

TINE Release 5.x.x News

(Nov 13, 2019: inching toward perfection ...)

“Remember: *Only the dead fish go with the flow ...*”



[Release 5.1.2]

- Fixes, Features, and Issues ...
 - the *powers that be* have decided that adweb.desy.de shall now be winweb.desy.de
 - <http://tine.desy.de> is (was) auto-redirected to <https://adweb.desy.de/mcs/tine>
 - this breaks not only a good many cached links but necessitates :
 - replacing *all internal references* within the tine web site (done)
 - replacing *all help references* in the TINE Studio applications (Jaka is doing it/has done it)
 - and probably something else someone will notice down the line ...



[Release 5.1.2]

- Fixes, Features, and Issues ...
 - **C-Lib Client side:**
 - always **suppress a link queue** if linked to a **tagged structure**
 - Leads to problems (*think crash*) if the structure is *not internally aligned* and an incoming linked gets queued.
 - ignore **.bak** files when **SRVLOGFILE** accesses a non **.log** file. (e.g. **.csv**)
 - i.e. does not append contents of last rotated file ..
 - this stock property can be used to read/write any text file ...



[Release 5.1.2]

- Fixes, Features, and Issues ...
 - **C-Lib:** new functions
 - **SetDieFunction**(void (*)(const char *)fcn)
 - will call the provided function with the reason for exiting as a message string
 - **RemoveDevice**(char *eqm,char *device)
 - removes device from device list
 - does not remove the device slot or affect device capacity.
 - **RemoveProperty**(char *eqm,char *property)
 - Removes property from the registered properties list

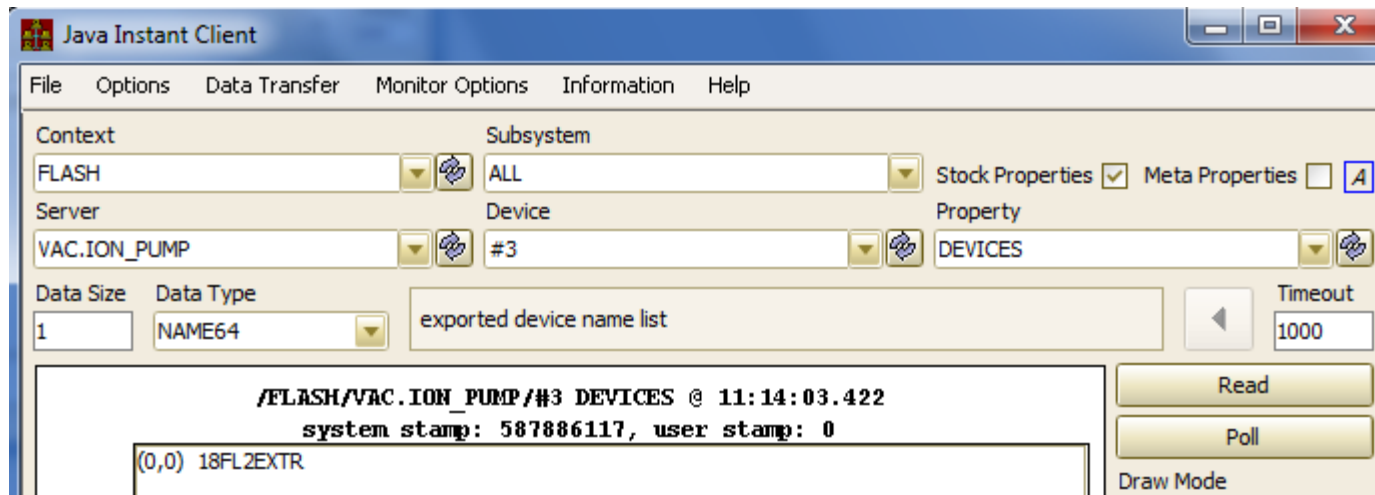
[Release 5.1.2]

- Fixes, Features, and Issues ...
 - **Remove Property Issue:** the buffered services windows DLL **missed** the *build all* and was not updated until later ...
 - 3-week period from mid-September.
 - e.g. LabView Servers were reporting `illegal_format` on any call ...
 - Apologies to Uwe R. and Uwe H.



Release 5.1.2

- Fixes, Features, and Issues ...
 - **removeDevice** issue: (**Java side**)
 - *long standing java bug was also fixed* :
 - query to '**DEVICES**' with device name target #3 should return the device name associated with device #3.
 - Was always returning device name for #0 !
 - exposed issues related to device name registration strategies fixed by Tobias ...



Release 5.1.2

- Fixes, Features, and Issues ...
 - New Stock Property : “**PROPERTY.STATS**”

The screenshot shows the Java Instant Client interface with the following configuration:

- Context: FLASH
- Subsystem: ALL
- Server: VAC.ION_PUMP
- Device: *
- Property: PROPERTY.STATS
- Data Size: 100
- Data Type: NAME64I
- exported property call statistics

The main display area shows the following output:

```
/FLASH/VAC.ION_PUMP/* PROPERTY.STATS @ 16:59:50.339
system stamp: 587229580, user stamp: 0
(0,0) [AB_IP_P_OK, 0]
(0,1) [AB_IP_P_THR, 24007]
(0,2) [AB_NAME, 2]
(0,3) [AUTO_ENABLE, 2]
(0,4) [AUTO_OFF, 0]
(0,5) [AUTO_PRE_OFF, 0]
(0,6) [CAN, 2]
(0,7) [CAN_ACTIVE, 0]
(0,8) [CAN_ALIVE, 0]
(0,9) [CAN_BAUDRATE, 0]
(0,10) [CAN_CONDITION, 0]
(0,11) [CAN_DEBUG_ID, 0]
(0,12) [CAN_HWVER, 0]
(0,13) [CAN_MANU, 0]
(0,14) [CAN_RESET, 0]
(0,15) [CAN_START, 0]
(0,16) [CAN_STOP, 0]
(0,17) [CAN_SWVER, 0]
(0,18) [CAPACITY, 0]
(0,19) [CARRY_GP_2PKLEFT_MASK, 2]
(0,20) [CARRY_GP_2PKLEFT_TRIGGER, 0]
(0,21) [CDI_ADR, 0]
(0,22) [CDI_HOST, 0]
```

Settings: UDP, Timer | Suppress Query Properties

Last request: 16:59:51.259 (8 ms)

number of times a property has been called since server startup ...

property **AB_IP_P_THR** seems to be popular ...

[Release 5.1.2]

- Fixes, Features, and Issues ...
 - Offer the doocs data type **CF_GSPECTRUM** (to accommodate archive requests)
 - like a spectrum with more header information.
 - spectrum array can be a doubly dimensioned array.
 - Arthur will incorporate it into the doocs2tine layer when TINE Release 5 is integrated into doocs.



[Release 5.1.2]

- Fixes, Features, and Issues ...
 - Stock Property **DEVLOCATION** now accepts WRITE-only transactions
 - jddd does not offer WRITE/READ and ...
 - it's silly to *force* the caller READ back what he is writing if he doesn't want to ...

[Release 5.1.2]

- Remaining Release 5 vs. Release 4 issues ?
 - **Case study A:**
 - A Rel. 4 server rolls *up* to Rel. 5 *with running clients with multiple links.*
 - If **downtime long enough**, a Rel. 5 java client *learns* that the server now speaks Rel. 5 but due to a bug (fixed!) it used a *Rel. 5 packet header with a Rel. 4 request header*
 - server did **NOT** like this!
 - Server now protects against this (should there still be a java client out there that didn't get a newer jar file)!



[Release 5.1.2]

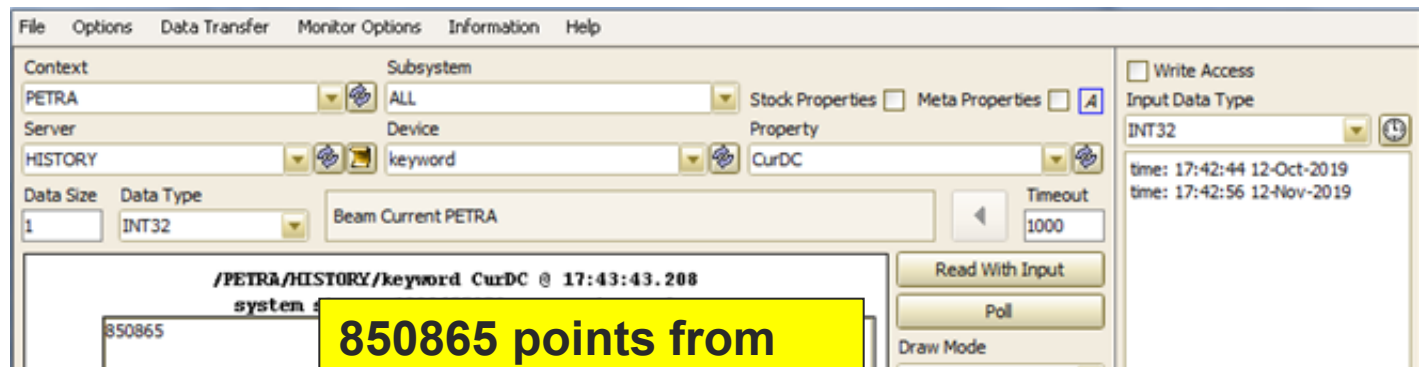
- Remaining Release 5 vs. Release 4 issues ?
 - **Case study B:**
 - a Rel. 5 server rolls *back* to Rel. 4 *with running clients with multiple links*.
 - clients begin to receive **illegal_protocol** from the server and then *react* by *decrementing the communication protocol* and :
 - *for any and all link associated with the target FEC.*
 - if there were **multiple links** (especially in single-threaded mode) then the next **illegal_protocol** continues to decrement until there's no option but to close the link!

[Archiving & Archive Retrieval]

- both **Central** and **Local Archiving** offer **APIs** for :
 - determining the *number of stored points in an interval*
 - specifying a desired *sampling raster*
 - not given ? => find the *best* raster value for the time range specified in the call !
 - this is how you do ‘optical zooming’ !
 - (and is not ‘garbage in/garbage out’)

Archiving & Archive Retrieval

- How to get 'all the stored points' in an interval ?
 - **Method 1:**
 - get the number of points in the interval
 - ask for that many points in the archive call.



**850865 points from
Oct. 12 to Nov. 12 :**

**Wow! That's quite a
few points ...**

Archiving & Archive Retrieval

- How to get 'all the stored points' in an interval ?
 - **Method 2:**
 - specify a *reasonable number* of points, but give a *sampling raster of 1*
 - move the *start* of the interval along and keep repeating the call ...

The screenshot shows the Java Instant Client interface with the following settings:

- Context: PETRA
- Subsystem: ALL
- Server: HISTORY
- Device: keyword
- Property: CurDC
- Data Size: 1000
- Data Type: DBLTIME
- Beam Current: PETRA
- Timeout: 1000
- Write Access:
- Input Data Type: TEXT
- starttime=12.10.2019 17:42:44, stoptime=12.11.2019 17:42:56, sample=1

The main display area shows the following data:

```

/PETRA/HISTORY/keyword CurDC @ 18:15:22.479
system stamp: 1309666944, user stamp: 0
(0,0) [100.1710433959961, 12.10.19 17:42:44.778 CEST]
(0,1) [100.10966491699219, 12.10.19 17:42:47.809 CEST]
(0,2) [100.03997802734375, 12.10.19 17:42:50.837 CEST]
(0,3) [100.41331481933594, 12.10.19 17:42:53.880 CEST]
(0,4) [100.82368469238281, 12.10.19 17:42:56.919 CEST]
(0,5) [100.77667236328125, 12.10.19 17:42:59.962 CEST]
(0,6) [100.71491241455078, 12.10.19 17:43:02.993 CEST]
(0,7) [100.65454864501953, 12.10.19 17:43:05.044 CEST]
(0,8) [100.60028076171875, 12.10.19 17:43:08.051 CEST]
(0,9) [100.5327377319336, 12.10.19 17:43:11.068 CEST]
(0,10) [100.46388244628906, 12.10.19 17:43:14.072 CEST]
(0,11) [100.40006256103516, 12.10.19 17:43:17.097 CEST]
(0,12) [100.34510040283203, 12.10.19 17:43:20.144 CEST]
(0,13) [100.277099609375, 12.10.19 17:43:23.185 CEST]
(0,14) [100.21788787841797, 12.10.19 17:43:26.208 CEST]
(0,15) [100.15776824951172, 12.10.19 17:43:29.230 CEST]

```

Settings: UDP, Timer | Suppress Query Properties, Property Query Precedence

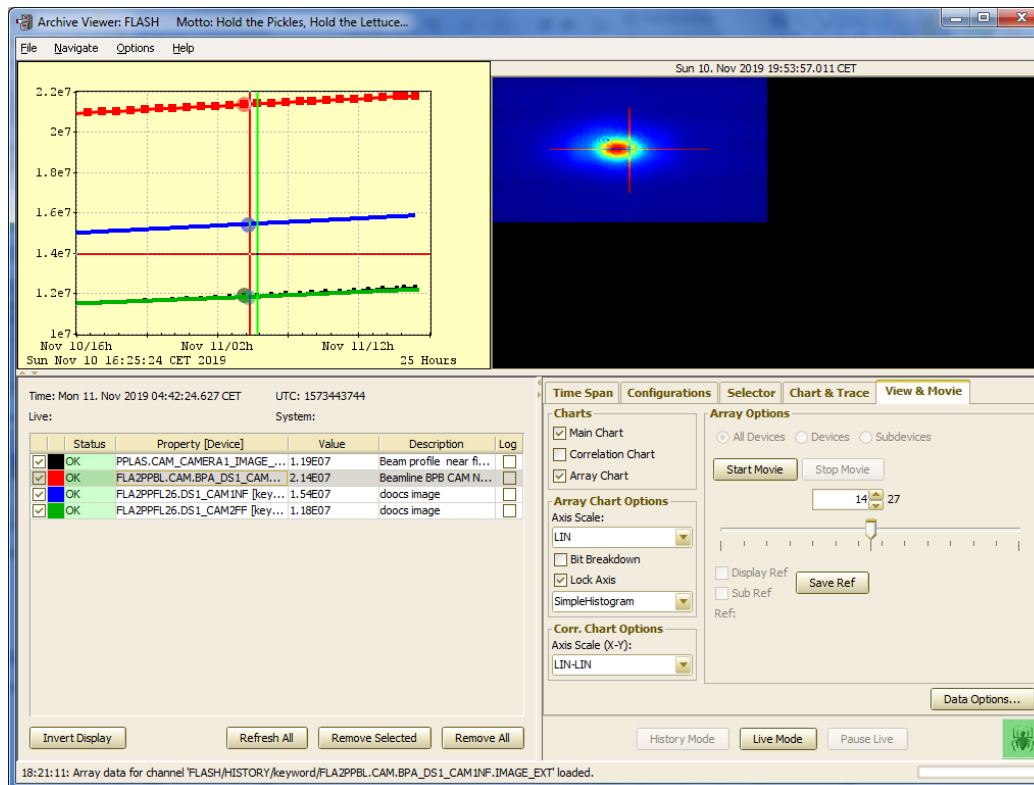
Last request: 18:15:22.415 (5 ms)

new start time = time of most recent valid archive timestamp returned in previous call.



Archiving & Archive Retrieval

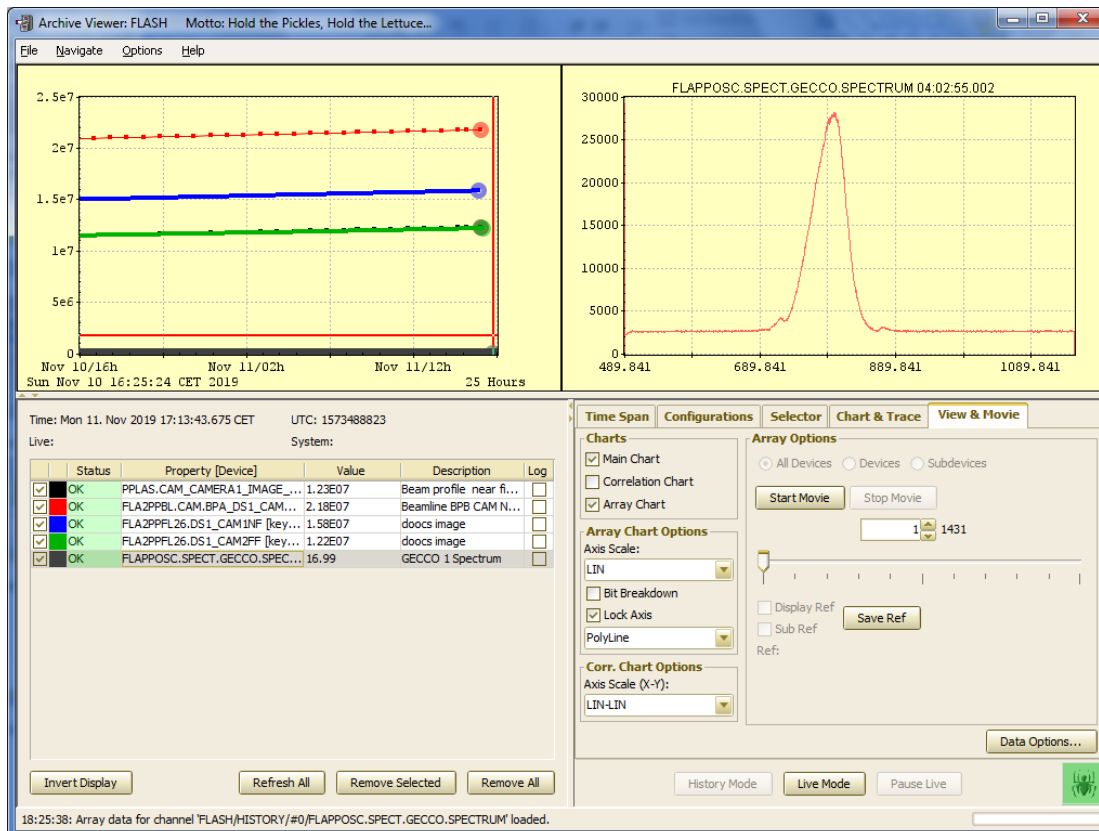
- Archiving Images and large waveforms ...



Size of a stored record can be large ...

Archiving & Archive Retrieval

- Pay attention to tolerances and filtering criteria



**Records stored only when warranted :
Out of tolerance
All filter criteria met (e.g. BEAM, RUNNING, etc.)**

There is a possibility of post facto reducing the stored data ...



Archiving & Archive Retrieval

- *massaging* stored data with **dbscan** ...

```
./dbscan
```

```
scans and massages the targeted archive data file
```

```
Usage : dbscan /f=<database file name> (/k=<archive keyword> or /r=<record>)  
        [/y=<year> /m=<month> /x=<scale correction> /o=<offset correction>  
        /d=<start::stop - deletes all records found in range>  
        /s=<TRUE: scan only (don't create 'fix' file)>  
        /v=<TRUE: dump scanned contents> /l=<dump array length>  
        /b=<TRUE: bailout on error> /a=<TRUE: add '-fix' extension>  
        /n=<new record length> /i=<TRUE: scan POI file>  
        /p=<prune data skip raster>]
```

```
e.g. dbscan /f=datapetr.csv /r=1791 /y=2018 /m=10 /p=10
```

```
scans record 1791 of the 'datapetr.csv' database from October 2018 and prunes the  
data by taking every 10th stored value
```

Next: post facto tolerance application ...



Archiving & Archive Retrieval

Archive Filters ...

The screenshot shows the 'Archive Database Manager' software. On the left, a table lists database entries with columns for Index, Active status, Device Server, Device Name, and Device Property. The entry with Index 6475 is selected. On the right, the configuration panel for Index 6475 is shown, including sections for Data Collection Configuration, Filtering of Data Storage, and Property Viewing Configuration. The 'Filtering of Data Storage' section has 'SLOW' selected. The 'Access Rate' is set to 60000 ms and the 'Archive Heartbeat' is set to 3600 sec. A text box at the bottom left explains the 'SLOW' filter setting.

Index	Active	Device Server	Device Name	Device Property
6456	ENABLED	DISML	PPLAS.CAM1	Image
6457	ENABLED	DISML	PPLAS.CAM2	Image
6458	ENABLED	DISML	BPB_DS1_CAM1NF	Image
6459	ENABLED	DISML	BPB_DS1_CAM2FF	Image
6460	ENABLED	DISML	BPA_DS1_CAM1NF	Image
6461	ENABLED	DISML	BPA_DS1_CAM2FF	Image
6462	ENABLED	FLA2PPDESKTOP.CAM	psAvesta	ROI_SPECTRUM.X.SIG
6463	ENABLED	FLA2PPDESKTOP.CAM	psAvesta	ROI_SPECTRUM.X.TD
6464	ENABLED	ComBobPolarix	#0	Power.Forw.Sample
6465	ENABLED	ComBobPolarix	#0	Power.Forw.Sample.NAM
6466	ENABLED	ComBobPolarix	#0	Power.Refl.Sample
6467	ENABLED	ComBobPolarix	#0	Power.Preamp.Sample
6469	ENABLED	FLA2PPDESKTOP.CAM	psAvesta	ROI_SPECTRUM.X.MAX
6470	ENABLED	FLA2PPDESKTOP.CAM	psAvesta	ROI_SPECTRUM.X.OFFS
6471	ENABLED	FLA2PPDESKTOP.CAM	psAvesta	ROI_SPECTRUM.X.MEAN
6472	ENABLED	ComBobPolarix	#0	Power.Preamp.Sample.N...
6473	ENABLED	ComBobPolarix	#0	Timing
6474	ENABLED	ComBobPolarix	#0	Timing.NAM
6475	ENABLED	DISML.26	DS1_CAM1NF	Image
6476	ENABLED	DISML.26	DS1_CAM2FF	Image
6477	ENABLED	DISML.26	DS2_CAM1NF	Image
6478	ENABLED	DISML.26	DS2_CAM2FF	Image
6479	ENABLED	FLASH2CPUFL24.PULSE...	CH00	PULSECOUNT
6480	ENABLED	FLASH2CPUFL24.PULSE...	CH00	PULSEENERGY.MEAN
6481	ENABLED	FLASH2CPUFL24.PULSE...	CH02	PULSECOUNT

Archive Filters ...

Index: 6475

Data Collection Configuration

Context: FLASH
Server: DISML.26
Device: DS1_CAM1NF
Property: Image
Format: IMAGE
Array Size: 200000
Input Format: NULL
Data Input:

Filtering of Data Storage

NEVER ONCE ALWAYS FAST
 SLOW FIXTIME HRT STATUS
 VOLATILE NOPOI BEAM KLYGLN
 KLYACC1 KLYACC23 KLYACC45 KLYACC6

Access Rate: 60000 ms
Archive Heartbeat: 3600 sec

Property Viewing Configuration

FLA2PPFL26.DS1_CAM1NF,IMAGE,200000,,0.0,0.0,0.0,0.0,LIN,1.0,0.0,docs image,,,PPLaser2

Maximum size [bytes]: 200000 Remaining elements: 0

Keyword	Data Format	Size	Units	Max	Min
CAM1NF	IMAGE	200000		0.0	0.0

ce	Rel. Tolerance	Plot Style	Offset	Scale
	0.0	LIN	0.0	1.0

Subsystem: PPLaser2 Associate:

Min Max Units

Bind To: Spectrum Axis:

Apply Add Remove

Reload DB Write DB Lock DB DB unlocked

If Access Rate > 1 minute then
'SLOW' => archive @ Archive Heartbeat

[Python News]

- **PyTINE API** offers *asynchronous* interface
 - `PyTine.attach()`

```
>>> import PyTine as pt
>>>
>>> def cb(id,cc,d):
...     print(d['timestring'])
...     print(d['data'])
...
>>>
>>> lid=pt.attach(address='/PETRA/Idc/Buffer-0',property='I',callback=cb)
>>> 30.05.15 11:50:13.413 CDT
100.5762939453125
30.05.15 11:50:14.297 CDT
100.57471466064453
30.05.15 11:50:15.401 CDT
```

- + `PyTine.detach()`

Callbacks and GUI operations ...



TINE and Python

From a 2015 TINE Users Meeting

■ PyQt callback paradigm:

```
import pyqtgraph as pg
import pyqtgraph.exporters
from pyqtgraph.Qt import QtCore, QtGui

from PyQt4.QtCore import QObject, pyqtSignal, pyqtSlot

import os
import threading
import ctypes as C
import numpy as np

import PyTine as tine
import time

class Update(QObject):
    update = pyqtSignal()
    def __init__(self):
        QObject.__init__(self)
    def do_update(self):
        self.update.emit()
```

```
@pyqtSlot()
def on_update():
    pl.setData(y=yd, x=xd);

upd = Update();
upd.update.connect(on_update);

xd = np.arange(0, 32768);
yd = np.arange(0, 32768);

def ptrain_cb(a, b, c):
    global yd
    yd = c['data'][0:32768];
    upd.update.emit();
```

Matplotlib + callbacks

```
import matplotlib.pyplot as plt
import matplotlib
import threading
import time
import queue
import functools
import copy
import PyTine as pt

#ript(Run In Plotting Thread) decorator
def ript(function):
    def ript_this(*args, **kwargs):
        global send_queue, return_queue, plot_thread
        if threading.currentThread() == plot_thread: #if called from the plotting
            return function(*args, **kwargs)
        else: #if called from a different thread -> send function to queue
            send_queue.put(functools.partial(function, *args, **kwargs))
            return_parameters = return_queue.get(True) # blocking (wait for ret
            return return_parameters
    return ript_this

#list functions in matplotlib you will use
functions_to_decorate = [[matplotlib.axes.Axes,'plot'],
                        [matplotlib.axes.Axes,'bar'],
                        [matplotlib.axes.Axes,'hist'],
                        [matplotlib.figure.Figure,'savefig'],
                        [matplotlib.backends.backend_tkagg.FigureCanvasTkAgg,'draw'],
                        [matplotlib.lines.Line2D,'set_ydata'],
                        ]

#add the decorator to the functions
for function in functions_to_decorate:
    setattr(function[0], function[1], ript(getattr(function[0], function[1])))

# function that checks the send_queue and executes any functions found
def update_figure(window, send_queue, return_queue):
    try:
        callback = send_queue.get(False) # get function from queue, false=doesn't block
        return_parameters = callback() # run function from queue
        return_queue.put(return_parameters)
    except:
        None
    window.after(10, update_figure, window, send_queue, return_queue)

# function to start plot thread
def plot():
    # we use these global variables because we need to access them from within the decorator
    global plot_thread, send_queue, return_queue
    return_queue = queue.Queue()
    send_queue = queue.Queue()
    plot_thread=threading.currentThread()
    # we use these global variables because we need to access them from the main thread
    global ax, fig
    fig, ax = plt.subplots()
    # we need the matplotlib window in order to access the main loop
    window=plt.get_current_fig_manager().window
    # we use window.after to check the queue periodically
    window.after(10, update_figure, window, send_queue, return_queue)
    # we start the main loop with plt.plot()
    plt.show()

q = queue.Queue(10)

def cb(id,cc,d):
    global q
    q.put(copy.copy(d))
```

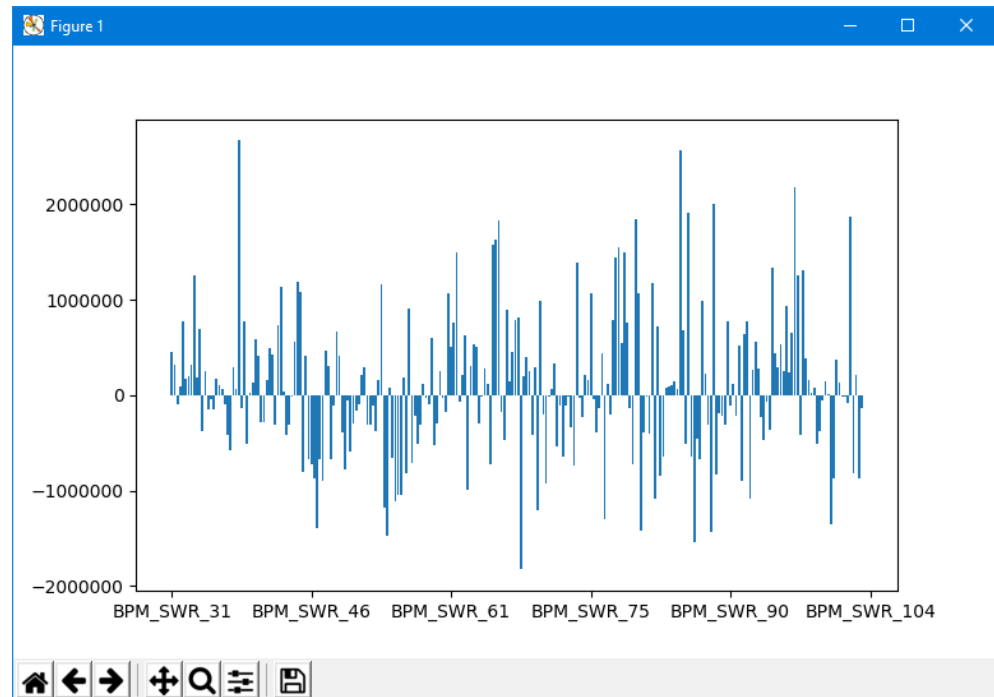
Some helper functions ...



Matplotlib + callbacks

```
def cb(id,cc,d):
    global q
    q.put(copy.copy(d))

def main():
    #start the plot and open the window
    thread = threading.Thread(target=plot)
    thread.setDaemon(True)
    thread.start()
    time.sleep(1) #we need the other thread to set 'fig' and 'ax' before we continue
    lid = pt.attach('/PETRA/BPM/#0','Orbit.X',callback=cb)
    names = pt.list(context='PETRA',server='BPM')
    global ax, fig, q, plotted, l1
    plotted = False
    while thread.isAlive():
        if not q.empty():
            d = q.get()
            if not plotted:
                #p1, = ax.plot(d['data'])
                b1 = ax.bar(range(len(d['data'])),d['data'])
                ax.set_xticklabels(names['devices'])
                plotted = True
            else:
                #p1.set_ydata(d['data'])
                for r, h in zip(b1, d['data']):
                    r.set_height(h)
                fig.canvas.draw()
            else:
                time.sleep(1)
    print('Done')
    #thread.join() #wait for user to close window
main()
```



[PyTine Synchronous API]

- Avoid the callback issues by using **synchronous** calls ?
 - **PyTine.get()** read a value without input
 - **PyTine.call()** read (and/or write) a value with/without input
 - unless the option **SYNC** is specified, the call will start an *asynchronous listener* from the buffered service layer
 - 'reflected memory'
 - make your call and update your GUI ...
- **BUT:**
 - built in latency
 - don't know when the data are fresh (when is a good time to call **PyTine.get()** ?
- Potential issue with a listener :
 - *when does it stop listening?*
 - **deadband** of 5 minutes (without a **get()**) will stop the listener.
 - (larger payloads have a smaller deadband)



[PyTine Synchronous API]

- What if a **GUI client** changes an address (e.g. a device in a combo box) and *knows* that it **won't call the old address again**?
 - waiting for a large payload to disappear via the **deadband** can lead to unnecessary network and cpu load.
 - e.g. video frames or large waveforms
- New call : **PyTine.stop_get()**
 - with same parameters as the original get() will stop the listener.



[PyTine news]

- **Orbit correction :**

- that old orbit correction/optics server (from 1993) can still be used !
 - uses (non-tagged) **structures** (we didn't have tagged structures back in 1993)

- **New PyTine calls :**

- **register_type()**
 - puts a user-defined tagged structure in the TINE structure registry
- **structure_to_bytes()**
 - Gets an array of bytes from a structure
- **bytes_to_structure()**
 - Makes a structure from an array of bytes

Orbcore Python example:

```
import PyTine as pt
hdr = {
    'Machine':{'char*8':'hera'},
    'Year':{'char*4':'2007'},
    'Text1':{'char*60':''},
    'Text2':{'char*60':''},
    'DateTime':{'char*32':''},
    'filename':{'char*80':''},
    'Optic':{'char*16':''},
    'Opticflag':{'char*16':''},
    'particleType':{'short':0},
    'dummy':{'BYTE':0},
    'CorrClass':{'BYTE':0},
    'errorStr':{'char*32':''},
    'Energy':920.0,
    'Arr_mon':{'short':0},
    'Arr_cor':{'short':0},
    'Arr_il':{'short':0},
    'Arr_jl':{'short':0},
    'Hdr_Size':{'short':0},
    'Spez_Hdr_Size':{'short':0},
    'Arr_Size':{'short':0},
    'MonNum':{'short':[0,0]},
    'CorNum':{'short':[0,0]},
    'LatNum':{'short':0},
    'EleNum':{'short':0},
    'OpticNum':{'short':0},
    'SpezNum':{'short':0},
    'Counter':{'short':0},
    'Qtheory':[0.0,0.0],
    'mom_comp':0.0,
    'Circumference':{'double':0.0},
    'NGDebug':{'short':0},
    'HPDebug':{'short':0},
    'Index':{'short':0},
    'Version':{'short':0}
}

bumpHdr = {
    'CoilName':{'char*16':''},
    'Strength':0.0,
    'Current':0.0,
    'dK':0.0,
    'dI':0.0,
    'dKmax':0.0,
    'dKmin':0.0,
    'Coil_MopsIndex':{'short':0},
    'Coil_CorMIndex':{'short':0},
    'kIMinMax':{'short':0},
    'ErrorFlag':{'short':0},
}

pt.register_type('BmpHdr', bumpHdr)

b = pt.structure_to_bytes('OptHdr', hdr)

output = pt.call(address='/Common/OPTICL3/pe14_p3x_v23', property='HEADER', size=len(b),
input=b, inputsize=len(b), format='STRUCT.BYTES', inputformat='STRUCT.BYTES', mode='READ|WRITE')

pt.register_type('OptHdr', hdr)
```

Orbcore Python example

```
psName = ["PKV_NOR_43", "PKVSX_SOR_89", "PKVSU_SL_68", "PKH_NOR_40", "PKHW_WL_31", "PCV_NWL_13",
          "PCVM_NWL_31", "PKDK_SWR_27", "PKPDA_NOR_45", "PCH_NWL_9", "PKVW_WL_25", "QS_W1"] # an array to test
strength = [0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005, 0.0005]

for n, s in zip(psName, strength):
    bumpHdr['CoilName'] = n
    bumpHdr['Strength'] = s
    b = b + pt.structure_to_bytes('BmpHdr', bumpHdr) #add all the bytes together ...

for i in range(len(psName), 400):
    b = b + pt.structure_to_bytes('BmpHdr', bumpHdr) #add all the bytes together ...

output = pt.call(address='/Common/K2I2KL3', property='GET-CURRENT', size=len(b), input=b, inputsize=len(b), format=
'STRUCT.BYTES', inputformat='STRUCT.BYTES', mode='READ|WRITE')

b1=output['data'][:376] # split the returned byte blob up into known quantities ...
b2=output['data'][376:424]

h1=pt.bytes_to_structure('OptHdr', b1)
h2=pt.bytes_to_structure('BmpHdr', b2)
```

Bare bones orbit correction, but Gajendra is writing some nice wrappers ...