PyTangoArchiving

https://github.com/tango-controls/PyTangoArchiving

```
$ pip install PyTangoArchiving
```

Sergi Rubio Manrique, Workshop at ICALEPCS'19

Archiving Clients: PyTangoArchiving API

Reasons why we developed a python API for Tango archiving system:

- Need to manage distributed configuration for thousands of attributes from the python shell.
- To plot archived and online data together, using our regular Taurus GUI's instead of having a separated application like Mambo or eGiga.
- To mix data from different databases in a single query (by schema and/or date of retrieval).
- To provide scientists with scripts to extract archiving data and use it in scripts, macros or façades (fandango.DynamicDS, PyAttributeProcessor)..
- To automate table maintenance and decimation (using fandango.WorkerDS)
- To notify problems with archiving (via PyAlarm)

Archiving Clients: PyTangoArchiving API

For each schema (HDB, TDB, HDB++, Snap, ...) there are three API levels:

- PyTangoArchiving.**ArchivingDB**: a python object mapping the structure of the MySQL database, all queries are done within this object.
- PyTangoArchiving.**ArchivingAPI**: the python object that, on top of the database, manages all interactions with archiving device servers (start/stop of attributes, create new devices, configure their properties).
- PyTangoArchiving.**Reader**: a simplified API for clients, it access the database and provides methods for check (is_attribute_archived(attr)) and query values (get_attribute_values(att,t0,t1)) and data interpolation (decimate_values(array)).

While all three classes have an "schema" atttribute; the Reader can be called as "universal", thus returning a polymorphic object with access to multiple databases.

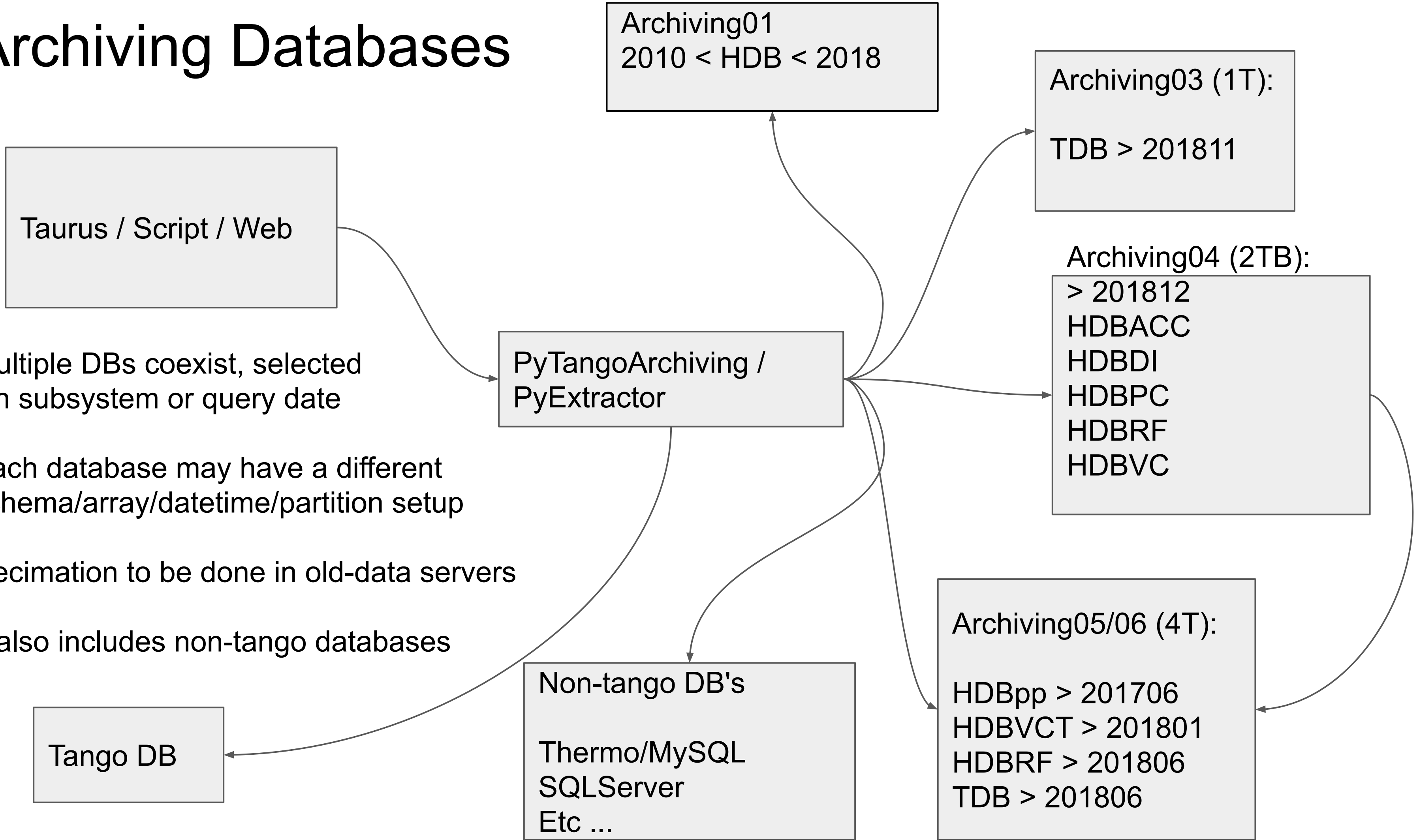
Archiving Databases

Multiple DBs coexist, selected
On subsystem or query date

Each database may have a different
schema/array/datetime/partition setup

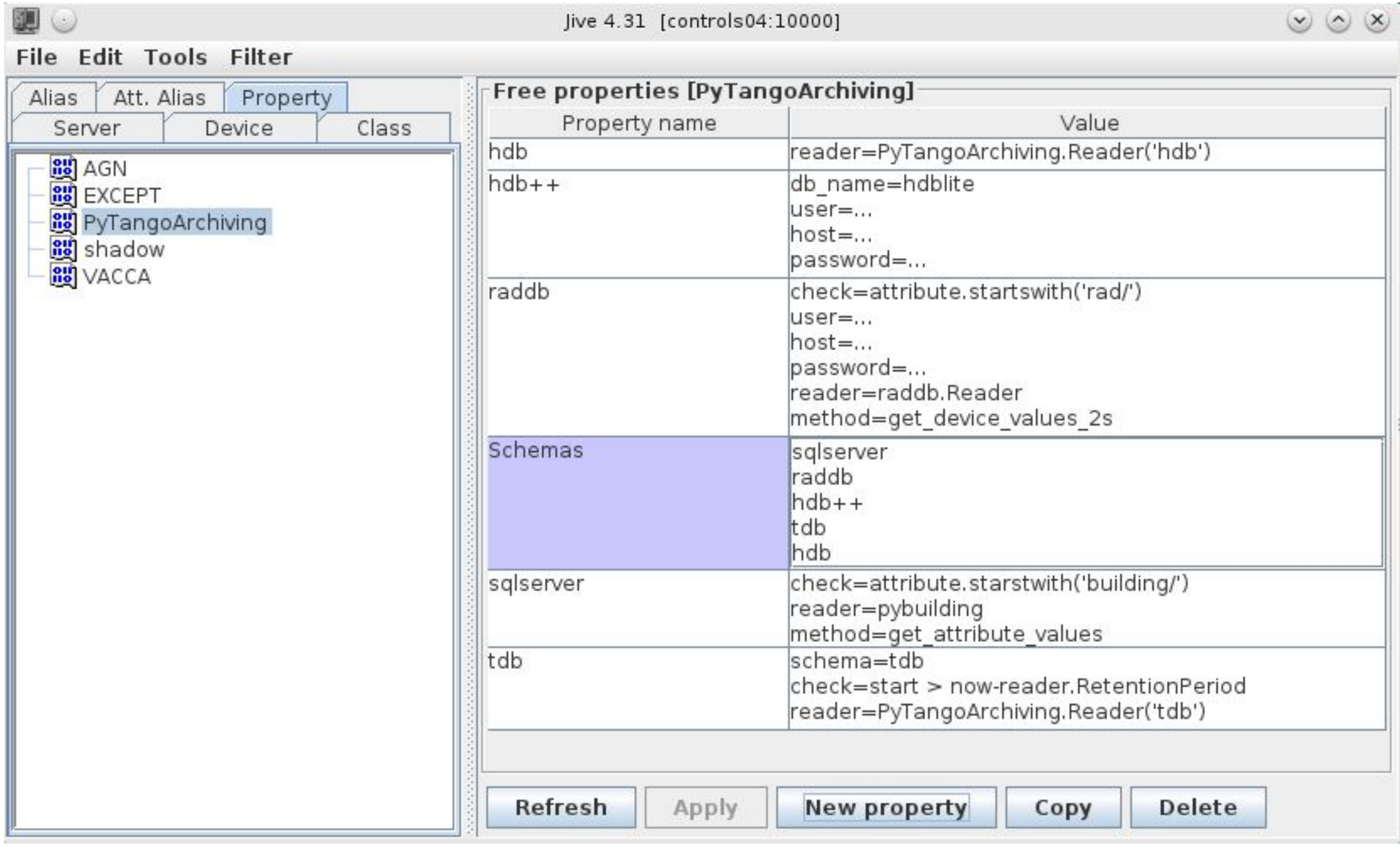
Decimation to be done in old-data servers

It also includes non-tango databases



Multiple Archiving Schemas

Tango properties are used to make PyTangoArchiving aware of the different databases available.



Free properties [PyTangoArchiving]

Property name	Value
hdb	reader=PyTangoArchiving.Reader('hdb')

hdb++	db_name=hdblite
	user=...
	host=...
	password=...

raddb	check=attribute.startswith('rad/')
	user=...
	host=...
	password=...
	reader=raddb.Reader
	method=get_device_values_2s

Schemas	sqlserver raddb hdb++ tdb hdb

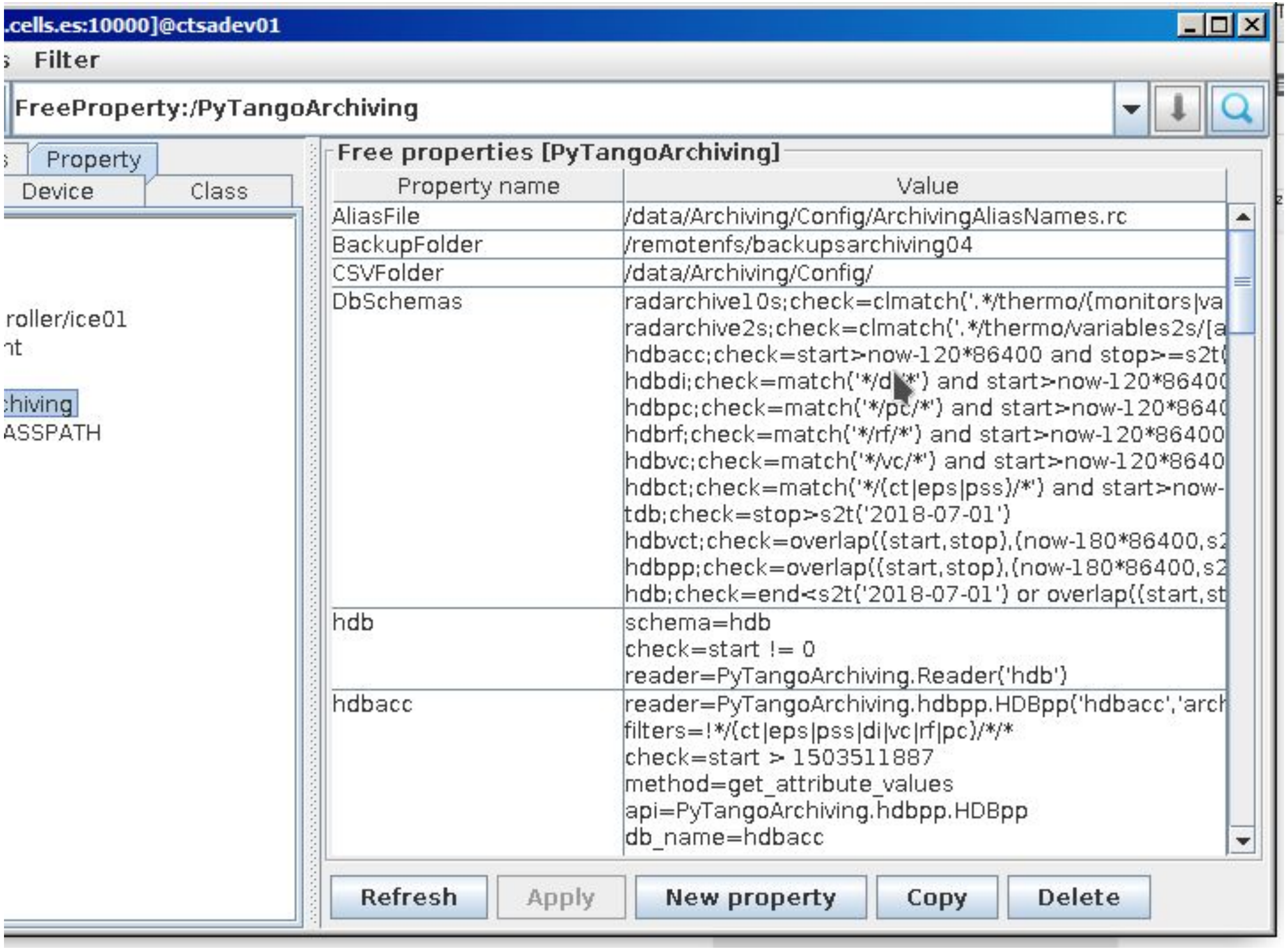
sqlserver	check=attribute.starstwith('building/')
	reader=pybuilding
	method=get_attribute_values

tdb	schema=tdb
	check=start > now-reader.RetentionPeriod
	reader=PyTangoArchiving.Reader('tdb')

Multiple Schemas

Functionality has been extended to load Archiving readers as plugins on runtime.

In addition, a fast check has been added to avoid loading modules if not necessary..



Free properties [PyTangoArchiving]

Property name	Value
AliasFile	/data/Archiving/Config/ArchivingAliasNames.rc
BackupFolder	/remotenfs/backupsarchiving04
CSVFolder	/data/Archiving/Config/
DbSchemas	radarchive10s;check=clmatch('.*/thermo/(monitors|va
	radarchive2s;check=clmatch('.*/thermo/variables2s/[a
	hdbacc;check=start>now-120*86400 and stop>=s2t(
	hdbdi;check=match('*/di/*') and start>now-120*86400
	hdbpc;check=match('*/pc/*') and start>now-120*8640
	hdbrf;check=match('*/rf/*') and start>now-120*86400
	hdbvc;check=match('*/vc/*') and start>now-120*8640
	hdbct;check=match('*/(ct|eps|pss)/*') and start>now-
	tdb;check=stop>s2t('2018-07-01')
	hdbvct;check=overlap((start,stop),(now-180*86400,s2
	hdbpp;check=overlap((start,stop),(now-180*86400,s2
	hdb;check=end<s2t('2018-07-01') or overlap((start,st
hdb	schema=hdb
	check=start != 0
	reader=PyTangoArchiving.Reader('hdb')
hdbacc	reader=PyTangoArchiving.hdbpp.HDBpp('hdbacc','arch
	filters=!*/(ct|eps|pss|di|vc|rf|pc)/*/*
	check=start > 1503511887
	method=get_attribute_values
	api=PyTangoArchiving.hdbpp.HDBpp
	db_name=hdbacc

PyTangoArchiving: main methods

api.start_archiving([attributes] , {'mode':[period,range] })

api.stop_archiving([attributes])

Periodic archived is supported for legacy and HDB++ archiving (we developed a PyHdbppPeriodicArchiver for that purpose).

When adding attributes to HDB++ without specifying a periodic mode, events pushed by device code are assumed.

It means that, unlike HdbppConfigurator, attribute events must be crosschecked separately (using

PyTangoArchiving: main methods

A Reader object can be created against a database or the whole system.

PyTangoArchiving.Reader.is_attribute_archived:

Returns the schemas actually archiving a given attribute (by preference)

```
rd = PyTangoArchiving.Reader()

rd.is_attribute_archived('sr/di/dcct/averagecurrent')

    ['hdbacc', 'hdbdi', 'tdb']
```

PyTangoArchiving: main methods

A Reader object can be created against a database or the whole system.

PyTangoArchiving.Reader(schema).get_attribute_values():

Returns the values actually archived by a given attribute (by preference)

```
attr = 'sr/di/dcct/averagecurrent'

values = PyTangoArchiving.Reader().get_attribute_values(attr, '-1h')

values = PyTangoArchiving.Reader('tdb').get_attribute_values(attr, '2019-09-01', '2019-10-01')
```

A Reader object will return values always as a [(timestamp,value,quality)] list, independently of the database that is providing the data.

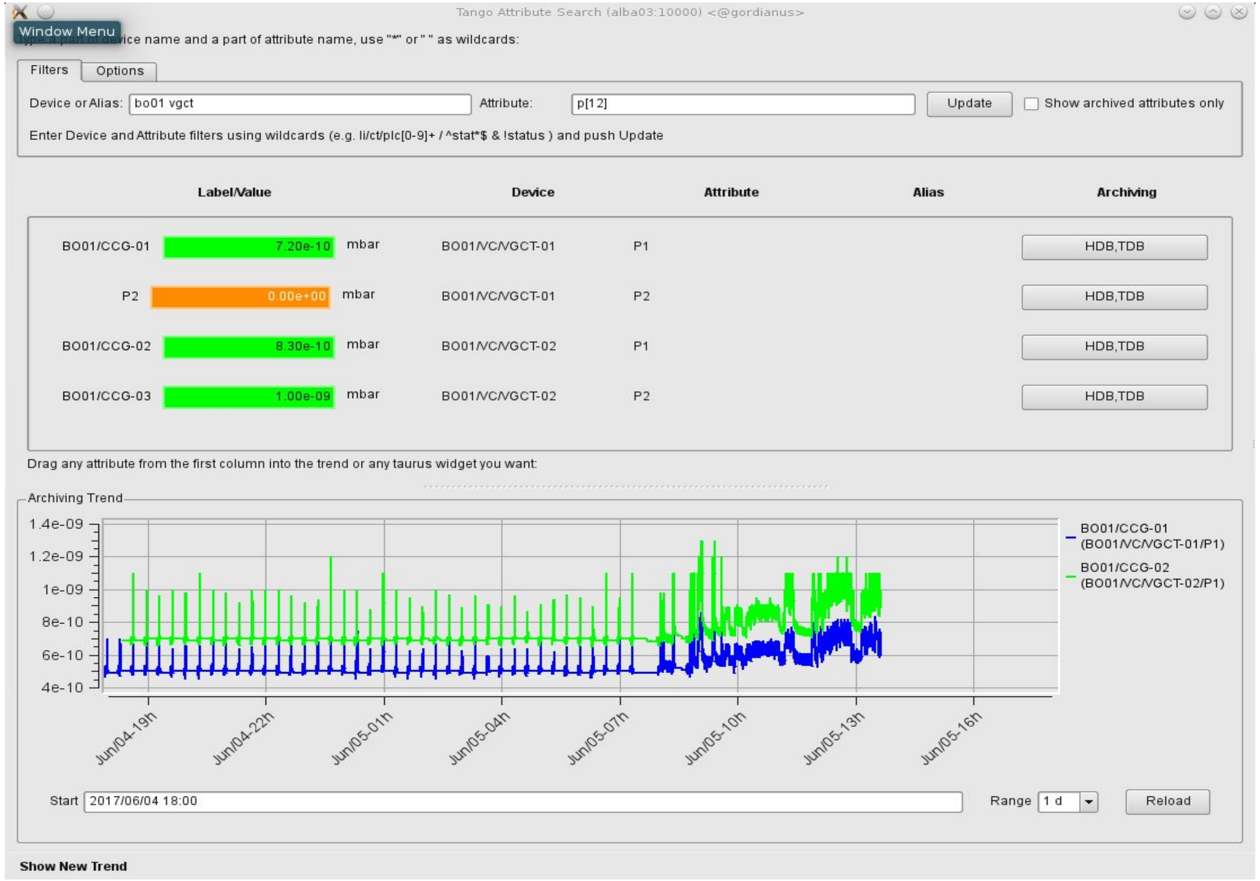
Taurus Clients

A PyQt user interface is provided for exploring archived attributes and plot/extract
Saved data.

Usage is PyQwt is currently deprecated, a new client based on PyQtGraph is under development

Drag and drop of attribute names from any taurus application (even from another process) is suported.

As Taurus is fully modular, archiving can be added to any UI just adding the TaurusTrend widget.



Tango Attribute Search (alba03:10000) <@gordianus>

Type a part of device name and a part of attribute name, use "*" or " " as wildcards:

Device or Alias: bo01 vgct Attribute: p[12] Update ☐ Show archived attributes only

Enter Device and Attribute filters using wildcards (e.g. li/ct/plc[0-9]+ / ^stat*$ & !status) and push Update

Label/Value			Device	Attribute	Alias	Archiving
BO01/CCG-01	7.20e-10	mbar	BO01/VC/VGCT-01	P1		HDB,TDB
P2	0.00e+00	mbar	BO01/VC/VGCT-01	P2		HDB,TDB
BO01/CCG-02	8.30e-10	mbar	BO01/VC/VGCT-02	P1		HDB,TDB
BO01/CCG-03	1.00e-09	mbar	BO01/VC/VGCT-02	P2		HDB,TDB

Drag any attribute from the first column into the trend or any taurus widget you want:

Archiving Trend

BO01/CCG-01 (BO01/VC/VGCT-01/P1)
BO01/CCG-02 (BO01/VC/VGCT-02/P1)

Start 2017/06/04 18:00 Range 1 d Reload

Show New Trend

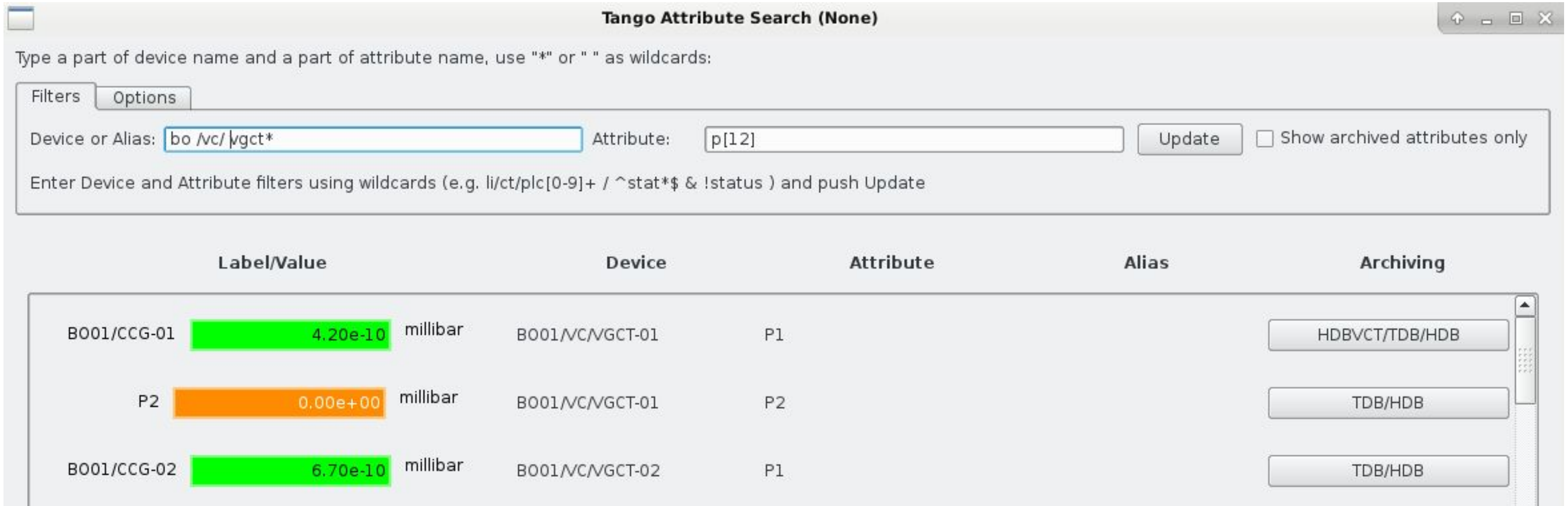
ArchivingBrowser: A Tango browser

This application (aka taurus finder) provides a toolbar for searching devices in tango/archiving database.

The first field in the search bar is for the device name, you can use "*" as wildcard for searching devices

The second field will filter the attributes for each device, common regular expressions characters can be used ([] ? $ *). The " " space character is used in both cases as wildcard.

Schemas preference can be modified from the right side button.



Type a part of device name and a part of attribute name, use "*" or " " as wildcards:

Device or Alias: bo /vc/ vgct*

Attribute: p[12]

☐ Show archived attributes only

Enter Device and Attribute filters using wildcards (e.g. li/ct/plc[0-9]+ / ^stat*$ & !status) and push Update

Label/Value			Device	Attribute	Alias	Archiving
B001/CCG-01	4.20e-10	millibar	B001/VC/VGCT-01	P1		HDBVCT/TDB/HDB
P2	0.00e+00	millibar	B001/VC/VGCT-01	P2		TDB/HDB
B001/CCG-02	6.70e-10	millibar	B001/VC/VGCT-02	P1		TDB/HDB

PyExtractor + WebTornadoDS Reports

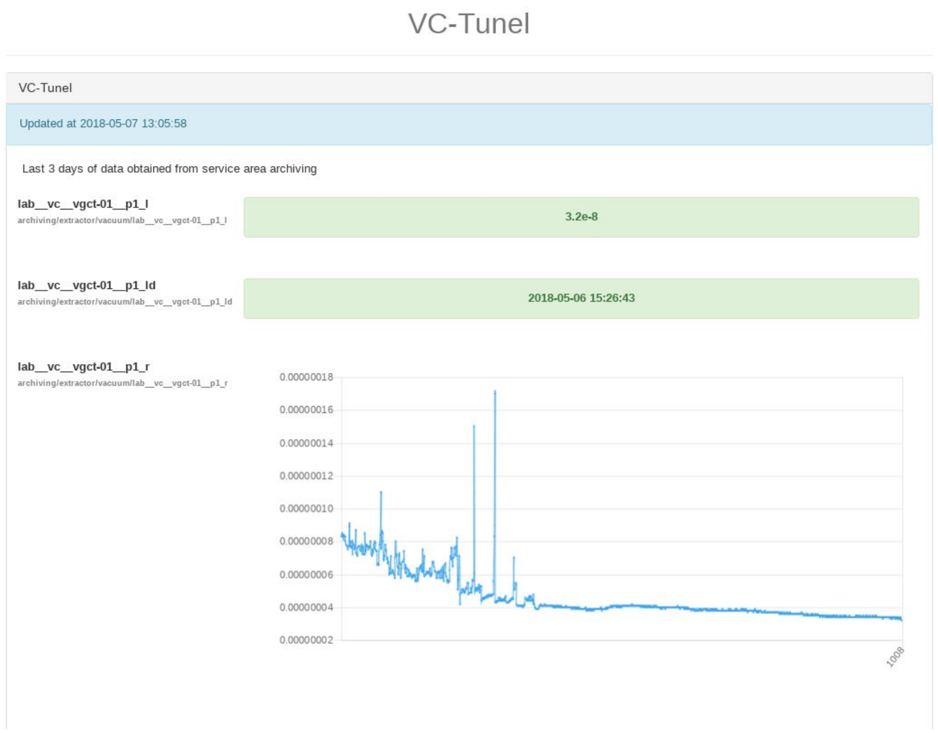
WebTornadoDS generates new web reports on demand. PyExtractor is a device server included in PyTangoArchiving.

The device provides a web frontend to add new attributes to a cfg file.

Via **PyExtractor**, the DS will query the attributes to the archiving system (from a machine outside TangoCS).

Data will be exported to a .json file and later loaded by the web front-end

To completely isolate CS and WWW, visualization/exporting can be separated in two instances, only sharing the cfg/json files



VC-Tunel

VC-Tunel

Updated at 2018-05-07 13:05:58

Last 3 days of data obtained from service area archiving

lab__vc__vgct-01__p1_I archiving/extractor/vacuum/lab__vc__vgct-01__p1_I	3.2e-8
lab__vc__vgct-01__p1_Id archiving/extractor/vacuum/lab__vc__vgct-01__p1_Id	2018-05-06 15:26:43
lab__vc__vgct-01__p1_r archiving/extractor/vacuum/lab__vc__vgct-01__p1_r	

archiving2csv

export archived data to a format that can be read by excel or matlab

if you don't specify any database, it will be selected automatically, don't worry about it unless you need it

archiving2csv

```
$ archiving2csv [--resolution=X(s)] [--hdb] [--tdb] [--modes] ["--arrsep=,"][attributes] ["Y-m-d H:M"] ["Y-m-d H:M"] filename.csv

--hdb/tdb : choose database
--modes : export modes instead of values
--config : same, in "human format"
--arrsep/--no-sep : default separator between arrays values
--sep : separator between columns
--linesep : character between lines
--resolution : force periodicity of values to a fix period
--noheader : do not include headers
--nodate : do not include datetime
--noepoch : do not include epochs
```

archiving2csv

```
$ archiving2csv test/acc/ps-clic-01/voltage test/acc/ps-clic-02/voltage 2018-02-21 2018-02-23 /tmp/test2.csv
```

```
 $ head /tmp/test2.csv

date   time   test/acc/ps-clic-01/voltage    test/acc/ps-clic-02/voltage

2018-02-21_15:13:00.000 1519222380      2081.46 -1006.68

2018-02-21_15:13:01.000 1519222381      2081.46 -1006.68

2018-02-21_15:13:02.000 1519222382      2081.46 -1006.68

2018-02-21_15:13:03.000 1519222383      2081.46 -1006.68

2018-02-21_15:13:04.000 1519222384      2081.46 -1006.68
```

ctarchiving

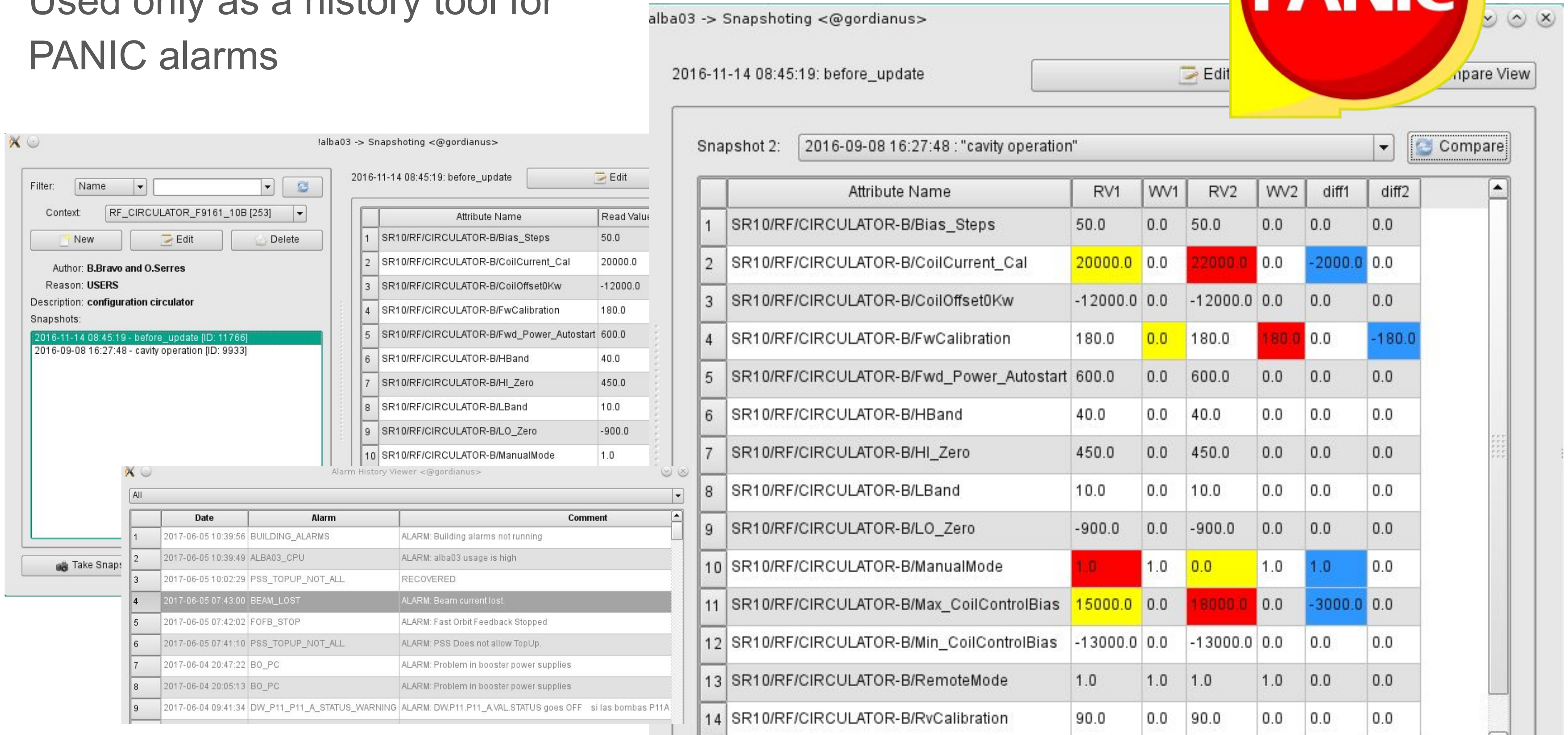
```
operator@caligula:~$ ctarchiving --help
```

script to manage Archiving services, from it you can launch the ArchivingBrowser or archiving2csv clients

```
Usage:

  ctarchiving --help

  ctarchiving --load <filename.csv> <hdb/tdb> [force] ; adds attributes to archiving

  ctarchiving --parse <filename.csv>

  ctarchiving --check <filename.csv/attrs> <hdb/tdb> [force] ; checks if attributes from .csv are archived

  ctarchiving --start/stop <attributes> <hdb/tdb> ; starts/stops attribute archiving

  ctarchiving --export [--resolution=X(s)] [--hdb] [--tdb] [--modes] [attributes] ["Y-m-d H:M"] ["Y-m-d H:M"]
filename.csv

  ctarchiving --search <device/attributes> ; searchs for matching archived attributes, returns configuration

  ctarchiving --search2 <device/attributes> ; searchs only for active attributes

  ctarchiving --gui [--range YYYY/MM/DD,XXh] <device/attributes> ; shows Archiving GUI, TaurusFinder
```

Discontinued tools, Snap

Used only as a history tool for PANIC alarms



!alba03 -> Snapshoting <@gordianus>

Filter: Name

Context: RF_CIRCULATOR_F9161_10B [253]

New | Edit | Delete

Author: **B.Bravo and O.Serres**

Reason: **USERS**

Description: **configuration circulator**

Snapshots:

2016-11-14 08:45:19 - before_update [ID: 11766]

2016-09-08 16:27:48 - cavity operation [ID: 9933]

Take Snaps

2016-11-14 08:45:19: before_update

	Attribute Name	Read Valu
1	SR10/RF/CIRCULATOR-B/Bias_Steps	50.0
2	SR10/RF/CIRCULATOR-B/CoilCurrent_Cal	20000.0
3	SR10/RF/CIRCULATOR-B/CoilOffset0Kw	-12000.0
4	SR10/RF/CIRCULATOR-B/FwCalibration	180.0
5	SR10/RF/CIRCULATOR-B/Fwd_Power_Autostart	600.0
6	SR10/RF/CIRCULATOR-B/HBand	40.0
7	SR10/RF/CIRCULATOR-B/HI_Zero	450.0
8	SR10/RF/CIRCULATOR-B/LBand	10.0
9	SR10/RF/CIRCULATOR-B/LO_Zero	-900.0
10	SR10/RF/CIRCULATOR-B/ManualMode	1.0

Alarm History Viewer <@gordianus>

All

	Date	Alarm	Comment
1	2017-06-05 10:39:56	BUILDING_ALARMS	ALARM: Building alarms not running
2	2017-06-05 10:39:49	ALBA03_CPU	ALARM: alba03 usage is high
3	2017-06-05 10:02:29	PSS_TOPUP_NOT_ALL	RECOVERED
4	2017-06-05 07:43:00	BEAM_LOST	ALARM: Beam current lost.
5	2017-06-05 07:42:02	FOFB_STOP	ALARM: Fast Orbit Feedback Stopped
6	2017-06-05 07:41:10	PSS_TOPUP_NOT_ALL	ALARM: PSS Does not allow TopUp.
7	2017-06-04 20:47:22	BO_PC	ALARM: Problem in booster power supplies
8	2017-06-04 20:05:13	BO_PC	ALARM: Problem in booster power supplies
9	2017-06-04 09:41:34	DW_P11_P11_A_STATUS_WARNING	ALARM: DW.P11.P11_A.VAL.STATUS goes OFF si las bombas P11A

alba03 -> Snapshoting <@gordianus>

2016-11-14 08:45:19: before_update

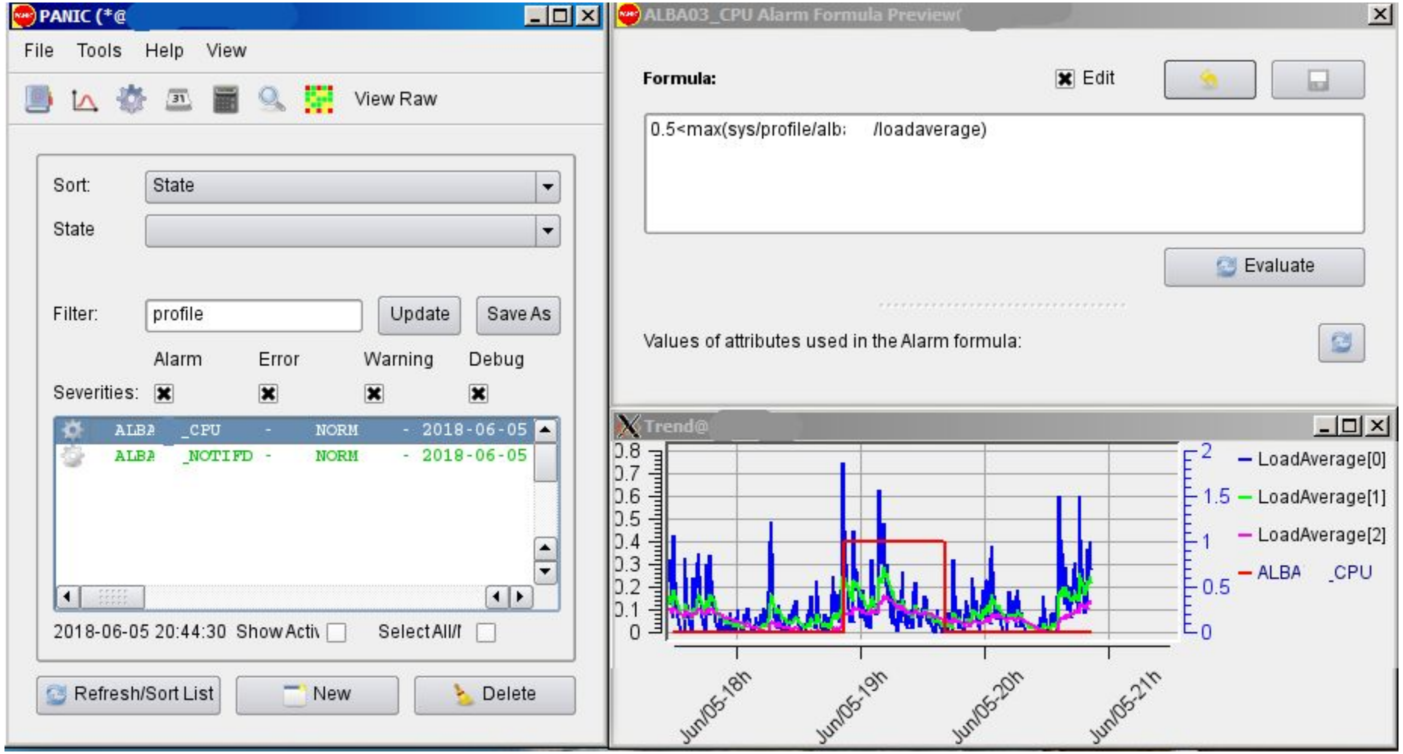
Edit | npare View

Snapshot 2: 2016-09-08 16:27:48 : "cavity operation" | Compare

	Attribute Name	RV1	WV1	RV2	WV2	diff1	diff2
1	SR10/RF/CIRCULATOR-B/Bias_Steps	50.0	0.0	50.0	0.0	0.0	0.0
2	SR10/RF/CIRCULATOR-B/CoilCurrent_Cal	20000.0	0.0	22000.0	0.0	-2000.0	0.0
3	SR10/RF/CIRCULATOR-B/CoilOffset0Kw	-12000.0	0.0	-12000.0	0.0	0.0	0.0
4	SR10/RF/CIRCULATOR-B/FwCalibration	180.0	0.0	180.0	180.0	0.0	-180.0
5	SR10/RF/CIRCULATOR-B/Fwd_Power_Autostart	600.0	0.0	600.0	0.0	0.0	0.0
6	SR10/RF/CIRCULATOR-B/HBand	40.0	0.0	40.0	0.0	0.0	0.0
7	SR10/RF/CIRCULATOR-B/HI_Zero	450.0	0.0	450.0	0.0	0.0	0.0
8	SR10/RF/CIRCULATOR-B/LBand	10.0	0.0	10.0	0.0	0.0	0.0
9	SR10/RF/CIRCULATOR-B/LO_Zero	-900.0	0.0	-900.0	0.0	0.0	0.0
10	SR10/RF/CIRCULATOR-B/ManualMode	1.0	1.0	0.0	1.0	1.0	0.0
11	SR10/RF/CIRCULATOR-B/Max_CoilControlBias	15000.0	0.0	18000.0	0.0	-3000.0	0.0
12	SR10/RF/CIRCULATOR-B/Min_CoilControlBias	-13000.0	0.0	-13000.0	0.0	0.0	0.0
13	SR10/RF/CIRCULATOR-B/RemoteMode	1.0	1.0	1.0	1.0	0.0	0.0
14	SR10/RF/CIRCULATOR-B/RvCalibration	90.0	0.0	90.0	0.0	0.0	0.0

PANIC

PANIC Alarms Logging with HDB++



```
ALBA   _CPU     -   NORM   - 2018-06-05
ALBA   _NOTIFD  -   NORM   - 2018-06-05
```

```
0.5<max(sys/profile/alb:    /loadaverage)
```

Discontinued Tools, Cassandra

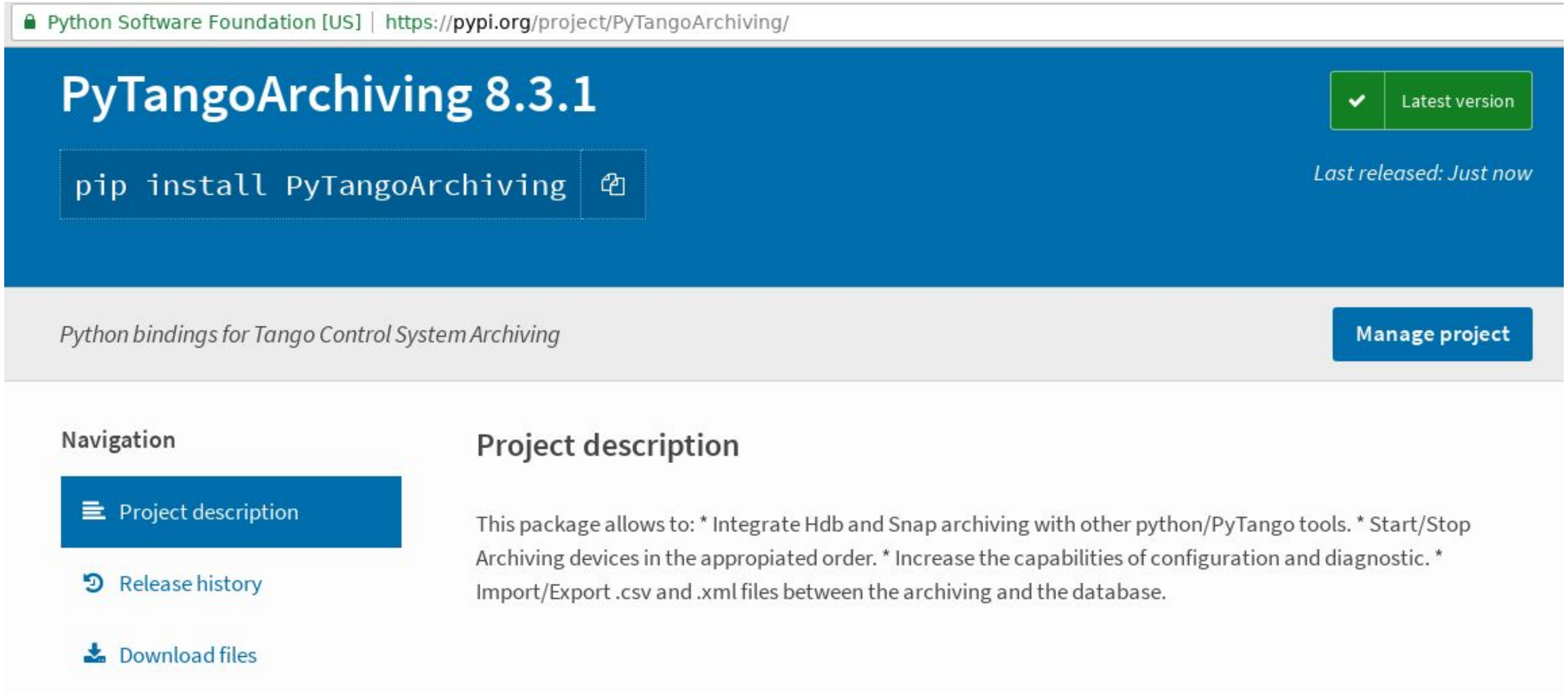
Support for extracting from Cassandra was developed as a separated module.

As ESRF drop its interest in Cassandra, the project has been discontinued and not merged.

Therefore, PyTangoArchiving is currently a Mariadb-only project.

But, the pluggable nature of schemas makes the project still valid (if anybody is interested in finish it).

PyTangoArchiving 8


Python Software Foundation [US] | https://pypi.org/project/PyTangoArchiving/

PyTangoArchiving 8.3.1

```
pip install PyTangoArchiving
```

Last released: Just now

Python bindings for Tango Control System Archiving

Project description

This package allows to: * Integrate Hdb and Snap archiving with other python/PyTango tools. * Start/Stop Archiving devices in the appropiated order. * Increase the capabilities of configuration and diagnostic. * Import/Export .csv and .xml files between the archiving and the database.