cbf-sdp-emulator-tango-device

Release 0.5.0

Rodrigo Tobar, Steve Ord

Aug 11, 2021

CONTENTS

CHAPTER

ONE

DEVICE DESCRIPTION

The Tango device included in this project is implemented in the *ska.cbf.tango_device.CbfSubarray* class.

1.1 Commands

This class inherits from `ska.base.SKASubarray` and as such it implements the Observation State Machine. This implies that it supports commands like `Configure` and `Scan`, which is where the action happens. In particular the following commands are meaningful in this device:

- `Configure` triggers the automatic download of the Measurement Set that will be sent by the emulator sender, if not already present on disk. The device will stay in `CONFIGURING` state until the Measurement Set is downloaded and expanded. In case of any error the device transitions to `FAULT`. See *Configuration* for a description of the input for the `Configure` command. Also during `CONFIGURING` the device will resolve the target hostname into its corresponding IP address to avoid name resolution errors at `Scan` time.
- `Scan` starts the transmission of the Measurement Set to the configured endpoint. While the sending is happening the device stays in the `SCANNING` state. After the sending finishes the device moves back to `READY`, unless an error occurs, in which case it moves to `FAULT`.
- `EndScan` cancels the current data sending, if any is occurring at the time.
- `Abort` is like `EndScan` but can be invoked at any time.

1.2 Attributes

The following attributes are supported by `CbfSubarray`:

- `sending` (bool): indicates whether data sending is currently in progress or not. It should closely reflect the device being in `SCANNING` state or not.

1.3 Configuration

Configuration of the Tango device occurs when the `Configure` command is called. `Configure` accepts a single string input, which must correspond to a JSON string. Its contents must be of the form agreed during the PSO-944 conversations as shown here.

The following options must be present in the `fsp` object within the `cbf` object of the root JSON content to configure the behavior of the data sender underlying the Tango device. Most of these options relate to options exposed by the `cbf-sdp-emulator` codebase, so please refer to its documentation for more information about them:

- `outputHost` (`list[list[int, str]]`) a list of pairs, where the second element is the target host. It maps to `transmission.target_host`, defaults to `127.0.0.1`. Only the first element of the outer list is currently considered.
- `outputPort` (`list[list[int, ulong]]`) a list of pairs, where the second element is the target port. It maps to `transmission.target_port_start`, defaults to 41000. Only the first element of the outer list is currently considered.
- `transmission_source` (string) points to the Measurement Set on disk to send, defaults to `sim-vis.ms`.
- `integrationTime` (int) Integration time for the correlation products, specified in milliseconds. It maps to `transmission.time_interval`, defaults to `0`.

As a current extension, the following extra options are supported as well:

- `transmission_rate` (ulong) maps to `tranmission.rate`, defaults to 14750.
- `transmission_num_channels` (ulong) maps to `transmission.channels_per_stream` and to `reader.num_channels`, defaults to 1.
- `transmission_source_url` points to the URL from where the Measurement Set to be sent can be downloaded from. It defaults to `https://gitlab.com/ska-telescope/cbf-sdp-emulator/-/raw/master/data/<transmission_source>.tar.gz`.
- `name_resolution_max_tries` (int): the maximum number of times the code should try to resolve the hostname received via `outputHost` to an IP. Defaults to 100.
- `name_resolution_retry_period` (float): the period between each name resolution retry, in seconds. Defaults to 0.2.
- `transmission_channels_per_stream` (int): transmit multiple streams with specified number of channels per stream. The destination URL is the same but port is incremented by 1 per stream.
- `reader_num_repeats` (int): re-transmit the dataset the given number of times. THe interval and rate is maintained - but the data is repeated.
- `reader_num_timestamps` (int): only read the given number of timestamps

1.4 Running

To start the server, you can run:

```
$> python -m ska.cbf.tango_device <instance_name>
```

By default Tango servers will try to connect to a central database, which involves extra setup prior to launching the command above. If one is simply testing the device on its own this can be avoided though via extra command-flags:

```
$> python -m ska.cbf.tango_device <instance_name> -ORBendPoint giop:tcp:127.0.0.1:10000 -
↪nodb -dlist a/b/c
```

This will start the server without connecting to the central database, exposing the server at the given ip/port combination, and "registering" it with the `a/b/c` name.

An even simpler approach is to use one of Tango's built-in utilities for exactly this purpose:

```
$> python -m tango.test_context ska.cbf.tango_device.CbfSubarray
[...]
Ready to accept request
```

(continues on next page)

(continued from previous page)

```
CbfSubarray started on port 8888 with properties {}
Device access: tango://127.0.0.1:8888/test/nodb/cbfsubarray#dbase=no
Server access: tango://127.0.0.1:8888/dserver/CbfSubarray/cbfsubarray#dbase=no
```

Clients can then connect to the device and its server with the provided URLs.

CHAPTER

TWO

API

class ska.cbf.tango_device.**CbfSubarray**(**args: Any*, ***kwargs: Any*)

A Tango device to control the CBF-SDP interface emulator sender.

This tango device has the exact same interface as Mid-CBF's CbfSubarray, as it's meant to be a drop-in replacement for the latter. To ease this use case and avoid having to write different contents on the Tango DB depending on whether the "real" CbfSubarray device or this one is deployed, this device's class name follows that of Mid-CBF's.

class **AbortCommand**(**args: Any*, ***kwargs: Any*)

do()

Stop the emulator

class **ConfigureScanCommand**(**args: Any*, ***kwargs: Any*)

configure()

do(*argin*)

class **EndScanCommand**(**args: Any*, ***kwargs: Any*)

do()

Like Abort, it ends the sending

class **InitCommand**(**args: Any*, ***kwargs: Any*)

do()

class **ScanCommand**(**args: Any*, ***kwargs: Any*)

do(*argin*)

Start the emulator

This project implements a Tango device that wraps the CBF-SDP Emulator.

Several aspects of the sending process can be adjusted via Tango attributes. This includes the Measurement Set to send, target host/port, data rate and others. In the future the way this configuration takes place might change to accommodate with the rest of the architecture.

The repository structure is based on the tango-example project. However, certain things have been simplified to ease local development without the need to run everything inside containers.

PYTHON MODULE INDEX

S

INDEX