tcs*libdocumentation*

Release 0.5.0.post0

HETDEX collaboration

May 17, 2018

Contents

`tcs_lib` provides some functionalities to simply or proxy the interaction with the HET Telescope Control System (TCS)

This documentation is also available on het-tcs-lib.readthedocs.io.

CHAPTER 1

Installation

1.1 Installation

1.1.1 Instructions

The recommended way

The recommended way to install `tcs_lib` is using pip:

```
pip install --extra-index-url https://gate.mpe.mpg.de/pypi/simple/ tcs_lib
```

It's possible to set the extra index URL permanently by adding the following lines to the `$HOME/.pip/pip.conf` file:

```
[global]
extra-index-url = https://gate.mpe.mpg.de/pypi/simple
```

or exporting the environment variable:

```
export PIP_EXTRA_INDEX_URL=https://gate.mpe.mpg.de/pypi/simple
```

The list of released versions can be seen on the MPE pypi server. A specific version can be installed using specifiers, e.g. issuing `pip install tcs_lib==0.1`.

`pip` will take care of installing *tcs_lib dependances*.

We suggest you install `tcs_lib` into a virtualenv, in an anaconda/conda or in similar environments.

Of course it is also possible to install `tcs_lib` without any of the above with:

```
pip install --user --extra-index-url https://gate.mpe.mpg.de/pypi/simple/ tcs_lib
```

This way the executables shipped with `tcs_lib` are installed in `$HOME/.local/bin`, so make sure to add this to the environment variable `PATH` to be able to easily use them on the command line.

The use of `sudo` when installing with pip is discouraged and potentially harmful.

From local `tcs_lib` copy

If you develop `tcs_lib` or want to use always the latest version, you can install it directly from the checked out svn repository. First you can obtain the source `tcs_lib` code with

```
svn checkout svn://luna.mpe.mpg.de/tcs_lib/trunk tcs_lib
```

If you already have the repository, you can of course keep it up to date with `svn update`

Then you can install the library with:

```
pip install /path/to/tcs_lib
```

or

```
cd /path/to/tcs_lib
pip install .
```

where `/path/to/tcs_lib` is the base directory containing the `setup.py` file.

From online svn repository

It is also possible to install `tcs_lib` directly from the svn repository with

```
pip install svn+svn://luna.mpe.mpg.de/tcs_lib/trunk#egg=tcs_lib
```

If you want to install a specific commit or from a different branch or tag, you can do it issuing one of the following commands

```
pip install svn+svn://luna.mpe.mpg.de/tcs_lib/trunk@5#egg=tcs_lib
pip install svn+svn://luna.mpe.mpg.de/tcs_lib/tag/v0.0.0#egg=tcs_lib
```

Other ways

Once you obtained the source code as in *From local tcs_lib copy*, you can install the code also using the good old

```
cd /path/to/tcs_lib
python setup.py build
python setup.py install
```

We do not recommend this method.

1.1.2 Dependances

Mandatory dependences

```
pyzmq
six
```

Optional dependences

- testing:

```
pytest
pytest-cov

tox
```

- documentation:

```
sphinx
numpydoc
```

- automatic documentation build:

```
sphinx-autobuild
```

1.1.3 Development

If you develop `tcs_lib` we suggest to checkout the svn repository and to install it in "editable" mode and to install all the optional dependances:

```
cd /path/to/tcs_lib
pip install -e .[livedoc]
```

You can also use [not recommended]

```
python setup.py develop
```

See *Contribute to tcs_lib* for more information.

CHAPTER 2

tcs_event - ZeroMQ based listener

This module is the core of tcs_lib.

It is a ZeroMQ based listener that wait for the next TCS event, composed of a header and a payload (a dictionary) and returns it.

2.1 tcs_event – ZMQ based listener of TCS events

This module provides functionalities to interface with TCS events.

TCSEvent is a listener for ZeroMQ events generated by the HETDEX Telescope Control System (TCS).

Urls and topics to subscribe can be provided when creating a new instance or using the *TCSEvent.connect()* and *TCSEvent.subscribe()*.

Events can be received either using the *TCSEvent.next()* method or in a for loop.

SafeTCSEvent is a fail safe version of *TCSEvent*.

TCSDict is a dictionary class that, on initialisation, makes sure that the keywords, that TCS needs to parse events, are initialised. It also provides the *TCSDict.topic* property to get and set the topic associated to the event.

2.1.1 Examples

```
>>> from __future__ import unicode_literals
>>> tcs_dict = TCSDict()
>>> print(*sorted(tcs_dict.keys()))
__data __data_time __key __source __system __wire_time
>>> # get to topic
>>> tcs_dict.topic
Traceback (most recent call last):
...
AttributeError: 'TCSDict' object has no attribute 'topic'
```

(continues on next page)

(continued from previous page)

```
>>> tcs_dict = TCSDict({'__system': 'tcs_lib', '__source': 'tcs_event',
...                     '__key': 'test'})
>>> print(tcs_dict.topic)
tcs_lib.tcs_event.test
>>> # set topic
>>> tcs_dict.topic = 'tcs_lib1.tcs_event1.test1'
>>> print(tcs_dict['__system'], tcs_dict['__source'], tcs_dict['__key'])
tcs_lib1 tcs_event1 test1
>>> tcs_dict.topic = 42
Traceback (most recent call last):
...
TypeError: The "topic" must be string, not "<... 'int'>"
>>> tcs_dict.topic = 'tcs_lib.tcs_event'
Traceback (most recent call last):
...
ValueError: The topic must be a string with format "system.source.key"
```

class tcs_lib.tcs_event.**TCSEvent**(*urls*, *topics=None*, *dates_to_float=False*, *context=None*)

Bases: object

TCS event class that connects to the given list of urls and listens for events. See *next()*

> **Warning:** This object currently does not support concurrency.

Parameters

urls [list of strings] urls to monitor; the list is passed to the *connect()* method

topics [list of strings, optional] topics to subscribe; the list is passed to the *subscribe()* method

dates_to_float [bool, optional] convert __wire_time and __data_time to float before returning the event

context [zmq.Context, optional] context to use when creating the sockets. If not given, uses the global instance returned by zmq.Context.instance()

Examples

```
>>> from tcs_lib import tcs_event
>>> events = tcs_event.TCSEvent(["tcp://127.0.0.1:30301", ],
...                             topics=["tcs.tracker.position",
...                                     "tcs.root.ra_dec"])
>>> for (h, e) in events:
...     if e is not None:
...         handle(e)
```

or

```
>>> while True:
...     (h, e) = events.next()
...     if e is not None:
...         handle(e)
```

subscribe(*topics*)
Subscribe to each of the topics in the given list.

The subscription mechanism works in the following way:

- if `topics` is `None` or an empty list (`[]`) and:
  - no topic has ever been subscribed, subscribe to all (empty string);
  - there are already subscribed topics, do nothing;
- if `topics` is a non empty list and:
  - no topic has ever been subscribed, subscribe to the given topics;
  - all topics are subscribed (empty string), unsubscribe it and then subscribe to the given topics;
  - otherwise subscribe to the new topics, avoiding duplicates

Parameters

topics [list of strings] topics to subscribe

connect(*urls*)
Connect to each of the urls in the given list.

Note that this does not affect any prior connections. However, receipt order is still at the whim of the producers.

Parameters

urls [list of strings] urls to monitor; the list is passed to the *connect()* method

next()
Returns a tuple containing the next event topic and dictionary received. Blocks indefinitely.

Todo: add timeout?

Returns

tuple (topic, dict) topic: string with the topic; dict: dictionary with the event information

Raises

TCSEventIndexError if the incoming multipart event does not have two elements

TCSJSONDecodeError if the json cannot be correctly decoded

__next__()
alias of *next()*, for iteration in python>=3

close()
Close the socket

__iter__()
Return the instance itself for use as iterator

_convert_dates_to_float(*event*)
Convert __wire_time and __data_time to floats. If they are not in the event dictionary, skip

Parameters

event [dict] event dictionary

Returns

event [dict] modified event

class tcs_lib.tcs_event.**SafeTCSEvent**(*log_func*, *re_raise*, **args*, ***kwargs*)

Bases: *tcs_lib.tcs_event.TCSEvent*

Wrap *TCSEvent.next()* in a try/except block to log exceptions and keep doing.

StopIteration and the exceptions passed to re_raise are raised again. All the other exceptions are trapped, a message is passed to the log_func so that the caller can handle the exception at need and the *TCSEvent.next()* method is called again.

Important: Do never ever initialise *SafeTCSEvent* with a function that drops the messages: if you do this you are asking for big troubles. The suggested ways are to use either the TCS logging capability (e.g. via the tcs_proxy.tcs_log class) or the standard python logging module to report error messages.

Parameters

log_func [callable] function accepting one arguments: * message (string): the error message with the traceback

re_raise [iteralbe of exceptions] list of exceptions to re-raise; StopIteration is always re-raised

args, kwarg : position and keyword arguments passed to *TCSEvent*

Examples

```
>>> from __future__ import print_function
>>> from tcs_lib import tcs_event
>>> from tcs_lib import errors
>>> def printer(msg):
...     print('[Error message]', msg)
>>> events = tcs_event.SafeTCSEvent(printer, [],
...                                 ["tcp://127.0.0.1:30301", ],
...                                 topics=["tcs.tracker.position",
...                                         "tcs.root.ra_dec"])
>>> for (h, e) in events:
...     if e is not None:
...         handle(e)
```

Prints to the standard output the error message and the traceback. If e.g. one of the events is ['test',], the following is printed but events keeps listening for events:

```
[Error message] It was not possible to retrieve an event because of the following␣
↪error.
If you think that it is a bug an error that should handle implicitly,
please report this with **the full traceback** to the developers.
Traceback (most recent call last):
File "/data01/montefra/HETDEX/Code/tcs_lib/tcs_lib/tcs_event.py", line 271, in␣
↪next
    return super(SafeTCSEvent, self).next()
File "/data01/montefra/HETDEX/Code/tcs_lib/tcs_lib/tcs_event.py", line 165, in␣
↪next
```

(continues on next page)

(continued from previous page)

```
    raise errors.TCSEventIndexError(msg)
tcs_lib.errors.TCSEventIndexError: The incoming multipart message ['test'] must␣
↪have two elements, not 1
```

If instead the same event comes in and the *SafeTCSEvent* is initialised in the following way:

```
>>> events = tcs_event.SafeTCSEvent(printer, [errors.TCSEventIndexError, ],
...                                 ["tcp://127.0.0.1:30301", ],
...                                 topics=["tcs.tracker.position",
...                                         "tcs.root.ra_dec"])
```

`events` would crash with the above traceback.

Attributes

log_func same as input

re_raise [tuple] exceptions to re-raise

next()
Fail-safe version of *TCSEvent.next()*.

class tcs_lib.tcs_event.**TCSDict**(**args*, ***kwargs*)
Bases: dict

Dictionary providing the basic keys necessary to comply with event TCS API.

When constructing the object, makes sure that the following keys a present:

- __wire_time (string): the time at which the messaging api put the message on the wire (default: '0');
- __data_time (string): a time provided by a developer, ideally this is the temporal reference point for the contents (default: *string_helpers.time_str()*);
- __data (string): boolean indicating whether a data payload follows (default: 'false');
- __system (string): the system that generated the event (first token in the event topic: "system.source.key") (default: empty string)
- __source (string): the source that generated the event (second token in the event topic: "system.source.key") (default: empty string)
- __key (string): the key that generated the event (third token in the event topic: "system.source.key") (default: empty string)

After initializing, execute *ensure_string_times()*. This method can be executed before sending out events, e.g. using *server.ZMQServer*, to make sure that the times are stored in the correct format.

Parameters

args, kwargs: arguments to initialise underlying dict

Attributes

topic Get or set the topic associated with current dictionary event (as stored in the __system, __source and __key keys).

topic
Get or set the topic associated with current dictionary event (as stored in the __system, __source and __key keys).

Raises

AttributeError if, when retrieving the topic, the above keys are empty

TypeError if the new topic is not a string

ValueError if the new topic is not in the form 'system.source.key'

set_data_time()
reset the __data_time value to *string_helpers.time_str()*

ensure_string_times()
Ensure that the __wire_time and __data_time are strings using *string_helpers.TIME_FMT*

CHAPTER 3

Mock TCS events

In order to test *tcs_event* and code using it, some TCS event is needed. Having your own telescope might not be ideal. Therefore we provide a ZeroMQ server, `tcs_lib.zmq_server.ZMQServer`, that can stream TCS like events, and two event generators, `tcs_lib.zmq_server.TCSMockEvent` and `tcs_lib.zmq_server.TCSDBReplay`, that can be plugged into the server. They are also conveniently packaged and exposed via the `tcs_replay` executable described in *Stream TCS events*.

3.1 Stream TCS events

`tcs_replay` provides `tcs_replay`, an executable that allows the user to replay an sqlite3 database as created during TCS operation or a single mock event.

If the name of the database is provided, `tcs_replay` will reconstruct each TCS event and send it via a ZMQ socket approximately at the same rate at which they were created. It is possible to speedup or slowdown the rate at which events are sent and to filter events via command line options. If no database name is provided, a mock event is created and sent continuously.

Run `tcs_replay -h` for further information:

```
usage: tcs_replay [-h] [-V] [-u URL] [--wait-setup WAIT_SETUP]
                  [--serve-forever] [-s SPEEDUP] [-t TOPICS [TOPICS ...]]
                  [-S {none,__data_time,__wire_time}] [--convert-number]
                  [--convert-bool] [--sleep SLEEP]
                  [db_name]

Replays a TCS sqlite database and sends multipart events in the form of
[topic, json(event)] via a ZMQ socket.

positional arguments:
db_name               Name of the database to replay. If not given, a single
                        mock event is sent through the ZMQ socket (default:
                        None)
```

(continues on next page)

(continued from previous page)

```
optional arguments:
-h, --help            show this help message and exit
-V, --version         show program's version number and exit
-u URL, --url URL     IP address and port to which bind the ZMQ socket
                        (default: tcp://127.0.0.1:5556)
--wait-setup WAIT_SETUP
                        Wait for wait_setup seconds after creating the servers
                        to allow to set up the connection (default: 1.0)
--serve-forever       Keep serving events. In the case of the database, it
                        will restart serving the database entries. (default:
                        False)

Database options:
The following options have effect only if the name of the database is
provided.

-s SPEEDUP, --speedup SPEEDUP
                        Speedup to apply when replaying the database. For
                        example, speedup=1 plays back at the original speed;
                        speedup=2 plays it twice as fast (default: 1.0)
-t TOPICS [TOPICS ...], --topics TOPICS [TOPICS ...]
                        List of topics to publish. If no topic is specified,
                        all the topics will be published. (default: None)
-S {none,__data_time,__wire_time}, --sort-by {none,__data_time,__wire_time}
                        Method to time-sort the events before serving them. If
                        'none', no sorting is done and the order of the
                        database is respected. (default: none)
--convert-number      Try to convert values marked as "number" to int or
                        float (default: False)
--convert-bool        Try to convert values marked as "boolean" to ``True``
                        or ``False`` (default: False)

Mock options:
The following options have effect only if the name of the database is not
provided.

--sleep SLEEP         Sleep for sleep seconds before emitting a new signal
                        (default: 1)
```

The implementation of this executable can also be used as example when using custom event generators.

3.2 `server` – ZMQ based server

ZMQ based server to stream content and TCS-like event generators.

class tcs_lib.server.**ZMQServer**(*url*, *context=None*)

Bases: object

Create a ZeroMQ server that publishes content on the give `ulr`.

Sending events is performed by the *send_event()* method using `zmq.Socket.send_multipart()`.

send_event() and *start()* allow to send generic events.

send_tcs_event() is designed to send events that reflect TCS expectations. If the `tcs_event` passed to the method does not contain the `__wire_time` key, set it to `time.time()`.

Parameters

url [string] url and port to bind the socket to

context [zmq.Context, optional] context to use when creating the sockets. If not given, uses the global instance returned by zmq.Context.instance()

send_event(*event*)
Send the event via the socket.

Parameters

event [list of string] event to send using zmq.Socket.send_multipart(); the events are converted to byte string using *string_to_bytes()*.

send_tcs_event(*tcs_topic*, *tcs_event*)
Version of *send_event()* specialized to send TCS-like events.

Set __wire_time in the tcs_event to time.time().

If event is a string, decode it into a dictionary using json.loads().

Parameters

tcs_topic [string] topic of the event

tcs_event [dict or string] event to send

start(*events*)
Start serving events via the socket

Parameters

events [generator yielding lists of strings or bytes] each event retrieved in a loop and sent using *send_event()*. If one event is None, it is not sent.

close()
Close the socket

class tcs_lib.server.**TCSDBReplay**(*db_name*, *sort_by='none'*, *speedup=1.0*, *topics=None*, *convert_number=False*, *convert_bool=False*)
Bases: object

Open a TCS sqlite3 database, query it and return one entry at a time when looping or using the next() builtin function.

Parameters

db_name [string] file containing the database

sort_by [string, optional] whether to sort or not the event_ids. Accepted values: 'none', '__data_time', '__wire_time'

speedup [float, optional] speedup to use to replay the database. A value larger than 1 fast-forwards the replay, a smaller value slows the replay down

topics [list, optional] list of topics to return when iterating. If None or [], all topics are returned.

convert_number, convert_bool [bool, optional] try to convert database entries to number (int or float) and to boolean, according to the value of the data_type column. The conversion is fail-safe

Attributes

event_ids [iterator] results of the query for the event_id. The attribute is filled by the *_reset_iter()*

start_wire_time, start_iter_time [float] wire time and current time when calling *__next__()* the first time. Call *_reset_iter()* to reset them before starting a new iteration

_reset_iter()
Reset the status and allow restarting the iterations.

- Create and execute the query to retrieve the `event_id` and save it in `event_ids`. The query is done on the `event` table, if no ordering `'none'` or the `'__data_time'` ordering is required, or on the `attribute` table, if the `'__wire_time'` ordering is required.
- Unset the `start_wire_time` and `start_iter_time`

This method is called when initializing the class. It can be called to reinitialize the iterator.

Raises

tcs_lib.DBOrderingError if the ordering is not known

__iter__()
Return the instance for use as iterator

__next__()
Get the next element of `event_id`, build the TCS object and return it.

Returns

list of strings or ''None'' if the topic is accepted returns `[topic, json(event)]`, otherwise returns `None`

_event_dict(*event_id*)
Create dictionary representing the event with id `event_id`

Parameters

event_id [string] id of the event

Returns

result [dictionary] event information

_convert_to_type(*value*, *type_*)
Try to convert value to the given type, if the conversion is required

Parameters

value [string] value to convert

type_ [string] type of value. Known types: "number", "boolean", "string". Any unknown type is treated as a string

Returns

value [int, float, bool or string] converted value

_convert_to_number(*value*)
Try to convert the input `value` from string to `int` or `float`, in that order. If it fails, returns `value` unchanged

_convert_to_bool(*value*)
Try to convert the input `value` from `"true"/"false"` to `True`/`False`. If it fails, returns `value` unchanged

_update_times(*event_dict*)
Update the `'__data_time'` and `'__wire_time'` as an offset from `start_iter_time` according to the required speedup.

Parameters

event_dict [dictionary] event information

Returns

event_dict [dictionary] updated input

class tcs_lib.server.**TCSMockEvent**(*sleep=1*)

Bases: object

Iterator class that always return a mock event `'lrs2.hardware.status'`

Parameters

sleep [float, optional] sleep `sleep` seconds between events

__iter__()

Return the instance for use as iterator

__next__()

Return a mock event after sleeping for one second. It never raises a StopIteration exception.

Returns

list of strings topic and mock event as `[topic, json(event)]`

CHAPTER 4

TCS proxy

4.1 `tcs_proxy` – Proxy to the TCS modules

Testing code that uses TCS logging and the TCS subsystem interface outside of HET is not possible. Therefore one need to create classes that mocks the original and a way to easily switch between the development and the production environment.

The `tcs_lib.tcs_proxy` provides a proxy between an application and the TCS systems. It is a thin layer that initialize the necessary objects and allow easy access to them in a transparent and flexible way.

After importing the module:

```
from tcs_lib import tcs_proxy
```

we must initialize the proxy before being able to use it:

```
tcs_proxy.init(conf, section='urls')
```

`conf` is a configuration object containing in the section `section` all the entries needed to create the logging object and the necessary TCS subsystems. The configuration entries necessary to initialize the proxy are described in *`init()`*. An example configuration to use to initialize the logger and two TCS subsystems is:

```
[urls]
# url to use for the TCS logger, if the module implementing it is found
tcs_log = tcp://127.0.0.1:5555
# if the TCS logger is not found, use a mock object and use the file of the
# next option to configuration the python loggers to used for the mocks
tcs_log_mock_path = /path/to/conf_file.cfg

# We also want two TCS subsystems called tcs and virus
subsystem_names = tcs, virus

# the tcs subsystem url. But we don't need the tcs_mock_path since the
# configuration is already in ``tcs_log_mock_path``
```

(continues on next page)

(continued from previous page)

```
tcs = tcp://127.0.0.1:5556
# and now the virus subsystem
virus = tcp://127.0.0.1:5557
# for virus we might have a difference logger configuration file for the
# mock object
virus_mock_path = /path/to/virus_logger.cfg
```

After the initialisation, the following attributes become available:

- `tcs_proxy.tcs_log`: this attribute refers either to an instance of `TCSLog.TCSLog` or a wrapper around a standard python logger that exposes the `log_*` methods listend in *_MockTCSLog*
- `tcs_proxy.tcs`, `tcs_proxy.virus`: these attributes refer either to instances of `tcssubsystem.TCSSubSystem` or of a mock class that echoes the method called and their arguments via a standard python logger.

Besides those, the TCS proxy exposes also an other attribute: `tcs_proxy.errors`. If the `tcssubsystem` is importable, this attribute is an alias of the module itself: i.e. `tcs_proxy.errors.error` is equivalent to `tcssubsystem.error`. If the module is not found, any variable name is associated with `Exception`: i.e. both `tcs_proxy.errors.error` and `tcs_proxy.errors.MyPersonalError` are equivalent to `Exception`. This attribute allows to catch and/or raise TCS exceptions, when available, or generic ones otherwise.

As said the mock objects use python loggers to log messages and method calls, whose names are `{mod_name}.{name}` where `mod_name` is the value of `tcs_proxy.mod_name`, defaulting to `tcs_lib.tcs_proxy`, and `name` is either `tcs_log` or the name of the TCS subsystem. For other software using the proxy it might be preferable to set the `mod_name` **before** initializing it, in order to *own* the loggers. So for a package `my_package` I advice to do the following:

```
from tcs_lib import tcs_proxy
tcs_proxy.mod_name = 'my_package.tcs_proxy'
tcs_proxy.init(conf, section='urls')
```

The initialised proxies can be cleared using the *clear()* function. The TCS proxy will be also marked as not initialized.

Note: Assigning `tcs_proxy.mod_name = 'my_package.tcs_proxy'` after initializing the proxy results in `AttributeError`. If necessary it is possible to *clear()* the proxy, set the `mod_name` and then re-initialize.

4.1.1 Implementation

Proxy for the python interface to the HET Telescope Control System (TCS).

This module tries to import `tcssubsystem` and `TCSLog`:

- if it succeeds, initialize all the requested TCS instances.
- otherwise use some mock classes that log every command are used

class tcs_lib._tcs_proxy.**TCSProxy**(*mod_name='tcs_lib._tcs_proxy'*)

Bases: `object`

Proxy implementation. One instance is exposed as `tcs_lib.tcs_proxy`.

The only method public method is *init()* and must be called before the proxy attributes can be used.

Parameters

mod_name [string, optional] name of the module to use by the mock `TCSLog` and `tcssubsystem`

Attributes

mod_name Name of the module used to initialize the proxy.

tcs_log [`TCSLog.TCSLog` or *_MockTCSLog*] instance of the TCSLog class or of the corresponding mock version

{name} [`tcssubsystem.TCSSubSystem` or *_MockTCSSubsystem*] for each name in the `subsystem_names` configuration option, instance of the TCSSubSystem class or of the corresponding mock version

errors proxy objects for TCS exceptions. If the `tcssubsystem` module is found, this attribute it's an alias of the module, otherwise it is an instances of *_MockErrors*

mod_name
Name of the module used to initialize the proxy. Can be modified only before calling *init()*

init(*conf*, *section='urls'*)
Initialize the TCSLog and tcssubsystems, or their mock counterpart.

The following configuration entries from the `section` are used:

- `tcs_log`, optional: url to use to initialise the `TCSLog.TCSLog` class. If not given, empty or the `TCSLog` module is not found, initialize a mock log class
- `tcs_log_mock_path`, optional: if the `TCSLog.TCSLog` cannot be initialized, a mock class is used instead. This class is a proxy for a standard python logger with name `{mod_name}.tcs_log`. By default the `NullHandler` is attached. However it is possible to customize the loggers, using a configuration file according to the logger configuration file format. The name of this configuration file can be passed using the `tcs_log_mock_path` option.
- `subsystem_names`, optional: list of comma separated names of TCS subsystems to initialize. For each `name` the following options are used
  - `{name}`, optional: url to use to initialise the `tcssubsystem.TCSSubSystem` class. If not given, empty or the `tcssubsystem` module is not found, initialize a mock subsystem class
  - `{name}_mock_path`, optional: the option has the same meaning of the `tcs_log_mock_path` one, with the difference that the python logger name use is `{mod_name}.{name}`

Note: If the logger configuration file contains the configurations for the mock logger and subsystems, there is no need to assign it to every `*_mock_path`, but only to the first one used, e.g. `tcs_log_mock_path`.

Parameters

conf [`configparser.ConfigParser` instance] configuration files necessary to initialize the TCS log and subsystems.

section [string, optional] name of the section containing the configuration

clear()
Clear the attributes created by *init()* and reset the status to uninitialized.

_init_tcs_log(*conf*, *section*)
Initialize the tcs logging

Parameters

conf [configparser.ConfigParser instance] configuration files necessary to initialize the TCS log and subsystems.

section [string, optional] name of the section containing the configuration

Returns

:class:'TCSLog.TCSLog' or :class:'_MockTCSLog'

_init_tcs_subystem(*name*, *conf*, *section*)
Initialize the TCS subsystem or mock subsystem and return it.

Parameters

name [string] name of the subsystem

conf [configparser.ConfigParser instance] configuration files necessary to initialize the TCS log and subsystems.

section [string, optional] name of the section containing the configuration

Returns

:class:'tcssubsystem.TCSSubSystem' or :class:'_MockTCSSubsystem'

__getattr__(*name*)
Try to get TCS log/subsystem called name. For sure this will fail if used before calling *init()*

Parameters

name [string] attribute name

Raises

AttributeError if name is not a valid attribute

class tcs_lib._tcs_proxy.**_MockErrors**
Bases: object

Instances of this class return an Exception as attribute.

class tcs_lib._tcs_proxy.**_MockTCSSubsystem**(*name*, *mod_name*, *log_conf_file*)
Bases: object

Class that log the methods called and their positional and keyword arguments.

The name of the logger used is {mod_name}.{name} where name is the name to the constructor.

In order to customize this logger, use the ocd configuration file as described here. If log_conf_file is an empty string or if the configuration parsing fails, the logging.NullHandler is added to the logger used here. In case of a parsing failure a warning will be printed.

Parameters

name: string name of the instance created

mod_name [string] name of module to associate to the mock proxy objects; used to create the logger name

log_conf_file [string] name of the configuration files necessary to initialize the mock tcs log

__getattr__(*name*)
Return a function that logs the call

Parameters

name [string] attribute name

class tcs_lib._tcs_proxy.**_MockTCSLog**(*name*, *mod_name*, *log_conf_file*)
Bases: *tcs_lib._tcs_proxy._MockTCSSubsystem*

Replacement for the TCSLog.TCSLog when the TCSLog module is not available.

This class translates the TCSLog.TCSLog log_* calls to the appropriate python logging levels:

method	log level
log_debug	10 (DEBUG)
log_info	20 (INFO)
log_warn	30 (WARNING)
log_error	40 (ERROR)
log_fatal	50 (CRITICAL)
log_alarm	60

The TCSLog.TCSLog.log_*() methods have the following signature:

```
log_info(msg, *args)
```

The log message is formatted with msg.format(*args) before emitting it.

Warning: Because of the formatting, log messages are emitted by a function in **this** module and not in the place where the log_* method is called.

The name of the logger used is {mod_name}.{name}. When initialised by *TCSProxy* {name} is tcs_log.

Parameters

name: string name of the logger to use

mod_name [string] name of module to associate to the mock proxy objects; used to create the logger name

log_conf_file [string] name of the configuration files necessary to initialize the mock tcs log

__getattr__(*name*)
convert calls to log_*.

Parameters

name [string] attribute name

Raises

AttributeError if name is not one of the valid log_* names

class tcs_lib._tcs_proxy.**_LogCall**(*log*, *obj*, *function*)
Bases: object

Callable class log the call to function with all its positional and keyword arguments.

The function is logged as info.

Parameters

log [logging.Logger] logger

obj, function [string] name of the object and the function called

__call__(**args*, ***kwargs*)
Log the call of the function as "{obj}.{function}({args}, {kwargs})".

Returns

:class:'_MockResponse' current instance that mimic the dictionary returned by each TCSSubSystem method

class tcs_lib._tcs_proxy.**_MockResponse**
Bases: dict

Extension of a dictionary that returns the key if not present.

Examples

```
>>> d = _MockResponse(a=42)
>>> d['a']
42
>>> key = 'b'
>>> d[key] == key
True
```

__getitem__(*key*)
If the key is not found, return it, otherwise returns the associated value

CHAPTER 5

Other modules

5.1 errors – Error defintions

Module with the Exception definitions used in tcs_lib

exception tcs_lib.errors.**TCSLibError**
Bases: Exception

Error used as base class for all the others

exception tcs_lib.errors.**DBOrderingError**
Bases: *tcs_lib.errors.TCSLibError*, ValueError

The database ordering keyword is not correct

exception tcs_lib.errors.**ConvertTypeError**
Bases: *tcs_lib.errors.TCSLibError*, TypeError

Raised when the type to encode/decode is not correct

exception tcs_lib.errors.**TCSEventIndexError**
Bases: *tcs_lib.errors.TCSLibError*, IndexError

Raise when an incoming TCS event is not composes of two parts

exception tcs_lib.errors.**TCSJSONDecodeError**
Bases: *tcs_lib.errors.TCSLibError*, ValueError

Raise when the second part of the TCS event is not a valid json

5.2 string_helpers – String/Byte helper functionalities

Utilities to deal with strings and bytes for a better python2/3 compatibility

tcs_lib.string_helpers.**TIME_FMT = '{t:.9f}'**
Default formatter to times in TCS events

tcs_lib.string_helpers.**bytes_to_string**(*byte*, *encoding='utf-8'*)

If the input is a string or a list of strings, returns it, otherwise interprets it as a byte string or a list of bytes strings.

Parameters

byte [(list of) byte string(s)] byte string or list of byte strings to decode

encoding [string, optional] encoding to use; see Standard Encodings for a list of possible encodings

Returns

Python2: input; Python3: (list of) string(s). If the input is not iterable returns it

Raises

ConvertTypeError if the input is not a string/byte string or a list thereof

tcs_lib.string_helpers.**string_to_bytes**(*string*, *encoding='utf-8'*)

If the input is a byte string or a list of byte strings, returns it, otherwise interprets it as a string or a list of strings.

Parameters

string [(list of) string(s)] string or list of strings to encode

encoding [string, optional] encoding to use; see Standard Encodings for a list of possible encodings

Returns

Python2: input; Python3: (list of) byte string(s). If the input is not iterable returns it

Raises

ConvertTypeError if the input is not a string/byte string or a list thereof

tcs_lib.string_helpers.**time_str**()

Returns

string `time.time` as a string with format *TIME_FMT*

CHAPTER 6

Contribution guide

6.1 Contribute to `tcs_lib`

6.1.1 How To

The suggested workflow for implementing bug fixes and/or new features is the following:

- Identify or, if necessary, add to our redmine issue tracker one or more issues to tackle. Multiple issues can be addressed together if they belong together. Assign the issues to yourself.
- Create a new branch from the trunk with a name either referring to the topic or the issue to solve. E.g. if you need to add a new executable, tracked by issue #1111 `do_something`:

```
svn cp ^/trunk ^/branches/do_something_1111\
-m 'create branch to solve issue #1111'
```

- Switch to the branch:

```
svn switch ^/branches/do_something_1111
```

- Implement the required changes and don't forget to track your progress on Redmine. If the feature/bug fix requires a large amount of time, we suggest, when possible, to avoid one big commit at the end in favour of smaller commits. In this way, in case of breakages, is easier to traverse the branch history and find the offending code. For each commit you should add an entry in the `Changelog` file.

  If you work on multiple issues on the same branch, close one issue before proceeding to the next. When closing one issue is good habit to add in the description on the redmine the revision that resolves it.

- Every function or class added or modified should be adequately documented as described in *Coding style*.

  Documentation is essential both for users and for your fellow developers to understand the scope and signature of functions and classes. If a new module is added, it should be also added to the documentation in the appropriate place. See the existing documentation for examples.

  Each executable should be documented and its description should contain enough information and examples to allow users to easily run it.

- Every functionality should be thoroughly tested for python 2.7, 3.4, 3.5 and 3.6 in order to ensure that the code behaves as expected and that future modifications will not break existing functionalities. When fixing bugs, add tests to ensure that the bug will not repeat. For more information see *Testing*.
- Once the issue(s) are solved and the branch is ready, merge any pending change **from** the trunk:

```
svn merge ^/trunk
```

  While doing the merge, you might be asked to manually resolve one or more conflicts. Once all the conflicts have been solved, commit the changes with a meaningful commit message, e.g.: `merge ^/trunk into ^/branches/do_something_1111`. Then rerun the test suite to make sure your changes do not break functionalities implemented while you were working on your branch.

- Then contact the maintainer of `tcs_lib` and ask to merge your branch **back to the trunk**.

Information about branching and merging can be found in the svn book. For any questions or if you need support do not hesitate to contact the maintainer or the other developers.

6.1.2 Coding style

All the code should be compliant with the official python style guidelines described in **PEP 8**. To help you keep the code in spec, we suggest to install plugins that check the code for you, like Synstastic for vim or flycheck for Emacs.

The code should also be thoroughly documented using the numpy style. See the existing documentation for examples.

6.1.3 Testing

`tcs_lib` uses the testing framework provided by the pytest package. The tests should cover every aspect of a function or method. If exceptions are explicitly raised, this should also tested to ensure that the implementation behaves as expected.

The preferred way to run the tests is using tox, an automatised test help package. If you have installed tox, with e.g. `pip install tox`, you can run it by typing:

```
tox
```

It will take care of creating virtual environments for every supported version of python (2.7, 3.4, 3.5 and 3.6), if it exists on the system, install `tcs_lib`, its dependences and the packages necessary to run the tests and runs `py.test`.

You can run the tests for a specific python version using:

```
py.test
```

or:

```
python setup.py test
```

The latter command fetches all the needed dependences, among others `pytest` itself, will be fetched and installed in a `.eggs` directory. Then it will run `py.test`. This command might fail when running in a virtual environment. Use the option `--addopts` to pass additional options to `py.test`.

You can run specific tests providing the file name(s) and, optionally the name of a test. E.g.:

```
py.test tests/test_logging_helper.py  # runs only the tests in the logging helper file
py.test tests/test_logging_helper.py::test_log_setup  # runs only one test
```

Relevant command line options:

```
-v                    verbose output: print the names and parameters of the
                      tests
-s                    capture standard output: can cause weird interactions
                      with the logging module
```

A code coverage report is also created thanks to the pytest-cov plugin and can be visualized opening into a browser `cover/index.html`. If you want a recap of the coverage directly in the terminal you can provide one of the following options when running `py.test`:

```
--cov-report term
--cov-report term-missing
```

Besides running the tests, the `tox` command also builds, by default, the documentation and collates the coverage tests from the various python interpreters and can copy then to some directory. To do the latter create, if necessary, the configuration file `~/.config/little_deploy.cfg` and add to it a section called `tcs_lib` with either one or both of the following options:

```
[tcs_lib]
# if given the deploys the documentation to the given dir
doc = /path/to/dir
# if given the deploys the coverage report to the given dir
cover = /path/to/other/dir

# it's also possible to insert the project name and the type of the document
# to deploy using the {project} and {type_} placeholders. E.g
# cover = /path/to/dir/{project}_{type_}
# will be expanded to /path/to/dir/tcs_lib_cover
```

For more information about the configuration file check little_deploy.

For other command line arguments type:

```
py.test -h
```

For a list of available fixtures type:

```
py.test --fixtures tests/
```

tox and pyenv

Many systems have a limited number of python versions installed. pyenv provides ways to have multiple python versions that can be used by `tox` via the tox-pyenv plugin.

Here we outline the steps necessary to make `tox` use `pyenv`:

- Install `pyenv` following these instructions. We suggest to use `brew` under Mac OS X or the automatic installer. When is done, follow the instructions to enables `pyenv`.
- Install the python versions that you need. E.g. if you have python 2.7 and 3.6 on you system, you can install only missing versions, e.g.:

  ```
  pyenv install 3.4.6
  pyenv install 3.5.3
  ```

  Of course you can also install 2.7 and 3.6 using pyenv.

- Install `tox-pyenv` in the same place where `tox` is installed, i.e. either on the system, a virtual environment or a `pyenv` instance:

```
pip install tox-pyenv
```

This way `tox` is can use `pyenv which` to locate a required python version

- The last step consists in letting `pyenv` know which python versions to use. If you have already set pyenv global to all the version required for testing you should be done. Otherwise go to the `tcs_lib` directory and run pyenv local:

```
pyenv local system 3.4.6 3.5.3
```

This command creates a file called `.python-version` that contains the following three lines:

```
system
3.4.6
3.5.3
```

It will make `pyenv which` look for python versions in the system directories as well as within the `pyenv` directory.

If you did installed also other versions of python (e.g. 3.6.0 and 2.7.13) under `pyenv` and want to use them instead of the system ones, you can use:

```
pyenv local 3.6.0 3.4.6 3.5.3 2.7.13
```

- Run `tox`: now you will be able to use all the python version that tox requires.

6.1.4 Documentation

To build the documentation you need the additional dependences described in *Optional dependences*. They can be installed by hand or during `tcs_lib` installation by executing one of the following commands on a local copy:

```
pip install /path/to/tcs_lib[doc]
pip install /path/to/tcs_lib[livedoc]
```

The first install `sphinx`, the `alabaster` theme and the `numpydoc` extension; the second also installs `sphinx-autobuild`.

To build the documentation in html format go to the `doc` directory and run:

```
make html
```

The output is saved in `doc/build/html`. For the full list of available targets type `make help`.

If you are updating the documentation and want avoid the `edit-compile-browser refresh` cycle, and you have installed `sphinx-autobuild`, type:

```
make livehtml
```

then visit http://127.0.0.1:8000. The html documentation is automatically rebuilt after every change of the source and the browser reloaded.

CHAPTER 7

About

7.1 LICENSE

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
```

(continues on next page)

(continued from previous page)

```
you modify it: responsibilities to respect the freedom of others.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
patents cannot be used to render the program non-free.

  The precise terms and conditions for copying, distribution and
modification follow.

                       TERMS AND CONDITIONS

  0. Definitions.

  "This License" refers to version 3 of the GNU General Public License.

  "Copyright" also means copyright-like laws that apply to other kinds of
works, such as semiconductor masks.

  "The Program" refers to any copyrightable work licensed under this
License.  Each licensee is addressed as "you".  "Licensees" and
"recipients" may be individuals or organizations.

  To "modify" a work means to copy from or adapt all or part of the work
in a fashion requiring copyright permission, other than the making of an
exact copy.  The resulting work is called a "modified version" of the
earlier work or a work "based on" the earlier work.
```

(continues on next page)

(continued from previous page)

```
  A "covered work" means either the unmodified Program or a work based
on the Program.

  To "propagate" a work means to do anything with it that, without
permission, would make you directly or secondarily liable for
infringement under applicable copyright law, except executing it on a
computer or modifying a private copy.  Propagation includes copying,
distribution (with or without modification), making available to the
public, and in some countries other activities as well.

  To "convey" a work means any kind of propagation that enables other
parties to make or receive copies.  Mere interaction with a user through
a computer network, with no transfer of a copy, is not conveying.

  An interactive user interface displays "Appropriate Legal Notices"
to the extent that it includes a convenient and prominently visible
feature that (1) displays an appropriate copyright notice, and (2)
tells the user that there is no warranty for the work (except to the
extent that warranties are provided), that licensees may convey the
work under this License, and how to view a copy of this License.  If
the interface presents a list of user commands or options, such as a
menu, a prominent item in the list meets this criterion.

  1. Source Code.

  The "source code" for a work means the preferred form of the work
for making modifications to it.  "Object code" means any non-source
form of a work.

  A "Standard Interface" means an interface that either is an official
standard defined by a recognized standards body, or, in the case of
interfaces specified for a particular programming language, one that
is widely used among developers working in that language.

  The "System Libraries" of an executable work include anything, other
than the work as a whole, that (a) is included in the normal form of
packaging a Major Component, but which is not part of that Major
Component, and (b) serves only to enable use of the work with that
Major Component, or to implement a Standard Interface for which an
implementation is available to the public in source code form.  A
"Major Component", in this context, means a major essential component
(kernel, window system, and so on) of the specific operating system
(if any) on which the executable work runs, or a compiler used to
produce the work, or an object code interpreter used to run it.

  The "Corresponding Source" for a work in object code form means all
the source code needed to generate, install, and (for an executable
work) run the object code and to modify the work, including scripts to
control those activities.  However, it does not include the work's
System Libraries, or general-purpose tools or generally available free
programs which are used unmodified in performing those activities but
which are not part of the work.  For example, Corresponding Source
includes interface definition files associated with source files for
the work, and the source code for shared libraries and dynamically
linked subprograms that the work is specifically designed to require,
such as by intimate data communication or control flow between those
subprograms and other parts of the work.
```

(continues on next page)

(continued from previous page)

```
  The Corresponding Source need not include anything that users
can regenerate automatically from other parts of the Corresponding
Source.

  The Corresponding Source for a work in source code form is that
same work.

  2. Basic Permissions.

  All rights granted under this License are granted for the term of
copyright on the Program, and are irrevocable provided the stated
conditions are met.  This License explicitly affirms your unlimited
permission to run the unmodified Program.  The output from running a
covered work is covered by this License only if the output, given its
content, constitutes a covered work.  This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

  You may make, run and propagate covered works that you do not
convey, without conditions so long as your license otherwise remains
in force.  You may convey covered works to others for the sole purpose
of having them make modifications exclusively for you, or provide you
with facilities for running those works, provided that you comply with
the terms of this License in conveying all material for which you do
not control copyright.  Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction
and control, on terms that prohibit them from making any copies of
your copyrighted material outside their relationship with you.

  Conveying under any other circumstances is permitted solely under
the conditions stated below.  Sublicensing is not allowed; section 10
makes it unnecessary.

  3. Protecting Users' Legal Rights From Anti-Circumvention Law.

  No covered work shall be deemed part of an effective technological
measure under any applicable law fulfilling obligations under article
11 of the WIPO copyright treaty adopted on 20 December 1996, or
similar laws prohibiting or restricting circumvention of such
measures.

  When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention
is effected by exercising rights under this License with respect to
the covered work, and you disclaim any intention to limit operation or
modification of the work as a means of enforcing, against the work's
users, your or third parties' legal rights to forbid circumvention of
technological measures.

  4. Conveying Verbatim Copies.

  You may convey verbatim copies of the Program's source code as you
receive it, in any medium, provided that you conspicuously and
appropriately publish on each copy an appropriate copyright notice;
keep intact all notices stating that this License and any
non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all
```

(continues on next page)

(continued from previous page)

```
recipients a copy of this License along with the Program.

  You may charge any price or no price for each copy that you convey,
and you may offer support or warranty protection for a fee.

  5. Conveying Modified Source Versions.

  You may convey a work based on the Program, or the modifications to
produce it from the Program, in the form of source code under the
terms of section 4, provided that you also meet all of these conditions:

    a) The work must carry prominent notices stating that you modified
    it, and giving a relevant date.

    b) The work must carry prominent notices stating that it is
    released under this License and any conditions added under section
    7.  This requirement modifies the requirement in section 4 to
    "keep intact all notices".

    c) You must license the entire work, as a whole, under this
    License to anyone who comes into possession of a copy.  This
    License will therefore apply, along with any applicable section 7
    additional terms, to the whole of the work, and all its parts,
    regardless of how they are packaged.  This License gives no
    permission to license the work in any other way, but it does not
    invalidate such permission if you have separately received it.

    d) If the work has interactive user interfaces, each must display
    Appropriate Legal Notices; however, if the Program has interactive
    interfaces that do not display Appropriate Legal Notices, your
    work need not make them do so.

  A compilation of a covered work with other separate and independent
works, which are not by their nature extensions of the covered work,
and which are not combined with it such as to form a larger program,
in or on a volume of a storage or distribution medium, is called an
"aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users
beyond what the individual works permit.  Inclusion of a covered work
in an aggregate does not cause this License to apply to the other
parts of the aggregate.

  6. Conveying Non-Source Forms.

  You may convey a covered work in object code form under the terms
of sections 4 and 5, provided that you also convey the
machine-readable Corresponding Source under the terms of this License,
in one of these ways:

    a) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by the
    Corresponding Source fixed on a durable physical medium
    customarily used for software interchange.

    b) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by a
    written offer, valid for at least three years and valid for as
```

(continues on next page)

(continued from previous page)

```
    long as you offer spare parts or customer support for that product
    model, to give anyone who possesses the object code either (1) a
    copy of the Corresponding Source for all the software in the
    product that is covered by this License, on a durable physical
    medium customarily used for software interchange, for a price no
    more than your reasonable cost of physically performing this
    conveying of source, or (2) access to copy the
    Corresponding Source from a network server at no charge.

    c) Convey individual copies of the object code with a copy of the
    written offer to provide the Corresponding Source.  This
    alternative is allowed only occasionally and noncommercially, and
    only if you received the object code with such an offer, in accord
    with subsection 6b.

    d) Convey the object code by offering access from a designated
    place (gratis or for a charge), and offer equivalent access to the
    Corresponding Source in the same way through the same place at no
    further charge.  You need not require recipients to copy the
    Corresponding Source along with the object code.  If the place to
    copy the object code is a network server, the Corresponding Source
    may be on a different server (operated by you or a third party)
    that supports equivalent copying facilities, provided you maintain
    clear directions next to the object code saying where to find the
    Corresponding Source.  Regardless of what server hosts the
    Corresponding Source, you remain obligated to ensure that it is
    available for as long as needed to satisfy these requirements.

    e) Convey the object code using peer-to-peer transmission, provided
    you inform other peers where the object code and Corresponding
    Source of the work are being offered to the general public at no
    charge under subsection 6d.

  A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

  A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, "normally used" refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product.  A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

  "Installation Information" for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source.  The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.
```

(continues on next page)

(continued from previous page)

```
  If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.

  "Additional permissions" are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law.  If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

  When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it.  (Additional permissions may be written to require their own
removal in certain cases when you modify the work.)  You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

  Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

    a) Disclaiming warranty or limiting liability differently from the
    terms of sections 15 and 16 of this License; or

    b) Requiring preservation of specified reasonable legal notices or
    author attributions in that material or in the Appropriate Legal
    Notices displayed by works containing it; or

    c) Prohibiting misrepresentation of the origin of that material, or
    requiring that modified versions of such material be marked in
```

(continues on next page)

(continued from previous page)

```
    reasonable ways as different from the original version; or

    d) Limiting the use for publicity purposes of names of licensors or
    authors of the material; or

    e) Declining to grant rights under trademark law for use of some
    trade names, trademarks, or service marks; or

    f) Requiring indemnification of licensors and authors of that
    material by anyone who conveys the material (or modified versions of
    it) with contractual assumptions of liability to the recipient, for
    any liability that these contractual assumptions directly impose on
    those licensors and authors.

  All other non-permissive additional terms are considered "further
restrictions" within the meaning of section 10.  If the Program as you
received it, or any part of it, contains a notice stating that it is
governed by this License along with a term that is a further
restriction, you may remove that term.  If a license document contains
a further restriction but permits relicensing or conveying under this
License, you may add to a covered work material governed by the terms
of that license document, provided that the further restriction does
not survive such relicensing or conveying.

  If you add terms to a covered work in accord with this section, you
must place, in the relevant source files, a statement of the
additional terms that apply to those files, or a notice indicating
where to find the applicable terms.

  Additional terms, permissive or non-permissive, may be stated in the
form of a separately written license, or stated as exceptions;
the above requirements apply either way.

  8. Termination.

  You may not propagate or modify a covered work except as expressly
provided under this License.  Any attempt otherwise to propagate or
modify it is void, and will automatically terminate your rights under
this License (including any patent licenses granted under the third
paragraph of section 11).

  However, if you cease all violation of this License, then your
license from a particular copyright holder is reinstated (a)
provisionally, unless and until the copyright holder explicitly and
finally terminates your license, and (b) permanently, if the copyright
holder fails to notify you of the violation by some reasonable means
prior to 60 days after the cessation.

  Moreover, your license from a particular copyright holder is
reinstated permanently if the copyright holder notifies you of the
violation by some reasonable means, this is the first time you have
received notice of violation of this License (for any work) from that
copyright holder, and you cure the violation prior to 30 days after
your receipt of the notice.

  Termination of your rights under this section does not terminate the
licenses of parties who have received copies or rights from you under
```

(continues on next page)

(continued from previous page)

```
this License.  If your rights have been terminated and not permanently
reinstated, you do not qualify to receive new licenses for the same
material under section 10.

  9. Acceptance Not Required for Having Copies.

  You are not required to accept this License in order to receive or
run a copy of the Program.  Ancillary propagation of a covered work
occurring solely as a consequence of using peer-to-peer transmission
to receive a copy likewise does not require acceptance.  However,
nothing other than this License grants you permission to propagate or
modify any covered work.  These actions infringe copyright if you do
not accept this License.  Therefore, by modifying or propagating a
covered work, you indicate your acceptance of this License to do so.

  10. Automatic Licensing of Downstream Recipients.

  Each time you convey a covered work, the recipient automatically
receives a license from the original licensors, to run, modify and
propagate that work, subject to this License.  You are not responsible
for enforcing compliance by third parties with this License.

  An "entity transaction" is a transaction transferring control of an
organization, or substantially all assets of one, or subdividing an
organization, or merging organizations.  If propagation of a covered
work results from an entity transaction, each party to that
transaction who receives a copy of the work also receives whatever
licenses to the work the party's predecessor in interest had or could
give under the previous paragraph, plus a right to possession of the
Corresponding Source of the work from the predecessor in interest, if
the predecessor has it or can get it with reasonable efforts.

  You may not impose any further restrictions on the exercise of the
rights granted or affirmed under this License.  For example, you may
not impose a license fee, royalty, or other charge for exercise of
rights granted under this License, and you may not initiate litigation
(including a cross-claim or counterclaim in a lawsuit) alleging that
any patent claim is infringed by making, using, selling, offering for
sale, or importing the Program or any portion of it.

  11. Patents.

  A "contributor" is a copyright holder who authorizes use under this
License of the Program or a work on which the Program is based.  The
work thus licensed is called the contributor's "contributor version".

  A contributor's "essential patent claims" are all patent claims
owned or controlled by the contributor, whether already acquired or
hereafter acquired, that would be infringed by some manner, permitted
by this License, of making, using, or selling its contributor version,
but do not include claims that would be infringed only as a
consequence of further modification of the contributor version.  For
purposes of this definition, "control" includes the right to grant
patent sublicenses in a manner consistent with the requirements of
this License.

  Each contributor grants you a non-exclusive, worldwide, royalty-free
```

(continues on next page)

(continued from previous page)

```
patent license under the contributor's essential patent claims, to
make, use, sell, offer for sale, import and otherwise run, modify and
propagate the contents of its contributor version.

  In the following three paragraphs, a "patent license" is any express
agreement or commitment, however denominated, not to enforce a patent
(such as an express permission to practice a patent or covenant not to
sue for patent infringement).  To "grant" such a patent license to a
party means to make such an agreement or commitment not to enforce a
patent against the party.

  If you convey a covered work, knowingly relying on a patent license,
and the Corresponding Source of the work is not available for anyone
to copy, free of charge and under the terms of this License, through a
publicly available network server or other readily accessible means,
then you must either (1) cause the Corresponding Source to be so
available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner
consistent with the requirements of this License, to extend the patent
license to downstream recipients.  "Knowingly relying" means you have
actual knowledge that, but for the patent license, your conveying the
covered work in a country, or your recipient's use of the covered work
in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

  If, pursuant to or in connection with a single transaction or
arrangement, you convey, or propagate by procuring conveyance of, a
covered work, and grant a patent license to some of the parties
receiving the covered work authorizing them to use, propagate, modify
or convey a specific copy of the covered work, then the patent license
you grant is automatically extended to all recipients of the covered
work and works based on it.

  A patent license is "discriminatory" if it does not include within
the scope of its coverage, prohibits the exercise of, or is
conditioned on the non-exercise of one or more of the rights that are
specifically granted under this License.  You may not convey a covered
work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment
to the third party based on the extent of your activity of conveying
the work, and under which the third party grants, to any of the
parties who would receive the covered work from you, a discriminatory
patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily
for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement,
or that patent license was granted, prior to 28 March 2007.

  Nothing in this License shall be construed as excluding or limiting
any implied license or other defenses to infringement that may
otherwise be available to you under applicable patent law.

  12. No Surrender of Others' Freedom.

  If conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot convey a
```

(continues on next page)

(continued from previous page)

```
covered work so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you may
not convey it at all.  For example, if you agree to terms that obligate you
to collect a royalty for further conveying from those to whom you convey
the Program, the only way you could satisfy both those terms and this
License would be to refrain entirely from conveying the Program.

  13. Use with the GNU Affero General Public License.

  Notwithstanding any other provision of this License, you have
permission to link or combine any covered work with a work licensed
under version 3 of the GNU Affero General Public License into a single
combined work, and to convey the resulting work.  The terms of this
License will continue to apply to the part which is the covered work,
but the special requirements of the GNU Affero General Public License,
section 13, concerning interaction through a network will apply to the
combination as such.

  14. Revised Versions of this License.

  The Free Software Foundation may publish revised and/or new versions of
the GNU General Public License from time to time.  Such new versions will
be similar in spirit to the present version, but may differ in detail to
address new problems or concerns.

  Each version is given a distinguishing version number.  If the
Program specifies that a certain numbered version of the GNU General
Public License "or any later version" applies to it, you have the
option of following the terms and conditions either of that numbered
version or of any later version published by the Free Software
Foundation.  If the Program does not specify a version number of the
GNU General Public License, you may choose any version ever published
by the Free Software Foundation.

  If the Program specifies that a proxy can decide which future
versions of the GNU General Public License can be used, that proxy's
public statement of acceptance of a version permanently authorizes you
to choose that version for the Program.

  Later license versions may give you additional or different
permissions.  However, no additional obligations are imposed on any
author or copyright holder as a result of your choosing to follow a
later version.

  15. Disclaimer of Warranty.

  THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
APPLICABLE LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT
HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY
OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM
IS WITH YOU.  SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF
ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

  16. Limitation of Liability.
```

(continues on next page)

(continued from previous page)

```
  IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS
THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF
DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
SUCH DAMAGES.

  17. Interpretation of Sections 15 and 16.

  If the disclaimer of warranty and limitation of liability provided
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximates
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.

                     END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
state the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <https://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.
```

(continues on next page)

(continued from previous page)

```
The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<https://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you
may consider it more useful to permit linking proprietary applications with
the library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.  But first, please read
<https://www.gnu.org/licenses/why-not-lgpl.html>.
```

7.2 Authors

The HETDEX collaboration:

- Francesco Montesano <montefra@mpe.mpg.de>
- Daniel Farrow <dfarrow@mpe.mpg.de>
- Jan Snigula <snigula@mpe.mpg.de>
- Jason R Ramsey <jasonramsey@utexas.edu>

7.3 `tcs_lib` release notes

7.3.1 Development version @ trunk

7.3.2 Version 0.5.0

- do not term context in `ZMQServer.close` and `TCSEvent.close` (issue #2536)
- add license GPL v3
- add `tcs_event.TCSDict` to provide basic structure necessary by TCS to parse events (issue #2535)
- `ZMQServer.send_tcs_event` set `__wire_time` (issue #2534)

7.3.3 Version 0.4.0

- add a `SafeTCSEvent` class (issue #2471)

7.3.4 Version 0.3.0

- `TCSEvent`: receive the full multipart event for better error handling (issue #2314)
- `TCSEvent`, `ZMQServer`: add possibility to explicitly pass a context (issue #2331)

7.3.5 Version 0.2.0

- Send a single event with the `ZMQServer` (issue #2061)
- Add `tcs_proxy` module (issue #2062)

7.4 Changelog

```
2018-05-17  Francesco Montesano  <montefra@mpe.mpg.de>

    * ReleaseNotes.rst: rename and update
    * doc/source/release_notes.rst: added. Resolves issue #2553
    * doc/source/index.rst: update
    * doc/source/conf.py: remove pyhetdex intersphinx (not necessary)

2018-05-17  Francesco Montesano  <montefra@mpe.mpg.de>

    * readthedocs.yml: added

2018-05-09  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: bump version to 0.5.0-post

2018-05-09  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: prepare for v0.5.0 release
    * ReleaseNotes.md: same

2018-05-09  Francesco Montesano  <montefra@mpe.mpg.de>

        * : merge ^/branches/conform_tcs_dict into ^/trunk

2018-05-09  Francesco Montesano  <montefra@mpe.mpg.de>

        * : merge ^/trunk into ^/branches/conform_tcs_dict

2018-04-30  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/string_helpers.py: function to return a string representation of
          time.time
    * tcs_lib/server.py: use it when setting __wire_time
    * tcs_lib/tcs_event.py: and when setting __data_time
    * tests/test_server.py: test the changes
    * tests/test_string_helpers.py: test the changes
    * tests/test_tcs_event.py: test the changes

2018-04-30  Francesco Montesano  <montefra@mpe.mpg.de>

    * ReleaseNotes.md: update release notes

2018-04-30  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/server.py: always set the __wire_time
```

```
    * tests/test_server.py: test it

2018-04-30  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/tcs_event.py: add dictionary with basic structure need by TCS.
          Resolves #2535
    * tests/test_tcs_event.py: test it

2018-04-27  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/server.py: Server.send_tcs_event set ``__wire_time``. Resolves
          issue #2534
    * tests/test_server.py: update the tests

2018-05-08  Francesco Montesano  <montefra@mpe.mpg.de>

    * LICENSE: added
    * doc/source/license.rst: properly renamed
    * doc/source/index.rst: update accordingly
    * tcs_lib/*.py: add copyright notice

2018-04-30  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/server.py: do not terminate context in ZMQServer.close. Resolves
          #2536
    * tcs_lib/tcs_event.py: do not terminate context in TCSEvent.close.␣
↪Resolves
          #2536
    * tests/test_server.py: update the tests
    * tests/test_tcs_event.py: same
    * ReleaseNotes.md: udate

2018-04-13  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: bump version to 0.4.0-post

2018-04-13  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: prepare for v0.4.0
    * ReleaseNotes.md: same
    * README.md: update

2018-04-13  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/tcs_event.py: add a SafeTCSEvent class. Resolves issue #2471
    * tests/test_tcs_event.py: test the new addition
    * ReleaseNotes.md: update

2018-02-13  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: bump version to 0.3.0-post

2018-02-13  Francesco Montesano  <montefra@mpe.mpg.de>
```

```
    * setup.py: prepare for release of v0.3.0
    * ReleaseNotes.md: same

2018-02-08  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/_tcs_proxy.py: fix documentation inconsistencies. Resolves issue
          #2265

2018-02-08  Francesco Montesano  <montefra@mpe.mpg.de>

    * : merge ^/branches/explicit_context/ into ^/trunk

2018-02-08  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: drop deprecated pytest-catchlog
    * tox.ini: same
    * tests/test_tcs_proxy.py: adapt to changes in pytest==3.4

2018-02-08  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/server.py: add possibility to explicitly pass a context to the
          ZMQServer (second part of issue #2331)
    * tests/test_server.py: test it

2018-02-07  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/tcs_event.py: add possibility to explicitly pass a context to␣
↪the
          TCSEvent (first part of issue #2331)
    * tests/test_tcs_event.py: test it

2018-01-31  Francesco Montesano  <montefra@mpe.mpg.de>

        * : merge ^/branches/recv_multipart into ^/trunk

2018-01-31  Francesco Montesano  <montefra@mpe.mpg.de>

    * ReleaseNotes.md: update

2018-01-30  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/tcs_event.py: remove ``unnecessary while True``.

2018-01-29  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/tcs_event.py: get the full multipart event and then convert to
          json. Also improve error messages for easier debugging. Resolves␣
↪issue
          #2314
    * tcs_lib/errors.py: add necessary exceptions
    * tests/test_tcs_event.py: test the changes

2017-10-17  Francesco Montesano  <montefra@mpe.mpg.de>
```

```
    * setup.py: bump version to 0.2.0-post

2017-10-17  Francesco Montesano  <montefra@mpe.mpg.de>

    * ReleaseNotes.md: add release information
    * setup.py: set version to 0.2.0

2017-10-16  Francesco Montesano  <montefra@mpe.mpg.de>

    * : merge ^/branches/tcs_proxy into ^/trunk

2017-10-16  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/_tcs_proxy.py: add a method to clear the proxies
    * tests/test_tcs_proxy.py: test it
    * doc/source/tcs_proxy.rst: document it

2017-10-13  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/_tcs_proxy.py: add ``errors`` attribute
    * tests/test_tcs_proxy.py: test it
    * doc/source/tcs_proxy.rst: document it
    * doc/Makefile: use random port and open the browser

2017-10-10  Francesco Montesano  <montefra@mpe.mpg.de>

    * doc/source/tcs_proxy.rst: add documentation about tcs_lib.tcs_proxy
    * tcs_lib/_tcs_proxy.py: add {name} attribute to the docs

2017-10-06  Francesco Montesano  <montefra@mpe.mpg.de>

        * tcs_lib/_tcs_proxy.py: reimplement tcs_proxy from ocd. Issue #2062
    * tcs_lib/tcs_proxy.py: public interface to the tcs_proxy
    * tests/test_tcs_proxy.py: test the tcs proxy
    * doc/source/tcs_proxy.rst: add the documentation
    * doc/source/index.rst: add to the index
    * pytest.ini: ignore tcs_lib/tcs_proxy.py
    * tox.ini: add dependencies
    * setup.py: same

2017-09-28  Francesco Montesano  <montefra@mpe.mpg.de>

    * : merge ^/branches/send_event into ^/trunk

2017-09-28  Francesco Montesano  <montefra@mpe.mpg.de>

    * tcs_lib/server.py: add send_event and send_tcs_event methods. Resolves
          issue #2061
    * tests/test_server.py: add tests
    * tests/conftest.py: update the fixtures
    * tests/test_tcs_event.py: same

2017-07-05  Francesco Montesano  <montefra@mpe.mpg.de>
```

```
    * setup.py: bump version to 0.1.0-pos
        * : ignore dist and build directories

2017-07-05  Francesco Montesano  <montefra@mpe.mpg.de>

    * setup.py: set version to 0.1.0 ahead of release

2017-07-04  Francesco Montesano  <montefra@mpe.mpg.de>

    * doc/source/authors.rst: added
    * doc/source/changelog.rs: added
    * doc/source/contributions.rs: added
    * doc/source/index.rst: update documentation, add new section
    * doc/source/install.rs: added
    * doc/source/licence.rs: added
    * doc/source/tcs_event.rst: update
    * doc/source/tcs_replay.rst: update
    * doc/source/todos.rs: added
    * doc/source/zmq_server.rst: update
    * setup.py: fix test dependences
    * tcs_lib/errors.py: update docstrings
    * tcs_lib/server.py: same
    * tcs_lib/string_helpers.py: same
    * tcs_lib/tcs_event.py: same
    * tox.ini: fix project name and doc

2017-07-04  Francesco Montesano  <montefra@mpe.mpg.de>

    * doc: added

2017-07-04  Francesco Montesano  <montefra@mpe.mpg.de>

    * .: ignore byproducts
    * ReleaseNotes.md: remove OCD references
    * setup.py: same
    * tcs_lib/__init__.py: same
    * tcs_lib/errors.py: same
    * tcs_lib/server.py: same
    * tcs_lib/tcs_event.py: same
    * tcs_lib/tcs_replay.py: same
    * tests: add and adapt tests
    * setup.cfg: tcs_lib/replay.py doesn't exist anymore

2017-07-04  Francesco Montesano  <montefra@mpe.mpg.de>

        * import code
```

7.5 TODO

Todo: add timeout?

original entry

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

Index

Symbols

B

C

D

E

I

M

N

P

S

T

Z