Development of the Software Tools Using Python for EPICS-Based Control System

Tatsuro Nakamura, Kazuro Furukawa, Jun-Ichi Odagiri, Noboru Yamamoto

KEK, Tsukuba, Ibaraki, Japan

Abstruct

In the commissioning phase of accelerators, many application programs are built and modified frequently by nonexpert programmers. Scripting language such as Python is suitable for such quick development. Since EPICS Channel Access interface library in Python was developed in KEKB accelerator control system, many programs has been written in Python. We have been developing, providing some tools and libraries for Python programming. Some of the recent developments in KEK are reported, and possible applications are also discussed.

Introduction ----- About KEKB

KEKB is an asymmetric electron-positron collider at 8 × 3.5 GeV/c for B-meson physics.

KEKB started in operation in Dec.1998.

KEKB accelerators control system:

EPICS-based

More than 100 VME/VxWorks computers as IOC

Several workstations of 4 kinds of platform

PA-RISC/HP-UX, Alpha/OSF1,

PC-AT/Linux, Macintosh/OSX

Introduction ----- About Python

Python is an interpretive programming language.

Suitable for rapid application development

Simple and easy to learn

Python has powerful features:

Efficient high level data structures

Object-oriented programming

Rich libraries, which cover wide range of areas

Python code is highly productive and maintainable.

Python-CA

Python-CA is a Python interface module to EPICS CA (Channel Access).

Python-CA provides basic functions of CA:

get, put, monitor

All of Python programs which perform CA are built based on this module.

Example of Python-CA (get operation)

```
import ca
chan = ca.channel("channel_name")
chan.wait_conn()
chan.get()
ca.pend_event(1.0)
value = chan.val
```

Simple-CA

Simple-CA library is

A wrapper library of Python-CA

Easier to use

Single get or put operation in single function call

(It implicitly waits until the request is completed.)

Example of synchronous get by Simple-CA

```
import cas
value = cas.caget("channel_name")
```

List operation of Simple-CA

Multiple channels at single function call

More efficient than multiple function call for each channels

Example of multiple gets by Simple-CA

```
import cas
v1,v2,v3 = cas.caget(["name1","name2","name3"])
```

CA-Widgets

CA-Widgets is a ready-made widget library

Convenient parts set to build GUI of control application programs.

CA-Widgets is:

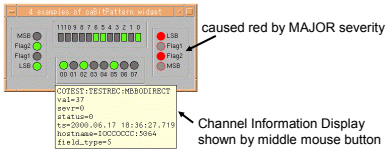
based on Tkinter (Python interface to Tcl/Tk)

easily extensible using object-oriented feature of Python/Tkinter.

Examples:

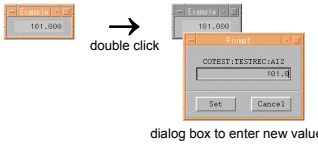
caBitPattern widgets

Bit pattern representation of channel value



caWritableLabel widget

Text representation of channel value



casave

casave is

a library module to save channel values to files.

Available formats are

"db"	Compatible to EPICS database file
"dbpf"	List of dbpf commands
"simple"	Each line contains channel name and value.
"valstat"	Each line contains channel name, value, severity, status, and timestamp.

Example of configuration file of casave

```
from casave import defSaveFile
defSaveFile(
    iocname = "ioc_name",
    filename = "save_file_name.db",
    filedir = "dir_path/",
    backup = "flat",
    format = "db",
    chanlist = [("channel_name1","rec_name1","INP","ai"),
                ("channel_name2","rec_name2","DOL","ao"),
                ("cahnnel_name3","rec_name3","DOL","bo")]
)
```

Example of saved file (**db** format)

```
record(ai,"rec_name1") {field(INP, " 1.25000000000000000E+00")}
record(ao,"rec_name2") {field(DOL, " 8.00000000000000000E+00")}
record(bo,"rec_name3") {field(DOL, " 0")}
```

Sample code of save operation using casave

```
from casave import SaveFileDict
import configuration1, configuration2, configuration3
for v in SaveFileDict.values():
    v.get()
    v.dbsave()
```

AutoSaver

AutoSaver ----- An application of casave

AutoSaver automatically save files when monitoring channels are disconnected.