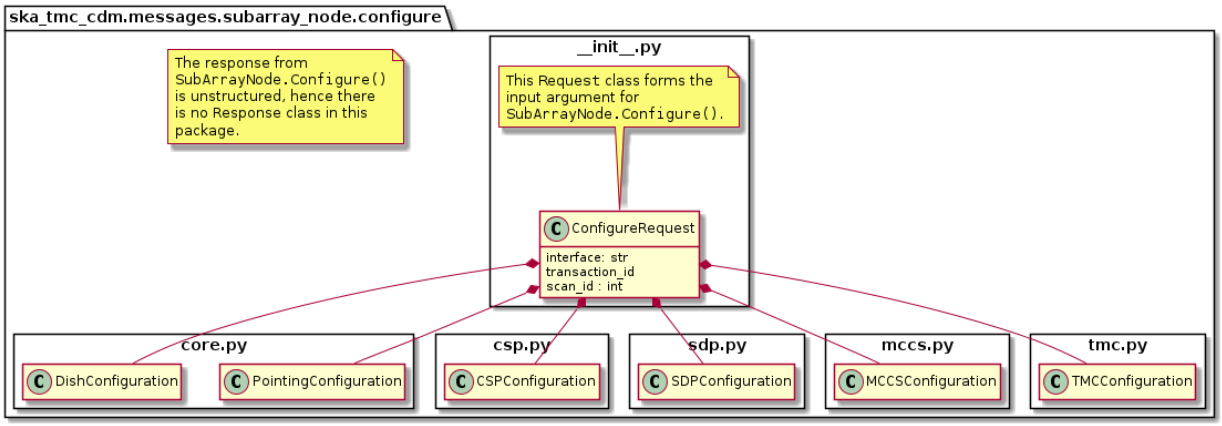


Fig. 2: High-level overview of the configure package

- *core.py*
- *tmc.py*
- *csp.py*
- *sdp.py*
- *mccs.py*

`__init__.py` references sub-modules in the main ConfigureRequest object, as illustrated in the diagram above.

In the context of a full JSON example object, __init__.py defines the a basic container object, while the sub-modules define the details.

```
# JSON modelled specifically by __init__.py
{
  "scanID": 12345,
  ...
}
```

6.2.1 core.py

The `core.py` module models receptor pointing and receiver band JSON elements. In the context of a full CDM JSON object, the elements this maps to are:

```
# JSON modelled specifically by core.py
{
  ...
  "pointing": {
    "target": {
      "reference_frame":"ICRS",
      "name": "NGC6251",
      "ra": 1.0,
      "dec": 1.0
    },
  },
  ...
  "dish": {
```

(continues on next page)

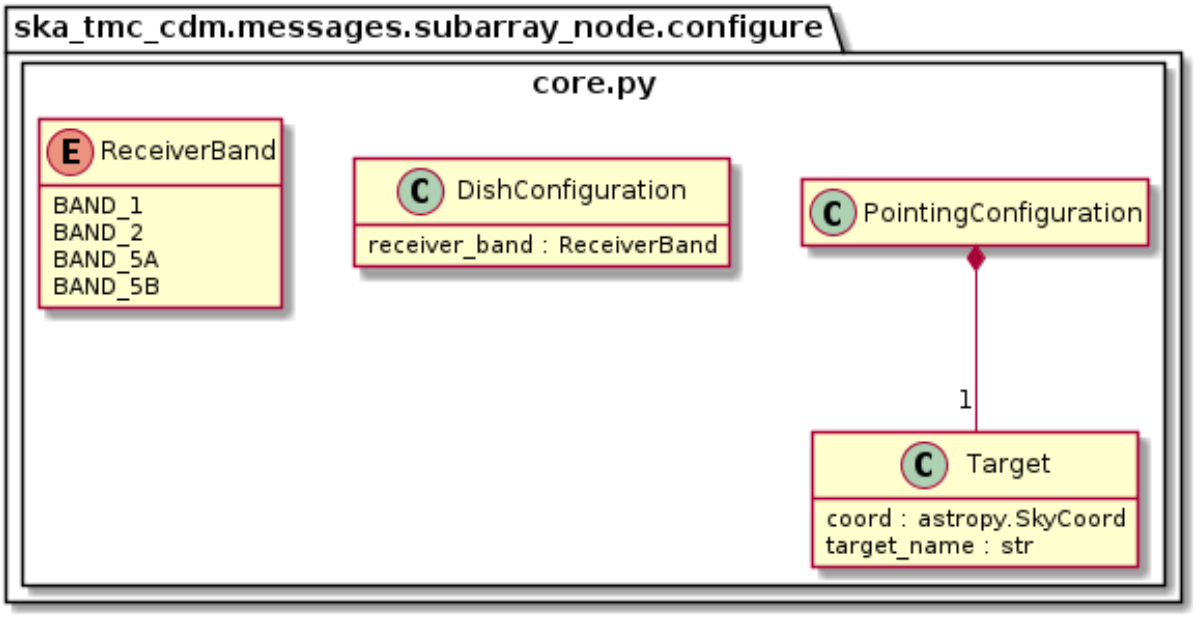


Fig. 3: core.py object model

(continued from previous page)

```
    "receiver_band": "1"
  }
  ....
}
```

6.2.2 tmc.py

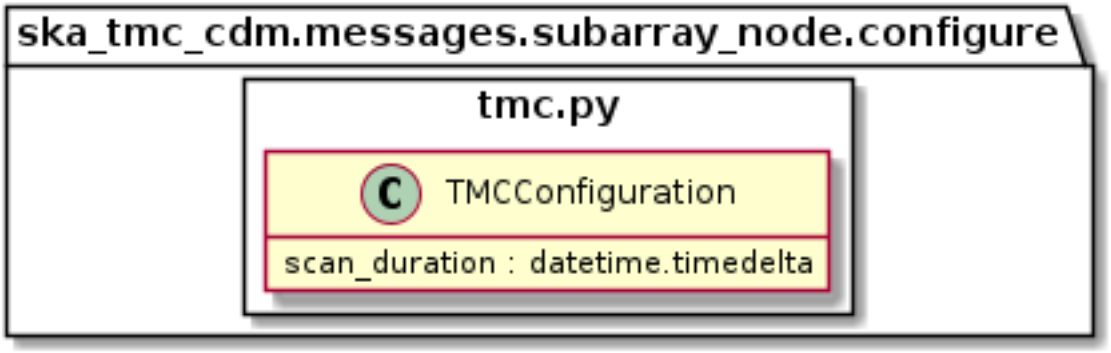


Fig. 4: tmc.py object model

The `tmc.py` module models TMC configuration JSON elements. Below is an example JSON command argument that this code can model.

```
# JSON modelled specifically by tmc.py
{
  "tmc": {
    "scan_duration": 10.0,
  }
}
```

6.2.3 csp.py

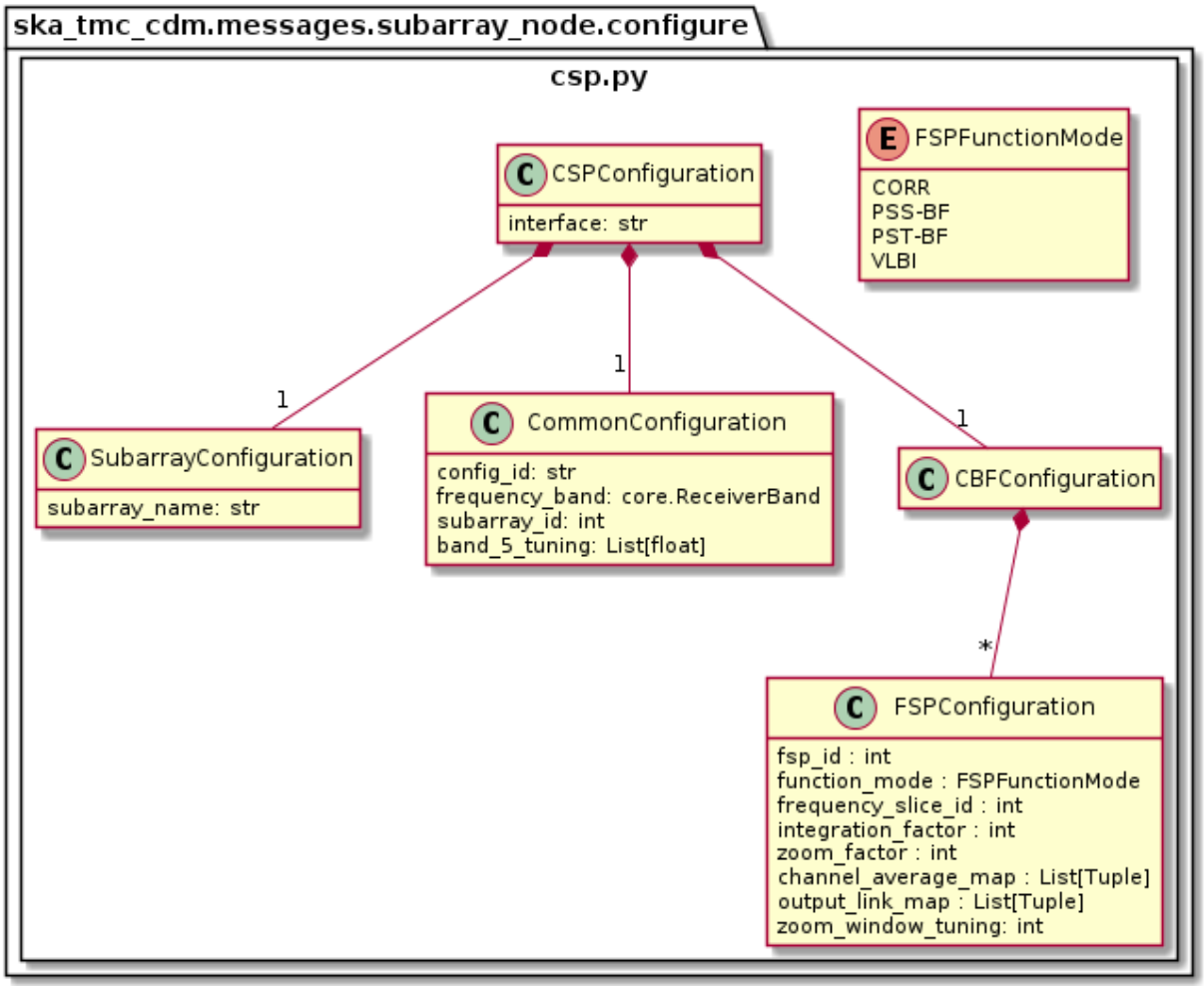


Fig. 5: csp.py object model

The `csp.py` module models CSP configuration JSON elements. In the context of a full CDM JSON object, the elements this maps to are:

```
# JSON modelled specifically by csp.py
{
  ...
  "csp": {
    "interface": "https://schema.skao.int/ska-csp-configure/2.0",
    "subarray": {
```

(continues on next page)

(continued from previous page)

```
      "subarray_name": "science period 23"
    },
    "common": {
      "config_id": "sbi-mvp01-20200325-00001-science_A",
      "frequency_band": "1",
      "subarray_id": 1
    },
    "cbf": {
      "fsp": [
        {
          "fsp_id": 1,
          "function_mode": "CORR",
          "frequency_slice_id": 1,
          "integration_factor": 10,
          "output_link_map": [
            [0,0],
            [200,1]
          ],
          "zoom_factor": 0,
          "channel_averaging_map": [
            [0, 2],
            [744, 0]
          ],
          "channel_offset": 0
        },
        {
          "fsp_id": 2,
          "function_mode": "CORR",
          "frequency_slice_id": 2,
          "integration_factor": 10,
          "zoom_factor": 1,
          "output_link_map": [
            [0,4],
           [200,5]
          ],
          "channel_averaging_map": [
            [0, 2],
            [744, 0]
          ],
          "channel_offset": 744,
          "zoom_window_tuning": 4700000
        }
      ]
    }
  },
  ...
}
```

6.2.4 sdp.py

The `sdp.py` module models SDHP configuration JSON elements. In the context of a full CDM JSON object, the elements this maps to are:

```
# JSON modelled specifically by sdp.py
{
```

(continues on next page)

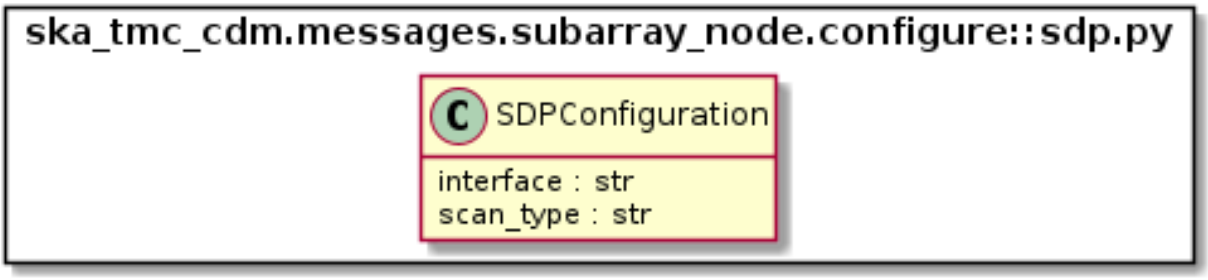


Fig. 6: sdp.py object model

(continued from previous page)

```
  ...
  "sdp": {
    "scan_type": "science_A"
  },
  ...
}
```

6.2.5 mccs.py

The `mccs.py` module models MCCS configuration JSON elements. In the context of a full CDM JSON object, the elements this maps to are:

```
# JSON modelled specifically by mccs.py
{
  "mccs": {
      "stations": [
        {
          "station_id": 1
        },
        {
          "station_id": 2
        }
      ],
      "subarray_beams": [
        {
          "subarray_beam_id": 1,
          "station_ids": [1, 2],
          "update_rate": 0,
          "channels": [
            [0, 8, 1, 1],
            [8, 8, 2, 1],
            [24, 16, 2, 1]
          ],
          "antenna_weights": [1, 1, 1],
          "phase_centre": [0, 0],
          "target": {
            "system": "HORIZON",
            "name": "DriftScan",
            "az": 180,
            "el": 45
```

(continues on next page)

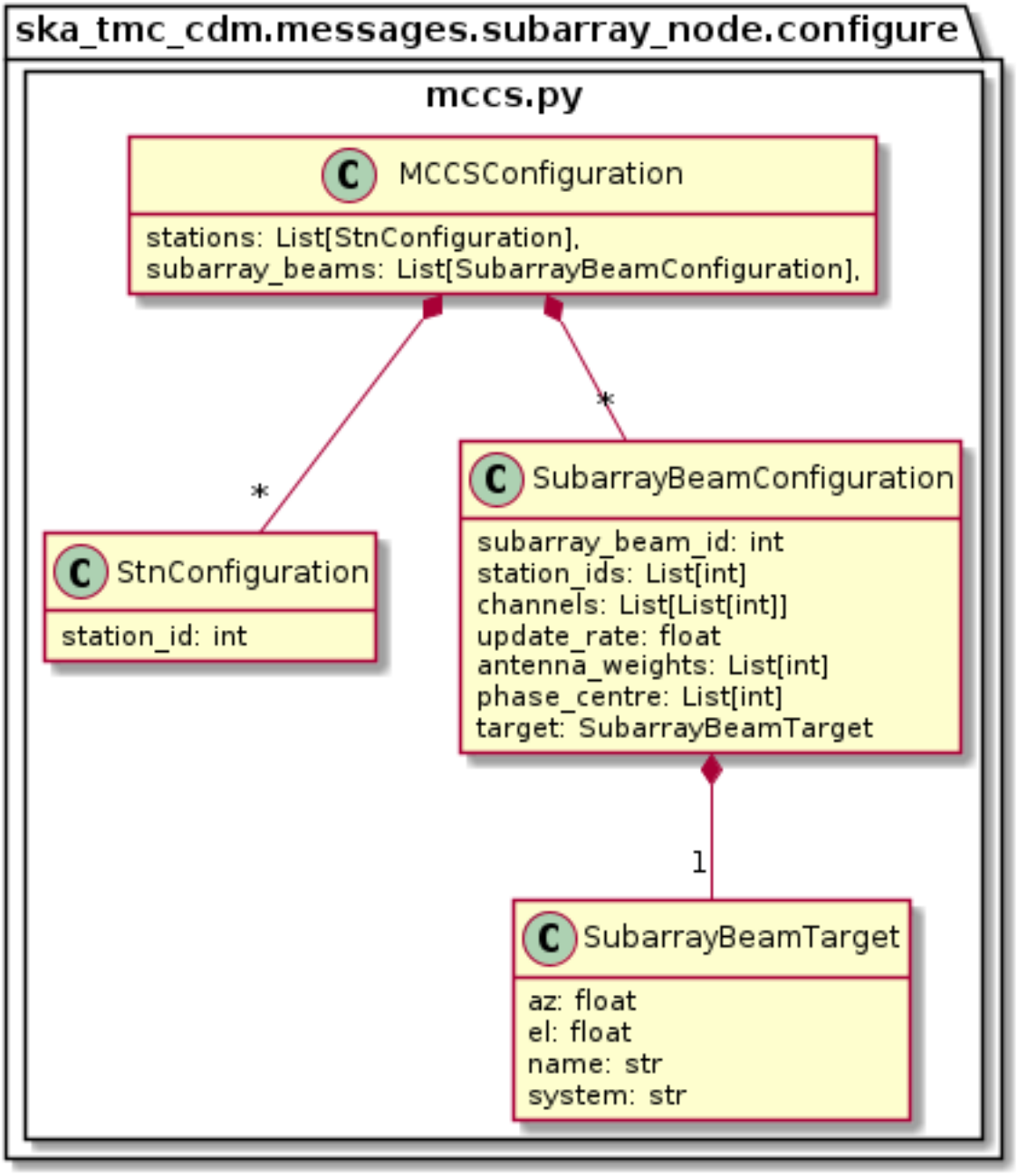


Fig. 7: mccs.py object model

(continued from previous page)

```
            }
          }
        ]
    }
}
```

6.3 assigned_resources.py

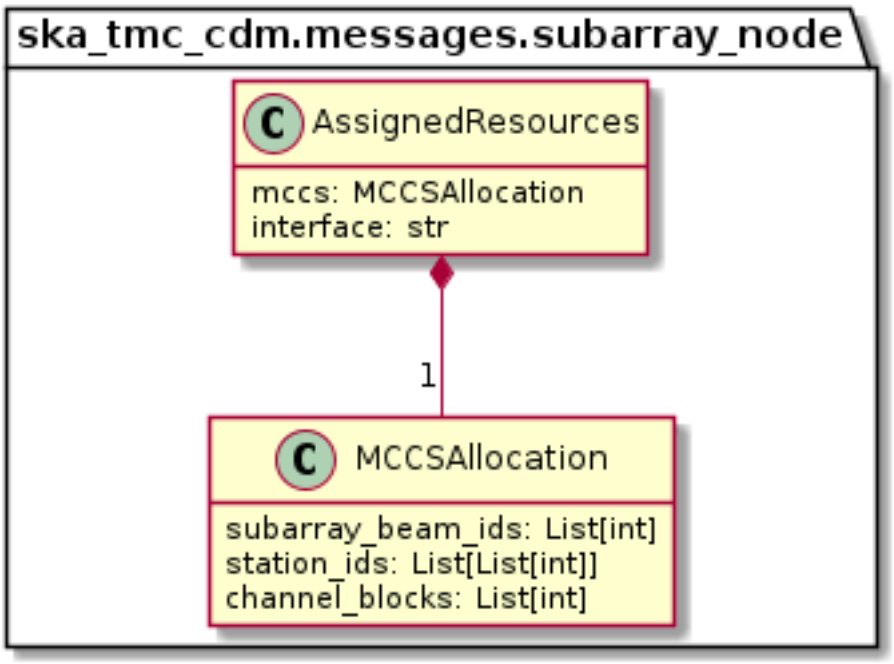


Fig. 8: assigned_resources.py object model

The `assigned_resources.py` module describes which resources have been assigned to the sub-array.

Examples below depict a populated sub-array and an empty one:

```
{
    "interface": "https://schema.skao.int/ska-low-tmc-assignedresources/2.0",
    "mccs": {
        "subarray_beam_ids": [1],
        "station_ids": [[1,2]],
        "channel_blocks": [3]
    }
}
```

```
{
    "interface": "https://schema.skao.int/ska-low-tmc-assignedresources/2.0",
    "mccs": {
        "subarray_beam_ids": [],
        "station_ids": [],
```

(continues on next page)

(continued from previous page)

```
        "channel_blocks": []
    }
}
```

6.4 scan.py

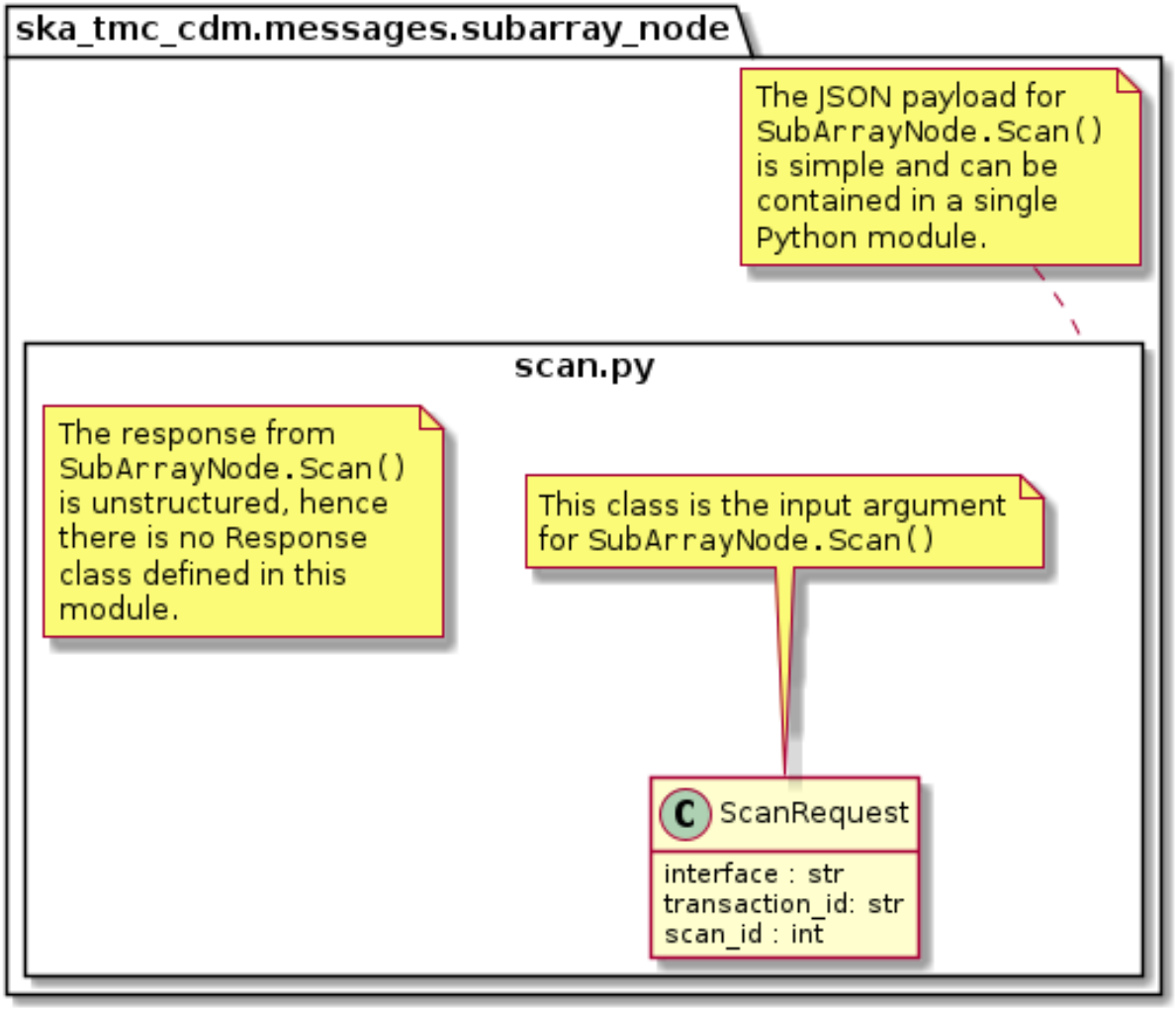


Fig. 9: scan.py object model

The `scan.py` module models the argument for the `SubArrayNode.scan()` command. Below is an example JSON command argument that this code can model.

```
{
  "interface": "https://schema.skao.int/ska-tmc-scan/2.0",
  "transaction_id": "txn-12345",
  "scan_id": 2
}
```

6.5 Example configuration JSON for MID

```
{
  "pointing": {
    "target": {
      "reference_frame":"ICRS",
      "name": "NGC1068",
      "ra": 0.70984,
      "dec": 0.000233
    },
  },
  "dish": {
    "receiver_band": "1"
  },
  "csp": {
    "interface": "https://schema.skao.int/ska-csp-configure/2.0",
    "subarray": {
      "subarray_name": "science period 23"
    },
    "common": {
      "id": "sbi-mvp01-20200325-00001-science_A",
      "frequencyBand": "1",
      "subarray_id": 1
    },
    "cbf": {
      "fsp": [
        {
          "fsp_id": 1,
          "function_mode": "CORR",
          "frequency_slice_id": 1,
          "integration_factor": 10,
          "output_link_map": [[0,0], [200,1]],
          "zoom_factor": 0,
          "channel_averaging_map": [[0, 2], [744, 0]],
          "channel_offset": 0
        },
        {
          "fsp_id": 2,
          "function_mode": "CORR",
          "frequency_slice_id": 2,
          "integration_factor": 10,
          "zoom_factor": 1,
          "output_link_map": [[0,4], [200,5]],
          "channel_averaging_map": [[0, 2], [744, 0]],
          "channel_offset": 744,
          "zoom_window_tuning": 4700000
        }
      ]
    }
  },
  "sdp": {
    "interface": "https://schema.skao.int/ska-sdp-configure/0.3",
    "scan_type": "science_A"
  },
  "tmc": {
    "scan_duration": 10.0,
  }
```

(continues on next page)

(continued from previous page)

```
}
```

6.6 Example configuration JSON for LOW

```
{
  "interface": "https://schema.skao.int/ska-low-tmc-configure/2.0",
  "mccs": {
    "stations":[
      {
        "station_id": 1
      },
      {
        "station_id":2
      }
    ],
    "subarray_beams": [
      {
        "subarray_beam_id": 1,
        "station_ids": [1, 2],
        "update_rate": 0.0,
        "channels": [
          [0,8,1,1],
          [8,8,2,1],
          [24,16,2,1]
        ],
        "antenna_weights": [1.0, 1.0, 1.0],
        "phase_centre": [0.0, 0.0],
        "target": {
          "system": "HORIZON",
          "name": "DriftScan",
          "az": 180.0,
          "el": 45.0
        }
      }
    ]
  },
  "tmc": {
    "scan_duration": 10.0
  }
}
```

CHAPTER 7

MCCSSubarray

7.1 Overview

MCCS configuration and scan control is achieved via communication with a MCCSSubarray Tango device. Additionally, the MCCSSubarray device presents a device attribute that lists the resources allocated to that subarray.

The diagram below shows the packages and high-level object model used for communication with an MCCSSubarray device.

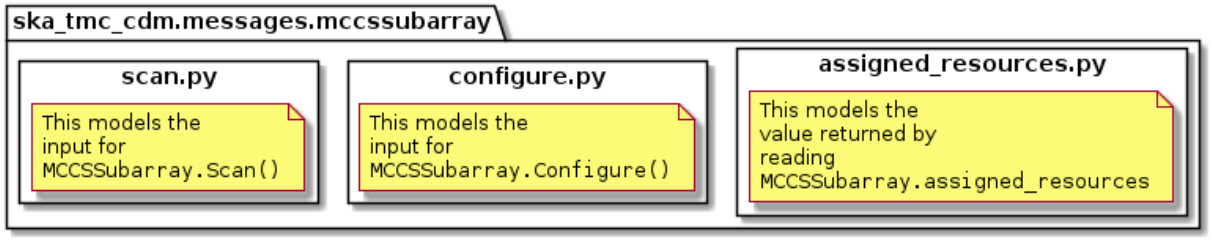


Fig. 1: High-level object model for communication with a MCCSSubarray device.

Classes in the *assigned_resources.py* module model the resource allocation status JSON string returned by the `MCCSSubarray.assigned_resources` attribute.

Classes in the *configure.py* module model the arguments for the `MCCSSubarray.Configure()` command.

Classes in the *scan.py* module model the arguments for the `MCCSSubarray.Scan()` command.

7.2 assigned_resources.py

The `assigned_resources.py` module models the the JSON returned by reading the `MCCSSubarray.assigned_resources` attribute.

Example JSON returned by `MCCSSubarray.assigned_resources`:

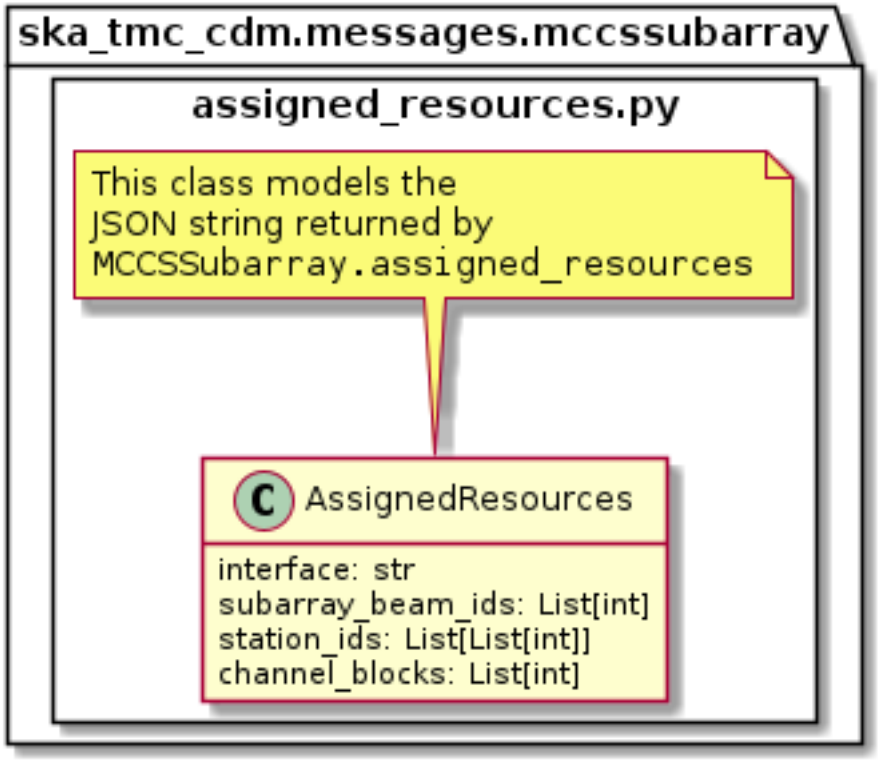


Fig. 2: High-level overview of the assigned_resources module

```
{
  "interface": "https://schema.skao.int/ska-low-mccs-assignedresources/2.0",
  "subarray_beam_ids": [1],
  "station_ids": [[1,2]],
  "channel_blocks": [3]
}
```

7.3 configure.py

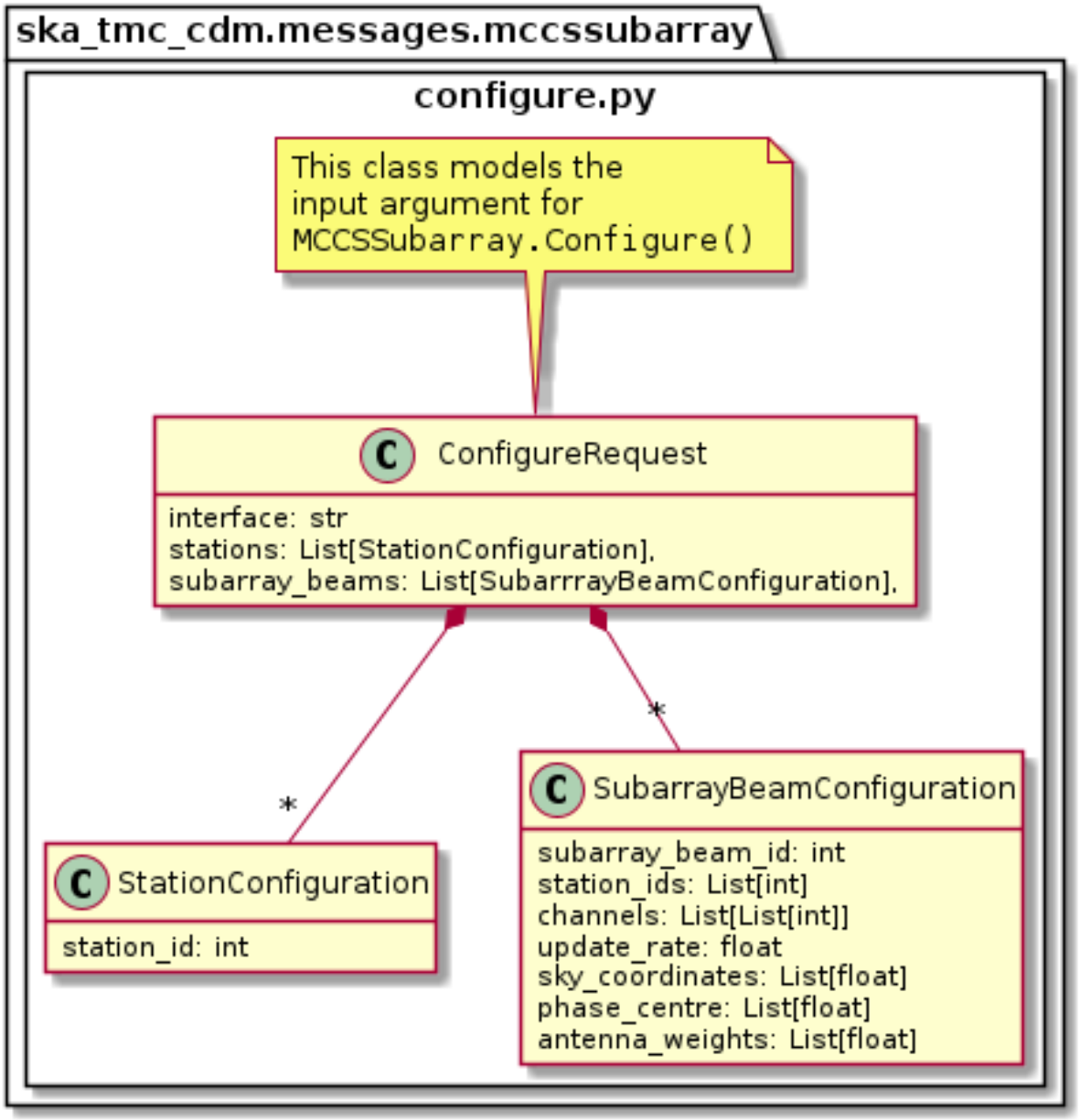


Fig. 3: High-level overview of the configure module

The `configure.py` module models the the JSON input for an `MCCSSubarray.configure()` command.

Example JSON input for an `MCCSSubarray.Configure` call:

```
{
  "interface": "https://schema.skao.int/ska-low-mccs-configure/2.0",
  "stations":[
    {
      "station_id": 1
    },
    {
      "station_id":2
    }
  ],
  "subarray_beams": [
    {
      "subarray_beam_id":1,
      "station_ids": [1, 2],
      "update_rate": 0.0,
      "channels": [
        [0,8,1,1],
        [8,8,2,1],
        [24,16,2,1]
      ],
      "sky_coordinates": [0.0, 180.0, 0.0, 45.0, 0.0],
      "antenna_weights": [1.0, 1.0, 1.0],
      "phase_centre": [0.0, 0.0]
    }
  ]
}
```

7.4 scan.py

The `scan.py` module models the argument for the `MCCSSubarray.scan()` command.

Example JSON input for an `MCCSSubarray.scan()` call:

```
{
  "interface": "https://schema.skao.int/ska-low-mccs-scan/2.0",
  "scan_id":1,
  "start_time": 0.0
}
```

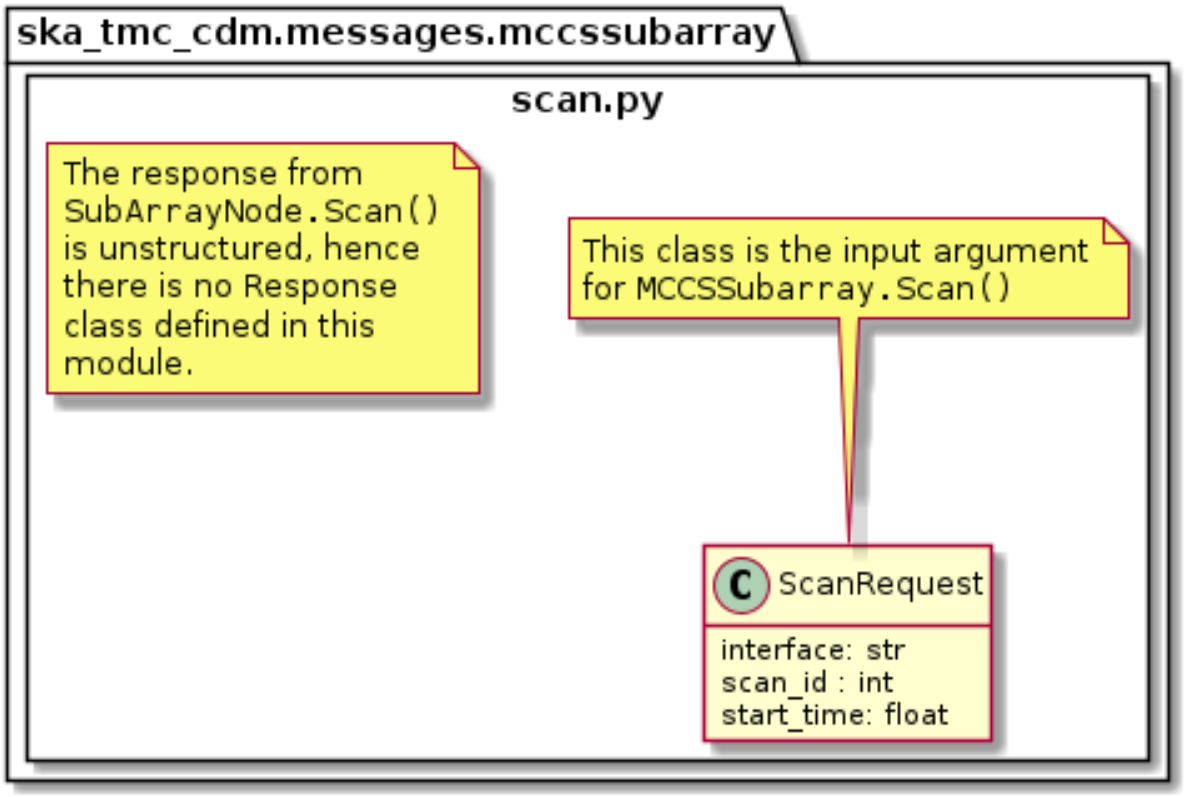


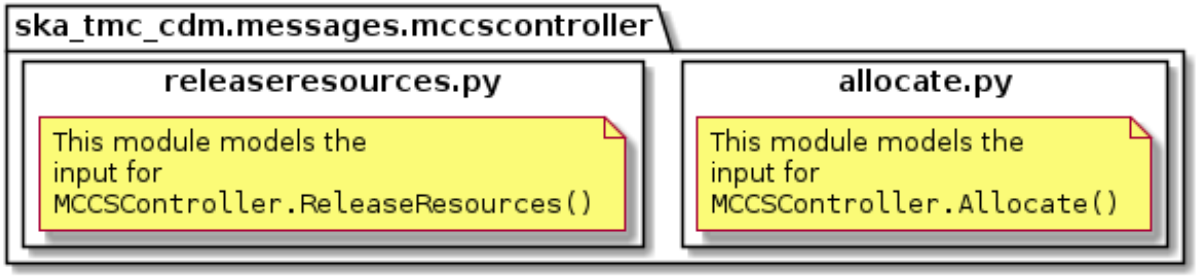
Fig. 4: scan.py object model

CHAPTER 8

MCCSController

8.1 Overview

MCCS resource allocation is achieved via communication with an MCCSController device. The `mccscontroller` package models the JSON input and for MCCSController commands. The contents of this package are shown in the figure below.



Classes in the *allocate.py* module model the arguments for the `MCCSController.Allocate()` command.

Classes in the *releaseresources.py* module model the arguments for the `MCCSController.ReleaseResources()` command.

8.2 allocate.py

The `allocate.py` module models the the JSON input for an `MCCSController.Allocate()` command.

Example JSON input modelled by `MCCSController.Allocate`:

```
{
  "interface": "https://schema.skao.int/ska-low-mccs-assignresources/2.0",
```

(continues on next page)

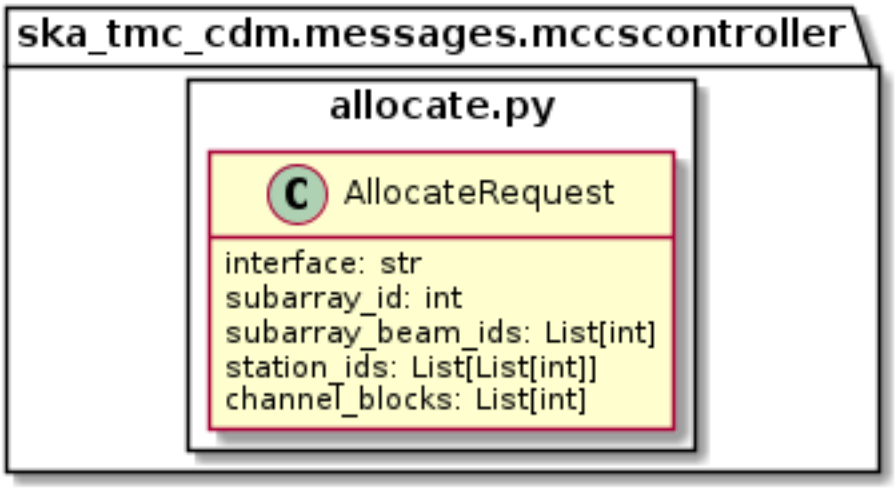


Fig. 1: allocate.py object model

(continued from previous page)

```
  "subarray_id": 1,
  "subarray_beam_ids": [1],
  "station_ids": [[1,2]],
  "channel_blocks": [3]
}
```

8.3 releaseresources.py

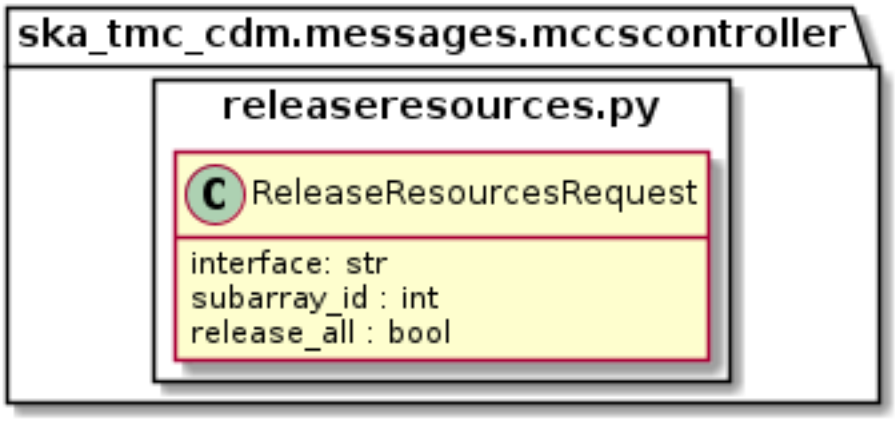


Fig. 2: releaseresources.py object model

The `releaseresources.py` module models the input JSON for a `MCCSController.ReleaseResources()` command.

Example ReleaseResourcesRequest JSON that requests all resources be released from sub-array #1:

```
{
  "interface": "https://schema.skao.int/ska-low-mccs-releaseresources/2.0",
  "subarray_id": 1,
  "release_all": true
}
```

CHAPTER 9

Using the CDM

To use this library in your project, create objects using the classes defined in `ska_tmc_cdm.messages` and convert them to/from JSON using `ska_tmc_cdm.schemas.CODEC`.

The Python snippet below is an example of constructing a JSON argument for a `CentralNode.ReleaseResources()` command. The resulting JSON can be sent to the device using a `DeviceProxy`.

```
# import the classes for ReleaseResources commands and CODEC for serialisation
from ska_tmc_cdm.messages import ReleaseResourcesRequest
from ska_tmc_cdm.schemas import CODEC

# create an object for a command that will release all resources on subarray #2
cmd_arg = ReleaseResourcesRequest(2, release_all=True)
# convert the argument to JSON, ready for use in a DeviceProxy call
as_json = CODEC.dumps(cmd_arg)
```

Below is an example of converting the JSON response from a `CentralNode.AssignResources()` command to Python objects. The example assumes you have the string response from the command call at hand.

```
# import the classes for ReleaseResources commands and CODEC for serialisation
from ska_tmc_cdm.messages import AssignResourcesResponse
from ska_tmc_cdm.schemas import CODEC

# assume that you have some JSON-formatted string returned by AssignResources()
json_response = ...
# convert the JSON to a Python object. This requires you to provide the class
# you want to convert to
unmarshalled = CODEC.loads(AssignResourcesResponse, json_response)
# This object can hold other objects, as defined by the schema. For example,
# the response for an AssignResources command includes the dish IDs of the
# dishes that were assigned to it. The schema converts this into a
# DishAllocation object we can inspect and manipulate
print(f'Dish IDs allocated: {unmarshalled.dish.receptor_ids}')
```

CHAPTER 10

API

10.1 ska_tmc_cdm.jsonschema

10.1.1 ska_tmc_cdm.jsonschema.json_schema

The JSON Schema module contains methods for fetching version-specific JSON schemas using interface uri and validating the structure of JSON against these schemas.

class JsonSchema

JSON Schema use for validating the structure of JSON data

static get_schema_by_uri(*uri: str*) → <sphinx.ext.autodoc.importer._MockObject object at 0x7fa228421050>

Retrieve JSON Schemas from remote server.

Parameters **uri** – Interface Version URI

Returns Interface schema

Raises SchemaNotFound if URI does not resolve to a schema

static validate_schema(*uri: str*, *instance: dict*, *strictness=None*) → None

Validate an instance dictionary under the given schema.

strictness can be set from 0-2. Values equal:

0: permissive warnings 1: permissive errors and strict warnings 2: strict errors

Parameters

- **uri** – The schema to validate with
- **instance** – The instance to validate
- **strictness** – strictness level

Returns None, in case of valid data otherwise, it raises an exception.

10.2 ska_tmc_cdm.messages

The ska_tmc_cdm.messages package contains modules that maps Tango structured arguments to Python.

10.2.1 ska_tmc_cdm.messages.central_node

The ska_tmc_cdm.messages.central_node package holds modules that translate TMC Central Node requests and responses to and from Python.

10.2.2 ska_tmc_cdm.messages.central_node.assign_resources

The messages module provides simple Python representations of the structured request and response for the TMC CentralNode.AssignResources command.

class AssignResourcesRequest(*subarray_id: int = None*, *dish_allocation: ska_tmc_cdm.messages.central_node.common.DishAllocation = None*, *sdp_config: ska_tmc_cdm.messages.central_node.sdp.SDPConfiguration = None*, *mccs: ska_tmc_cdm.messages.central_node.mccs.MCCSAllocate = None*, *interface: str = None*, *transaction_id: str = None*)

AssignResourcesRequest is a Python representation of the structured argument for a TMC CentralNode.AssignResourcesRequest request.

classmethod from_dish(*subarray_id: int*, *dish_allocation: ska_tmc_cdm.messages.central_node.common.DishAllocation*, *sdp_config: ska_tmc_cdm.messages.central_node.sdp.SDPConfiguration = None*, *interface: str = None*, *transaction_id: str = None*)

Create a new AssignResourcesRequest object.

Parameters

- **subarray_id** – the numeric SubArray ID (1..16)
- **dish_allocation** – object holding the DISH resource allocation for this request.
- **sdp_config** – sdp configuration

Returns AssignResourcesRequest object

classmethod from_mccs(*subarray_id: int*, *mccs: ska_tmc_cdm.messages.central_node.mccs.MCCSAllocate*, *sdp_config: ska_tmc_cdm.messages.central_node.sdp.SDPConfiguration = None*, *interface: str = None*, *transaction_id: str = None*)

Create a new AssignResourcesRequest object.

Parameters

- **subarray_id** – the numeric SubArray ID (1..16)
- **mccs** – MCCS subarray allocation
- **sdp_config** – SDP configuration
- **interface** – url string to determine JsonSchema version

Returns AssignResourcesRequest object

class AssignResourcesResponse(*dish_allocation: ska_tmc_cdm.messages.central_node.common.DishAllocation = None*)

AssignResourcesResponse is a Python representation of the structured response from a TMC CentralNode.AssignResources request.

10.2.3 ska_tmc_cdm.messages.central_node.release_resources

The release_resources module provides simple Python representations of the structured request and response for a TMC CentralNode.ReleaseResources call.

class ReleaseResourcesRequest(**_, interface: str = None, transaction_id: str = None, subarray_id: int = None, release_all: bool = False, dish_allocation: Optional[ska_tmc_cdm.messages.central_node.common.DishAllocation] = None*)

ReleaseResourcesRequest is a Python representation of the structured request for a TMC CentralNode.ReleaseResources call.

10.2.4 ska_tmc_cdm.messages.central_node.common

The messages module provides simple Python representations of the structured request and response for the TMC CentralNode.AssignResources command.

class DishAllocation(*receptor_ids: Optional[List[str]] = None*)

DishAllocation represents the DISH allocation part of an AssignResources request and response.

10.2.5 ska_tmc_cdm.messages.central_node.sdp

The messages module provides simple Python representations of the structured request and response for the TMC CentralNode.AssignResources command.

class SDPWorkflow(*name: str, kind: str, version: str*)

Class to hold SDPWorkflows for ProcessingBlock

class SDPConfiguration(*eb_id: str, max_length: float, scan_types: List[ska_tmc_cdm.messages.central_node.sdp.ScanType], processing_blocks: List[ska_tmc_cdm.messages.central_node.sdp.ProcessingBlockConfiguration], interface: str = None*)

Class to hold SDPConfiguration

class ProcessingBlockConfiguration(*pb_id: str, workflow: ska_tmc_cdm.messages.central_node.sdp.SDPWorkflow, parameters: Dict[KT, VT], dependencies: List[ska_tmc_cdm.messages.central_node.sdp.PbDependency] = None*)

Class to hold ProcessingBlock configuration

class PbDependency(*pb_id: str, pb_type: List[str]*)

Class to hold Dependencies for ProcessingBlock

class ScanType(*scan_type_id, reference_frame: str, ra: str, dec: str, channels: List[ska_tmc_cdm.messages.central_node.sdp.Channel]*)

Class to hold ScanType configuration

class Channel(*count: int, start: int, stride: int, freq_min: float, freq_max: float, link_map: List[List[T]]*)

Class to hold Channels for ScanType

10.2.6 ska_tmc_cdm.messages.central_node.mccs

class MCCSAllocate(*station_ids: Sequence[Sequence[int]], channel_blocks: Sequence[int], subarray_beam_ids: Sequence[int]*)

MCCSAllocate is a Python representation of the structured argument for a TMC CentralNode.AssignResourcesRequest.

10.2.7 ska_tmc_cdm.messages.mccscontroller.allocate

The allocate module defines a Python object model for the structured JSON given in an MCCSController.Allocate call.

class AllocateRequest(**, interface: Optional[str] = 'https://schema.skao.int/ska-low-mccs-assignresources/2.0', subarray_id: int, subarray_beam_ids: List[int] = None, station_ids: List[List[int]] = None, channel_blocks: List[int] = None*)

AssignResourcesRequest is the object representation of the JSON argument for an MCCSController.Allocate command.

10.2.8 ska_tmc_cdm.messages.mccscontroller.releaseresources

The allocate module defines a Python object model for the structured JSON that forms the argument for an MCCSController.ReleaseResources call.

class ReleaseResourcesRequest(**, interface: Optional[str] = 'https://schema.skao.int/ska-low-mccs-releaseresources/2.0', subarray_id: int, release_all: bool*)

ReleaseResourcesRequest is the object representation of the JSON argument for an MCCSController.ReleaseResources command.

10.2.9 ska_tmc_cdm.messages.mccssubarray.assigned_resources

class AssignedResources(**, interface: Optional[str] = 'https://schema.skao.int/ska-low-mccs-assignedresources/2.0', subarray_beam_ids: List[int] = None, station_ids: List[List[int]] = None, channel_blocks: List[int] = None*)

AssignedResources is the object representation of the JSON returned by the MCCSSubarray.assigned_resources attribute.

10.2.10 ska_tmc_cdm.messages.mccssubarray.configure

The mccssubarray.configure module contains a Python object model for the various structured bits of JSON given in an MCCSSubarray.Configure call.

class ConfigureRequest(**, interface: Optional[str] = 'https://schema.skao.int/ska-low-mccs-configure/2.0', stations: List[ska_tmc_cdm.messages.mccssubarray.configure.StationConfiguration], subarray_beams: List[ska_tmc_cdm.messages.mccssubarray.configure.SubarrayBeamConfiguration*

Class to hold all subarray configuration.

class StationConfiguration(*station_id: int*)

A class to hold station configuration

class SubarrayBeamConfiguration(**, subarray_beam_id: int, station_ids: List[int], update_rate: float, channels: List[List[int]], sky_coordinates: List[float], antenna_weights: List[float], phase_centre: List[float]*)

A class to hold subarray beam configuration attributes

10.2.11 ska_tmc_cdm.messages.mccssubarray.scan

The scan module defines Python object representations of the structured request for an MCCSSubarray.Scan command.

class ScanRequest(*, *interface: Optional[str] = 'https://schema.skao.int/ska-low-mccs-scan/2.0'*, *scan_id: int*, *start_time: float*)
ScanRequest represents the request argument for MCCSSubarray.Scan call.

10.2.12 ska_tmc_cdm.messages.subarray_node

The ska_tmc_cdm.messages.subarray_node package holds modules containing classes that represent arguments, requests, and responses for TMC SubArrayNode devices.

10.2.13 ska_tmc_cdm.messages.subarray_node.assigned_resources

TMC Assigned Resources

class MCCSAllocation(*subarray_beam_ids: List[int]*, *station_ids: List[List[int]]*, *channel_blocks: List[int]*)
MCCSAllocation is a Python representation of the structured JSON representing the resources assigned to an MCCS subarray.

is_empty()
Determine that the current MCCSAllocation instance is empty (none of the attribute Lists are populated)

class AssignedResources(*, *interface: Optional[str] = 'https://schema.skao.int/ska-low-tmc-assignedresources/2.0'*, *mccs: ska_tmc_cdm.messages.subarray_node.assigned_resources.MCCSAl*
AssignedResources models the structured JSON returned when the MCCSSubarray.assigned_resources Tango attribute is read.

is_empty() → bool
Determine that the current MCCSAllocation instance is empty (none of the attribute Lists are populated)

10.2.14 ska_tmc_cdm.messages.subarray_node.configure

The configure package contains modules that define Python classes for all of the permissible arguments for a SubArrayNode.configure() call.

class ConfigureRequest(*pointing: ska_tmc_cdm.messages.subarray_node.configure.core.PointingConfiguration = None*, *dish: ska_tmc_cdm.messages.subarray_node.configure.core.DishConfiguration = None*, *sdp: ska_tmc_cdm.messages.subarray_node.configure.sdp.SDPConfiguration = None*, *csp: ska_tmc_cdm.messages.subarray_node.configure.csp.CSPConfiguration = None*, *mccs: ska_tmc_cdm.messages.subarray_node.configure.mccs.MCCSConfiguration = None*, *tmc: ska_tmc_cdm.messages.subarray_node.configure.tmc.TMCConfiguration = None*, *interface: Optional[str] = 'https://schema.skao.int/ska-tmc-configure/2.0'*, *transaction_id: Optional[str] = None*)
ConfigureRequest encapsulates the arguments required for the TMC SubArrayNode.Configure() command.

10.2.15 ska_tmc_cdm.messages.subarray_node.configure.core

The configure.common module contains simple Python representations of the structured request and response for the TMC SubArrayNode.Configure command.

As configurations become more complex, they may be rehomed in a submodule of this package.

class PointingConfiguration(*target: ska_tmc_cdm.messages.subarray_node.configure.core.Target*)
PointingConfiguration specifies where the subarray receptors are going to point.

class Target(*ra*, *dec*, *target_name=''*, *reference_frame='icrs'*, *unit=('hourangle'*, *'deg')*)
Target encapsulates source coordinates and source metadata.

The SubArrayNode ICD specifies that RA and Dec must be provided, hence non-ra/dec frames such as galactic are not supported.

class ReceiverBand
ReceiverBand is an enumeration of SKA MID receiver bands.

class DishConfiguration(*receiver_band: ska_tmc_cdm.messages.subarray_node.configure.core.ReceiverBand*)
DishConfiguration specifies how SKA MID dishes in a sub-array should be configured. At the moment, this is limited to setting the receiver band.

10.2.16 ska_tmc_cdm.messages.subarray_node.configure.csp

The configure.csp module contains Python classes that represent the various aspects of CSP configuration that may be specified in a SubArrayNode.configure command.

class CSPConfiguration(*interface: str = None*, *subarray_config: ska_tmc_cdm.messages.subarray_node.configure.csp.SubarrayConfiguration = None*, *common_config: ska_tmc_cdm.messages.subarray_node.configure.csp.CommonConfiguratio = None*, *cbf_config: ska_tmc_cdm.messages.subarray_node.configure.csp.CBFConfiguration = None*, *pst_config: ska_tmc_cdm.messages.subarray_node.configure.csp.PSTConfiguration = None*, *pss_config: ska_tmc_cdm.messages.subarray_node.configure.csp.PSSConfiguration = None*)
Class to hold all CSP configuration.

class FSPConfiguration(*fsp_id: int*, *function_mode: ska_tmc_cdm.messages.subarray_node.configure.csp.FSPFunctionMode*, *frequency_slice_id: int*, *integration_factor: int*, *zoom_factor: int*, *channel_averaging_map: List[Tuple] = None*, *output_link_map: List[Tuple] = None*, *channel_offset: int = None*, *zoom_window_tuning: int = None*)
FSPConfiguration defines the configuration for a CSP Frequency Slice Processor.

class FSPFunctionMode
FSPFunctionMode is an enumeration of the available FSP modes.

class CBFConfiguration(*fsp_configs: List[ska_tmc_cdm.messages.subarray_node.configure.csp.FSPConfiguration]*, *vlbi_config: ska_tmc_cdm.messages.subarray_node.configure.csp.VLBIConfiguration = None*)
Class to hold all FSP and VLBI configurations.

class SubarrayConfiguration(*subarray_name: str*)
Class to hold the parameters relevant only for the current sub-array device.

class CommonConfiguration(*config_id: str*, *frequency_band: ska_tmc_cdm.messages.subarray_node.configure.core.ReceiverB subarray_id: int = None*, *band_5_tuning: Optional[List[float]] = None*)
Class to hold the CSP sub-elements.

10.2.17 ska_tmc_cdm.messages.subarray_node.configure.sdp

The configure.sdp module contains Python classes that represent the various aspects of SDP configuration that may be specified in a SubArrayNode.configure command.

class SDPConfiguration(*, *interface: Optional[str] = 'https://schema.skao.int/ska-sdp-configure/0.3'*, *scan_type: str*)
Message class to hold SDP configuration aspect of a TMC SubArrayNode.Configure call.

10.2.18 ska_tmc_cdm.messages.subarray_node.configure.mccs

The configure.mccs module contains Python classes that represent the various aspects of MCCS configuration that may be specified in a SubArray.configure command.

class MCCSConfiguration(*_, *station_configs: List[ska_tmc_cdm.messages.subarray_node.configure.mccs.StnConfiguration], subarray_beam_configs: List[ska_tmc_cdm.messages.subarray_node.configure.mccs.SubarrayBea*
Class to hold all subarray configuration.

class StnConfiguration(*station_id: int*)
A class to hold station configuration configuration

class SubarrayBeamConfiguration(*subarray_beam_id: int, station_ids: List[int], channels: List[List[int]], update_rate: float, target: ska_tmc_cdm.messages.subarray_node.configure.mccs.SubarrayBeamTarget, antenna_weights: List[float], phase_centre: List[float]*)
A class to hold subarray_beam configuration attributes

class SubarrayBeamTarget(*az: float*, *el: float*, *target_name: str*, *reference_frame: str*)
Target encapsulates source coordinates and source metadata.

The SubArrayNode ICD specifies that az and el must be provided

10.2.19 ska_tmc_cdm.messages.subarray_node.configure.tmc

Configuration specific to TMC. scan_duration (in seconds) is the duration to be used for all scan commands following this configuration.

class TMCConfiguration(*scan_duration: datetime.timedelta*)
Class to hold TMC configuration

10.2.20 ska_tmc_cdm.messages.subarray_node.scan

The scan module defines simple Python representations of the structured request for a TMC SubArrayNode.Scan command.

class ScanRequest(*, *interface: Optional[str] = 'https://schema.skao.int/ska-tmc-scan/2.0'*, *transaction_id: Optional[str] = None*, *scan_id: int*)
ScanRequest represents the JSON for a SubArrayNode.scan call.

10.3 ska_tmc_cdm.schemas

The schemas for the SKA Configuration Data Model (CDM).

10.3.1 ska_tmc_cdm.schemas.central_node

The schemas.central_node package contains Marshmallow schemas that convert JSON to/from the Python classes contained in ska_tmc_cdm.messages.central_node.

class AssignResourcesRequestSchema(**args*, ***kwargs*)
Marshmallow schema for the AssignResourcesRequest class.

class Meta
marshmallow directives for AssignResourcesRequestSchema.

create_request(*data*, **_)
Convert parsed JSON back into an AssignResources request object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AssignResources object populated from data

validate_on_dump(*data*, **_)
Validating the structure of JSON against schemas and Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

class AssignResourcesResponseSchema(**args*, ***kwargs*)
Marshmallow schema for the AssignResourcesResponse class.

class Meta
Marshmallow directives for AssignResourcesResponseSchema.

create_response(*data*, **_)
Convert parsed JSON from an AssignResources response back into an AssignResourcesResponse object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AssignResourcesResponse object populated from data

class ReleaseResourcesRequestSchema(**args*, ***kwargs*)
Marshmallow schema for the ReleaseResourcesRequest class.

class Meta
Marshmallow directives for ReleaseResourcesRequestSchema.

create_request(*data*, **_)
Convert parsed JSON from an ReleaseResources request back into an ReleaseResourcesRequest object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ReleaseResourcesRequest object populated from data

filter_args(*data*, **_)
Filter Marshmallow's JSON based on the value of release_all.

If release_all is True, other resource definitions should be stripped from the request. If release_all for MID set to False, the 'release_all' key itself should be stripped. If release_all_low for LOW set to False, the 'release_all_low' key itself should be stripped.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for request submission

class DishAllocationSchema(**args*, ***kwargs*)

Marshmallow schema for the DishAllocation class.

create(*data*, **_)

Convert parsed JSON back into a DishAllocation object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns DishAllocation object populated from data

class DishAllocationResponseSchema(**args*, ***kwargs*)

Marshmallow schema for the DishAllocation class when received in the response.

create(*data*, **_)

Convert parsed JSON from an AssignResources response back into a DishAllocation object.

This 'duplicate' schema is required as the DishAllocation is found under a different JSON key in the response as compared to the request.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns DishAllocation object populated from data

class SDPConfigurationSchema(**args*, ***kwargs*)

Marsmallow class for the SDPConfiguration class

create_sdp_config(*data*, **_)

Convert parsed JSON back into a SDPConfiguration object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SDPConfiguration object populated from data

filter_nulls(*data*, **_)

Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for PB configuration

class MCCSAllocateSchema(**args*, ***kwargs*)
Marshmallow schema for the MCCSAllocate class.

create_mccs_allocate(*data*, ***_*)
Convert parsed JSON back into a MCCSAllocate object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns MCCSAllocate object populated from data

10.3.2 ska_tmc_cdm.schemas.central_node.assign_resources

The schemas.central_node module defines Marshmallow schemas that map TMC Central Node message classes to/from a JSON representation.

class AssignResourcesRequestSchema(**args*, ***kwargs*)
Marshmallow schema for the AssignResourcesRequest class.

class Meta
marshmallow directives for AssignResourcesRequestSchema.

create_request(*data*, ***_*)
Convert parsed JSON back into an AssignResources request object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AssignResources object populated from data

validate_on_dump(*data*, ***_*)
Validating the structure of JSON against schemas and Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

class AssignResourcesResponseSchema(**args*, ***kwargs*)
Marshmallow schema for the AssignResourcesResponse class.

class Meta
Marshmallow directives for AssignResourcesResponseSchema.

create_response(*data*, ***_*)
Convert parsed JSON from an AssignResources response back into an AssignResourcesResponse object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AssignResourcesResponse object populated from data

10.3.3 ska_tmc_cdm.schemas.central_node.common

The schemas.central_node module defines Marshmallow schemas that map TMC Central Node message classes to/from a JSON representation.

class DishAllocationSchema(**args*, ***kwargs*)
Marshmallow schema for the DishAllocation class.

create(*data*, **_*)
Convert parsed JSON back into a DishAllocation object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns DishAllocation object populated from data

class DishAllocationResponseSchema(**args*, ***kwargs*)
Marshmallow schema for the DishAllocation class when received in the response.

create(*data*, **_*)
Convert parsed JSON from an AssignResources response back into a DishAllocation object.

This 'duplicate' schema is required as the DishAllocation is found under a different JSON key in the response as compared to the request.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns DishAllocation object populated from data

10.3.4 ska_tmc_cdm.schemas.central_node.sdp

The schemas.central_node module defines Marshmallow schemas that map TMC Central Node message classes to/from a JSON representation.

class ScanTypeSchema(**args*, ***kwargs*)
Marshmallow schema for the ScanType class.

create_scan_type(*data*, **_*)
Convert parsed JSON back into a ScanType object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ScanType object populated from data

class SDPWorkflowSchema(**args*, ***kwargs*)
Represents the type of workflow being configured on the SDP

create_sdp_wf(*data*, **_*)
Convert parsed JSON back into a SDP Workflow object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values

- _ – kwargs passed by Marshmallow

Returns SDP Workflow object populated from data

class PbDependencySchema(**args*, ***kwargs*)
Marshmallow schema for the PbDepedency class.

create_pb_dependency(*data*, **_)
Convert parsed JSON back into a PbDependency object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns PbDependency object populated from data

class ChannelSchema(**args*, ***kwargs*)
Marshmallow schema for the SubBand class.

create_channel(*data*, **_)
Convert parsed JSON back into a Channel object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SubBand object populated from data

class ProcessingBlockSchema(**args*, ***kwargs*)
Marshmallow schema for the ProcessingBlock class.

create_processing_block_config(*data*, **_)
Convert parsed JSON back into a PB object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns PB object populated from data

filter_nulls(*data*, **_)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for PB configuration

class SDPConfigurationSchema(**args*, ***kwargs*)
Marsmallow class for the SDPConfiguration class

create_sdp_config(*data*, **_)
Convert parsed JSON back into a SDPConfiguration object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SDPConfiguration object populated from data

filter_nulls(*data*, **_)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for PB configuration

10.3.5 ska_tmc_cdm.schemas.central_node.mccs

The schemas.central_node module defines Marshmallow schemas that map TMC Central Node message classes to/from a JSON representation.

class MCCSAllocateSchema(**args*, ***kwargs*)
Marshmallow schema for the MCCSAllocate class.

create_mccs_allocate(*data*, **_)
Convert parsed JSON back into a MCCSAllocate object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns MCCSAllocate object populated from data

10.3.6 ska_tmc_cdm.schemas.codec

The codec module contains classes used by clients to marshall CDM classes to and from JSON. This saves the clients having to instantiate and manipulate the Marshmallow schema directly.

class MarshmallowCodec
MarshmallowCodec marshalls and unmarshalls CDM classes.

The mapping of CDM classes to Marshmallow schema is defined in this class.

dumps(*obj*, *validate: bool = True*, *strictness: Optional[int] = None*)
Return a string JSON representation of a CDM instance.

The default strictness of the Telescope Model schema validator can be overridden by supplying the validate argument.

Parameters

- **obj** – the instance to marshall to JSON
- **validate** – True to enable schema validation
- **strictness** – optional validation strictness level (0=min, 2=max)

Returns JSON representation of obj

load_from_file(*cls*, *path*, *validate: bool = True*, *strictness: Optional[int] = None*)
Load an instance of a CDM class from disk.

Parameters

- **cls** – the class to create from the file

- **path** – the path to the file
- **validate** – True to enable schema validation
- **strictness** – optional validation strictness level (0=min, 2=max)

Returns an instance of cls

loads(*cdm_class*, *json_data*, *validate: bool = True*, *strictness: Optional[int] = None*)
Create an instance of a CDM class from a JSON string.

The default strictness of the Telescope Model schema validator can be overridden by supplying the validate argument.

Parameters

- **cdm_class** – the class to create from the JSON
- **json_data** – the JSON to unmarshall
- **validate** – True to enable schema validation
- **strictness** – optional validation strictness level (0=min, 2=max)

Returns an instance of cls

register_mapping(*cdm_class*)
A decorator that is used to register the mapping between a Marshmallow schema and the CDM class it serialises.

Parameters **cdm_class** – the CDM class this schema maps to

Returns the decorator

set_schema(*cdm_class*, *schema_class*)
Set the schema for a CDM class.

Parameters

- **schema_class** – Marshmallow schema to map
- **cdm_class** – CDM class the schema maps to

10.3.7 ska_tmc_cdm.schemas.mccscontroller.allocate

The schemas.central_node module defines Marshmallow schemas that map MCCSController AllocateRequest message classes to/from their JSON representation.

class AllocateRequestSchema(**args*, ***kwargs*)
Marshmallow schema for the MCCSController AllocateRequest class.

create_allocaterequest(*data*, **_) → ska_tmc_cdm.messages.mccscontroller.allocate.AllocateRequest
Convert parsed JSON back into an AllocateRequest object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AllocateRequest object populated from data

10.3.8 ska_tmc_cdm.schemas.mccscontroller.releaseresources

The releaseresources module defines Marshmallow schemas that map MCCSController ReleaseResourcesRequest objects to/from their JSON representation.

class ReleaseResourcesRequestSchema (**args*, ***kwargs*)
Marshmallow schema for the ReleaseResourcesRequest class.

create_request (*data*, **_)
Convert parsed JSON from an ReleaseResources request back into an ReleaseResourcesRequest object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ReleaseResourcesRequest object populated from data

10.3.9 ska_tmc_cdm.schemas.mccssubarray.assigned_resources

The assigned_resources module defines Marshmallow schemas that maps the MCCSSubarray.assigned_resources attribute to/from a JSON representation.

class AssignedResourcesSchema (**args*, ***kwargs*)
Marshmallow schema for the MCCSSubarray AssignedResources class.

create_allocaterequest (*data*, **_) → ska_tmc_cdm.messages.mccssubarray.assigned_resources.AssignedResources
Convert parsed JSON back into an AssignedResources object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AssignedResources object populated from data

10.3.10 ska_tmc_cdm.schemas.mccssubarray.configure

The configure module defines Marshmallow schemas that maps the MCCSSubarray.Configure call arguments to/from a JSON representation.

class ConfigureRequestSchema (**args*, ***kwargs*)
Marshmallow schema for the mccssubarray.ConfigureRequest class

create (*data*, **_) → ska_tmc_cdm.messages.mccssubarray.configure.ConfigureRequest

Convert parsed JSON back into a ConfigureRequest object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ConfigureRequest instance populated to match JSON

class StationConfigurationSchema (**args*, ***kwargs*)

create (*data*, **_) → ska_tmc_cdm.messages.mccssubarray.configure.StationConfiguration

Convert parsed JSON back into a StationConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns StnConfiguration instance populated to match JSON

class SubarrayBeamConfiguration(**, subarray_beam_id: int, station_ids: List[int], update_rate: float, channels: List[List[int]], sky_coordinates: List[float], antenna_weights: List[float], phase_centre: List[float]*)

A class to hold subarray beam configuration attributes

10.3.11 ska_tmc_cdm.schemas.mccssubarray.scan

The schemas module defines Marshmallow schemas that map CDM message classes and data model classes to/from a JSON representation.

class ScanRequestSchema(**args*, ***kwargs*)

Create the Schema for ScanRequest

create_scanrequest(*data*, ***_*)

Convert parsed JSON back into a ScanRequest

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ScanRequest instance populated to match JSON

10.3.12 ska_tmc_cdm.schemas.shared

The schemas module defines Marshmallow schemas that are shared by various other serialisation schemas.

class UpperCasedField(**args*, ***kwargs*)

Field that serializes to an upper-case string and deserializes to a lower-case string.

class OrderedSchema(**args*, ***kwargs*)

Subclass of Schema, anything inheriting from Schema has the order of its JSON properties respected in the message. Saves adding a Meta class to everything individually

class Meta

marshmallow directive to respect order of JSON properties in message.

class ValidatingSchema(**args*, ***kwargs*)

ValidatingSchema is a marshmallow schema that calls the appropriate Telescope Model schema validation functions when serialising or deserialising JSON.

validate_json(*data*, *process_fn*)

Validate JSON using the Telescope Model schema.

The process_fn argument can be used to process semantically correct but schematically invalid Python to something equivalent but valid, e.g., to convert a list of Python tuples to a list of lists.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values

- **process_fn** – data processing function called before validation

Returns

validate_on_dump(*data*, *process_fn=<function ValidatingSchema.<lambda>>*, **_)
Validate the serialised object against the relevant Telescope Model schema.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- **process_fn** – function to process data before validation
- _ – unused kwargs passed by Marshmallow

Returns dict suitable for writing as a JSON string

validate_on_load(*data*, *process_fn=<function ValidatingSchema.<lambda>>*, **_)
Validate the JSON string to deserialise.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- **process_fn** – function to process data before validation
- _ – unused kwargs passed by Marshmallow

Returns dict suitable for object constructor.

10.3.13 ska_tmc_cdm.schemas.subarray_node

The schemas.subarray_node package contains Marshmallow schemas that convert JSON to/from the Python classes contained in ska_tmc_cdm.messages.subarray_node.

10.3.14 ska_tmc_cdm.schemas.subarray_node.assigned_resources

This module defines Marshmallow schemas that map CDM classes to/from JSON.

class MCCSAllocationSchema(**args*, ***kwargs*)
Marshmallow schema for the MCCSAllocation class.

create_mccs_allocation(*data*, **_)
Convert parsed JSON back into a MCCSAllocation object.

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns MCCSAllocation object populated from data

class AssignedResourcesSchema(**args*, ***kwargs*)
AssignedResourcesSchema maps the AssignedResources class to/from a JSON representation.

create_assigned_resources(*data*, **_)
Convert parsed JSON back into an AssignedResources object

Parameters

- **data** – Marshmallow-provided dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns AssignedResources object populated from data

10.3.15 ska_tmc_cdm.schemas.subarray_node.configure

The schemas module defines Marshmallow schemas that map CDM message classes and data model classes to/from a JSON representation.

class ConfigureRequestSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.ConfigureRequest class.

create_configuration(*data*, **_)
Converted parsed JSON backn into a subarray_node.ConfigureRequest object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ConfigurationRequest instance populated to match JSON

filter_nulls(*data*, **_)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

class DishConfigurationSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.DishConfiguration class.

convert(*dish_configuration: ska_tmc_cdm.messages.subarray_node.configure.core.DishConfiguration*, **_)
Process DishConfiguration instance so that it is ready for conversion to JSON.

Parameters

- **dish_configuration** – the dish configuration
- _ – kwargs passed by Marshmallow

Returns DishConfiguration instance populated to match JSON

create_dish_configuration(*data*, **_)
Converted parsed JSON back into a subarray_node.DishConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns DishConfiguration instance populated to match JSON

class PointingSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.Pointing class.

create(*data*, **_)
Convert parsed JSON back into a subarray_node.Pointing object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns Pointing instance populated to match JSON

class TargetSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.Target class

convert_to_icrs(*target: ska_tmc_cdm.messages.subarray_node.configure.core.Target*, **_)
Process Target co-ordinates by converting them to ICRS frame before the JSON marshalling process begins.

Parameters

- **target** – Target instance to process
- _ – kwargs passed by Marshallow

Returns SexagesimalTarget with ICRS ra/dec expressed in hms/dms

create_target(*data*, **_)
Convert parsed JSON back into a Target object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns Target instance populated to match JSON

class CSPConfigurationSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.CSPConfiguration class

create(*data*, **_)

Convert parsed JSON back into a CSPConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns CSPConfiguration instance populated to match JSON

validate_on_dump(*data*, **_)
Validating the structure of JSON against schemas and Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

class SubarrayConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_)

Convert parsed JSON back into a SubarrayConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SubarrayConfiguration instance populated to match JSON

Return type *SubarrayConfiguration*

class CommonConfigurationSchema(**args*, ***kwargs*)

convert(*common_configuration: ska_tmc_cdm.messages.subarray_node.configure.csp.CommonConfiguration*, **_)

Process CommonConfiguration instance so that it is ready for conversion to JSON.

Parameters

- **CommonConfiguration** – Common configuration to process
- _ – kwargs passed by Marshmallow

Returns CommonConfiguration instance populated to match JSON

create(*data*, **_)

Convert parsed JSON back into a CSPConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns CommonConfiguration instance populated to match JSON

filter_nulls(*data*, **_)

Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for FSP configuration

class CBFConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_)

Convert parsed JSON back into a CBFConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns CBFConfiguration instance populated to match JSON

Return type *CBFConfiguration*

class FSPConfigurationSchema(**args*, ***kwargs*)

Marshmallow schema for the subarray_node.FSPConfiguration class

convert (*fsp_configuration: ska_tmc_cdm.messages.subarray_node.configure.csp.FSPConfiguration*, **_)

Process FSPConfiguration instance so that it is ready for conversion to JSON.

Parameters

- **fsp_configuration** – FSP configuration to process
- _ – kwargs passed by Marshmallow

Returns FspConfiguration instance populated to match JSON

create (*data*, **_)

Convert parsed JSON back into a FSPConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns FSPConfiguration instance populated to match JSON

filter_nulls (*data*, **_)

Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for FSP configuration

class SDPConfigurationSchema (**args*, ***kwargs*)

Marshmallow class for the SDPConfiguration class

create_sdp_configuration (*data*, **_)

Convert parsed JSON back into a set containing all the scans :param data: dict containing parsed JSON values :param _: kwargs passed by Marshmallow :return: SDPConfiguration instance populated to match JSON

class MCCSConfigurationSchema (**args*, ***kwargs*)

Marshmallow schema for the subarray_node.MCCSConfiguration class

create (*data*, **_)

Convert parsed JSON back into a MCCSConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns MCCSConfiguration instance populated to match JSON

Return type *MCCSConfiguration*

filter_nulls_and_validate_schema (*data*, **_)

validating the structure of JSON against schemas and Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values

- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

validate_json(*data*, *process_fn=<function MCCSConfigurationSchema.<lambda>>*)
validating the structure of JSON against schemas

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- **function** (*lambda*) – use for converting list of tuples to list of list

Returns

validate_schema(*data*, **_)
validating the structure of JSON against schemas

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for CSP configuration

class StnConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_)

Convert parsed JSON back into a StnConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns StnConfiguration instance populated to match JSON

Return type *StnConfiguration*

class SubarrayBeamConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_) → ska_tmc_cdm.messages.subarray_node.configure.mccs.SubarrayBeamConfiguration

Convert parsed JSON back into a SubarrayBeamConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SubarrayBeamConfiguration instance populated to match JSON

10.3.16 ska_tmc_cdm.schemas.subarray_node.configure.core

The schemas module defines Marshmallow schemas that map shared CDM message classes for SubArrayNode configuration to/from a JSON representation.

class ConfigureRequestSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.ConfigureRequest class.

create_configuration(*data*, ***_*)
Converted parsed JSON backn into a subarray_node.ConfigureRequest object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ConfigurationRequest instance populated to match JSON

filter_nulls(*data*, ***_*)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

class DishConfigurationSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.DishConfiguration class.

convert(*dish_configuration: ska_tmc_cdm.messages.subarray_node.configure.core.DishConfiguration*, ***_*)
Process DishConfiguration instance so that it is ready for conversion to JSON.

Parameters

- **dish_configuration** – the dish configuration
- _ – kwargs passed by Marshmallow

Returns DishConfiguration instance populated to match JSON

create_dish_configuration(*data*, ***_*)
Converted parsed JSON back into a subarray_node.DishConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns DishConfiguration instance populated to match JSON

class PointingSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.Pointing class.

create(*data*, ***_*)
Convert parsed JSON back into a subarray_node.Pointing object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns Pointing instance populated to match JSON

class TargetSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.Target class

convert_to_icrs(*target: ska_tmc_cdm.messages.subarray_node.configure.core.Target*, ***_*)
Process Target co-ordinates by converting them to ICRS frame before the JSON marshalling process begins.

Parameters

- **target** – Target instance to process
- _ – kwargs passed by Marshallow

Returns SexagesimalTarget with ICRS ra/dec expressed in hms/dms

create_target(*data*, **_)

Convert parsed JSON back into a Target object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns Target instance populated to match JSON

10.3.17 ska_tmc_cdm.schemas.subarray_node.configure.csp

This module defines Marshmallow schemas that map the CDM classes for SubArrayNode CSP configuration to/from JSON.

class CSPConfigurationSchema(**args*, ***kwargs*)

Marshmallow schema for the subarray_node.CSPConfiguration class

create(*data*, **_)

Convert parsed JSON back into a CSPConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns CSPConfiguration instance populated to match JSON

validate_on_dump(*data*, **_)

Validating the structure of JSON against schemas and Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

class FSPConfigurationSchema(**args*, ***kwargs*)

Marshmallow schema for the subarray_node.FSPConfiguration class

convert(*fsp_configuration: ska_tmc_cdm.messages.subarray_node.configure.csp.FSPConfiguration*, **_)

Process FSPConfiguration instance so that it is ready for conversion to JSON.

Parameters

- **fsp_configuration** – FSP configuration to process
- _ – kwargs passed by Marshmallow

Returns FspConfiguration instance populated to match JSON

create(*data*, **_)

Convert parsed JSON back into a FSPConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns FSPConfiguration instance populated to match JSON

filter_nulls(*data*, **_)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for FSP configuration

class SubarrayConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_)

Convert parsed JSON back into a SubarrayConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SubarrayConfiguration instance populated to match JSON

Return type *SubarrayConfiguration*

class CommonConfigurationSchema(**args*, ***kwargs*)

convert(*common_configuration: ska_tmc_cdm.messages.subarray_node.configure.csp.CommonConfiguration*, **_)
Process CommonConfiguration instance so that it is ready for conversion to JSON.

Parameters

- **CommonConfiguration** – Common configuration to process
- _ – kwargs passed by Marshmallow

Returns CommonConfiguration instance populated to match JSON

create(*data*, **_)

Convert parsed JSON back into a CSPConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns CommonConfiguration instance populated to match JSON

filter_nulls(*data*, **_)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for FSP configuration

class CBFConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_)

Convert parsed JSON back into a CBFConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns CBFConfiguration instance populated to match JSON

Return type *CBFConfiguration*

10.3.18 ska_tmc_cdm.schemas.subarray_node.configure.sdp

This module defines Marshmallow schemas that map the SDPConfiguration message classes to/from JSON.

class SDPConfigurationSchema(**args*, ***kwargs*)
Marshmallow class for the SDPConfiguration class

create_sdp_configuration(*data*, **_)
Convert parsed JSON back into a set containing all the scans :param data: dict containing parsed JSON values :param _: kwargs passed by Marshmallow :return: SDPConfiguration instance populated to match JSON

10.3.19 ska_tmc_cdm.schemas.subarray_node.configure.mccs

This module defines Marshmallow schemas that map the CDM classes for SubArrayNode MCCS configuration to/from JSON.

class MCCSConfigurationSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.MCCSConfiguration class

create(*data*, **_)

Convert parsed JSON back into a MCCSConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns MCCSConfiguration instance populated to match JSON

Return type *MCCSConfiguration*

filter_nulls_and_validate_schema(*data*, **_)
validating the structure of JSON against schemas and Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

validate_json(*data*, *process_fn=<function MCCSConfigurationSchema.<lambda>>*)
validating the structure of JSON against schemas

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- **function** (*lambda*) – use for converting list of tuples to list of list

Returns

validate_schema(*data*, **_)
validating the structure of JSON against schemas

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for CSP configuration

class StnConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_)

Convert parsed JSON back into a StnConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns StnConfiguration instance populated to match JSON

Return type *StnConfiguration*

class SubarrayBeamConfigurationSchema(**args*, ***kwargs*)

create(*data*, **_) → ska_tmc_cdm.messages.subarray_node.configure.mccs.SubarrayBeamConfiguration

Convert parsed JSON back into a SubarrayBeamConfiguration object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns SubarrayBeamConfiguration instance populated to match JSON

class SubarrayBeamTargetSchema(**args*, ***kwargs*)
Marshmallow schema for the subarray_node.Target class

create_target(*data*, **_)
Convert parsed JSON back into a Target object.

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns Target instance populated to match JSON

10.3.20 ska_tmc_cdm.schemas.subarray_node.configure.tmc

The schemas module defines Marshmallow schemas that map CDM message classes and data model classes to/from a JSON representation.

class TMCConfigurationSchema(**args*, ***kwargs*)
Create the Schema for ScanDuration using timedelta

convert_scan_duration_number_to_timedelta(*data*, **_)
Convert parsed JSON back into a TMConfiguration

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns TMCConfiguration instance populated to match JSON

convert_scan_duration_timedelta_to_float(*data: ska_tmc_cdm.messages.subarray_node.configure.tmc.TMCCo*
**_)
Process scan_duration and convert it to a float

Parameters

- **data** – the scan_duration timedelta
- _ – kwargs passed by Marshallow

Returns float converted

10.3.21 ska_tmc_cdm.schemas.subarray_node.scan

The ska_tmc_cdm.schemas.subarray_node.scan module contains Marshmallow schema that map ska_tmc_cdm.schemas.subarray_node.scan message classes to/from JSON.

class ScanRequestSchema(**args*, ***kwargs*)
ScanRequestSchema is the Marshmallow schema that marshals a ScanRequest to/from JSON.

create_scanrequest(*data*, **_)
Convert parsed JSON back into a ScanRequest

Parameters

- **data** – dict containing parsed JSON values
- _ – kwargs passed by Marshmallow

Returns ScanRequest instance populated to match JSON

filter_nulls(*data*, **_)
Filter out null values from JSON.

Parameters

- **data** – Marshmallow-provided dict containing parsed object values
- _ – kwargs passed by Marshmallow

Returns dict suitable for SubArrayNode configuration

CHAPTER 11

ska-tmc-cdm documentation

11.1 Project description

ska-tmc-cdm provides a Python object model and serialisation library for resource allocation commands and telescope configuration commands, with a focus on TMC interfaces with other subsystems. an ICD support library, intended to be used by the Tango clients and Tango servers on opposing sides of a telescope control interface.

11.1.1 Status

This library supports control and configuration payloads for the following Tango devices:

- TMC CentralNode
- TMC SubArrayNode
- MCCSController
- MCCSSubArrayNode

11.2 Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

D

F

S

T

U

V