Infrastructure-independent Device Control

Andrei Sukhanov, James Jamilkowski.

Date 08/24/21

Abstract

In modern accelerator control systems there is often a need for controlling devices which have been developed with interface to a different control infrastructure. For example, the future EIC collider will have mixture of **EPICS**, **ADO**[1] and **liteServer**[2] controlled devices. In this paper we describe the control tools and middle-level interface which allows to control the devices independently of their dedicated architecture.

Infrastructure-independent tools:

- **pype**t: spreadsheet-based parameter editing tool which covers functionality of the EDM, MEDM and PET.
- **ImageViewer**[3]**:** interactive image analysis tool, supporting AreaDetector, USB and ADO - controlled cameras.
- **pvplot** is simple parameter plotting tool.
- **apstrim**: logger and extractor of time-series data.

The tools are operational at RHIC collider and they provide control for devices from various infrastructures (ADO, EPICS and liteServer).

Simple middle-level interface

Interface to an infrastructure-depended level is provided by an Access class of corresponding module. All functions accept multiple Control System parameters (EPICS PV) as arguments.

The Access class provides set of 5 functions, which covers all required functionality:

- **info(*args)**: Returns information on device or PV as a 3-level dictionary.
- **set(*args)**: Set multiple PVs to values.
  get(*args): Get values of multiple PVs.
- **subscribe(callback, *args)**: Sets up a user-supplied callback function to be called on change of the first of requested PVs.
- **unsubscribe()**: Cancel all subscriptions.

The returned objects are two-level maps, the first level is mapped with PV name, the second level - with PV properties (‘value’, ‘timestamp’, etc.).

[1] D.S. Barton et al. RHIC control system, NIM A, Volume 499, Issues 2–3, 1 March 2003, Pages 356-371
[2] https://pypi.org/project/liteserver/
[3] https://accelconf.web.cern.ch/icalepcs2019/doi/JACoW-ICALEPCS2019-TUCPR06.html

Pypet application

- control of EPICS PVs and ADO parameters,
- automatic generation of control pages for devices [2],
- configuration format: **Python modules**,
- macro substitution from command line: single configuration file can be used for many similar devices,
- merged cells, adjustable size of rows and columns, fonts and colors,
- horizontal and vertical **slider widgets**,
- **embedded** displays from other programs,
- plotting of selected cells using pvplot,
- content-driven cell **coloring**,
- pypet pages are managed by **git,**
- **snapshots**: full page can be saved and the selected cells could be restored from the saved snapshots,
- **history** of parameter settings [3].
- **slicing** of vector parameters.

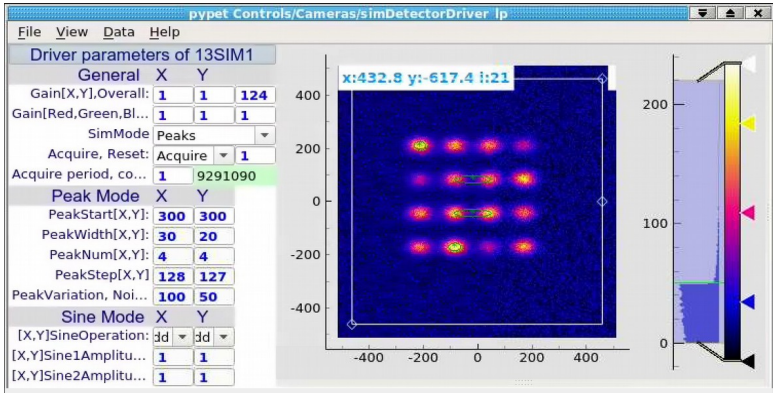


Fig.1 pypet page for EPICS simDetectorDriver with embedded display from imageViewer.

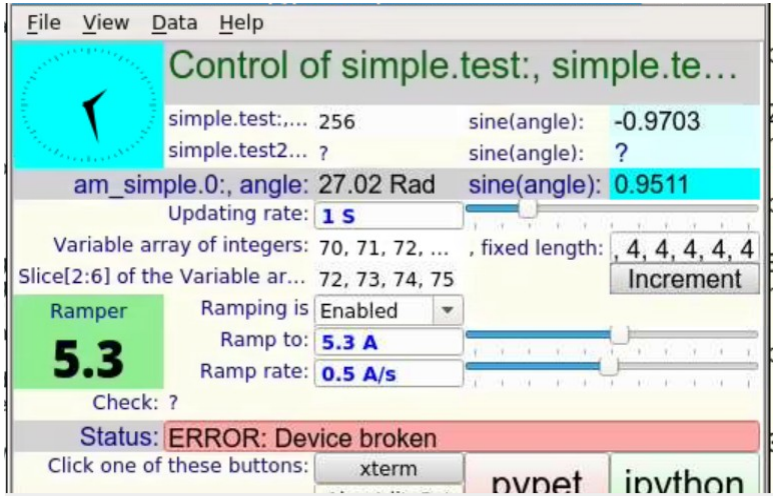


Fig.2 pypet page for a test ADO.

Brookhaven National Laboratory

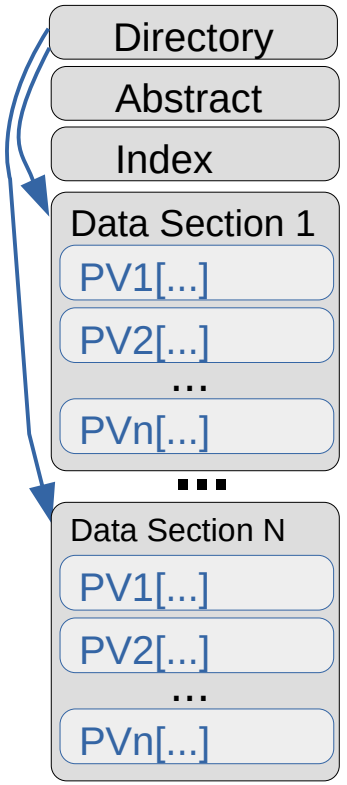
[2] not supported for EPICS, due to lack of introspection in EPICS devices.
[3] not supported for EPICS.

apstrim: Logger and extractor of time-series data

- Data are objects, indexed in time order.
- Wide range of data objects: strings, lists, maps, numpy arrays, custom.
- Data objects could be inhomogeneous and have arbitrary frequency.
- Self-describing data format.
- Data objects are binary-serialized using MessagePack.
- Fast online **compression**.
- Fast **random-access** retrieval of data objects for selected time interval.
- Simultaneous writing and reading.
- Typical speed of compressed serialization to a logbook file is **70 MB/s**.
- De-serialization speed is up to **1200 MB/s** when the logbook is cached in memory.

Implementation:
apstrim code: Python
msgpack: all languages.

File format



Positions of data sections.

Version, compression type, etc.

List of all PV names

List of PV read during time interval #1

List of values and timestamps of PV1

List of values and timestamps of PV2

List of values and timestamps of PVn

Time-based bucketing.
Data are accumulated asynchronously during section intervals into time-stamped sections.
When interval elapses, the section is compressed and dumped to the logbook.

Python installation:
https://pypi.org/project/apstrim

Brookhaven National Laboratory

Summary and Acknowledgments

Simple middle level Python interface to underlying infrastructure-dependent modules (**caproto** for EPICS and **cad_io** for ADO).

Complete toolset to control devices from different infrastructures.

The tools are operational at RHIC collider and they provide control for devices from ADO, EPICS and liteServer infrastuctures.

ACKNOWLEDGMENTS

Work supported by Brookhaven Science Associates, LLC under Contract No. DE-SC0012704 with the U.S. Department of Energy

Brookhaven National Laboratory