ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOSH4001

A LIBRARY OF FUNDAMENTAL BUILDING BLOCKS FOR EXPERIMENTAL CONTROL SOFTWARE

M. Scarcia†, R. Borghes, M. Lonza, M. Manfredda, E. Pedersoli, R. Mincigrucci
Elettra Sincrotrone Trieste S.C.p.A., Trieste, Italy

Abstract

In many experimental facilities there is a rising interest by users and beamline scientists to take part in the experiment control software development process. This necessity arises from the flexibility and adaptability of many beamlines, that can run very different experiments, requiring changes in the software even during experiments.

On the other side, we still need a professional and controlled approach in order to be able to maintain the software efficiently.

Our proposed solution is to exploit the object oriented nature of programming languages to create a library that provides a uniform interface both to the different controlled devices and to experimental procedures. Every component and procedure can be represented as an object, a building block for experiment control scripts.

INTRODUCTION

The scientific data acquisition workflow at the FERMI [1] facility of Elettra Sincrotrone is organized around two software components. The FermiDAQ [2] takes care of the actual data sources acquisition (including corresponding metadata) and of its synchronization with the information gathered from the FEL bunches. The Executer runs different experimental procedures in the form of Python scripts.

Initially some standard experiment templates were defined for each beamline by its staff and were implemented through Python scripts. These scripts were meant not be modifiable by beamline scientists and users, but only managed by the software developers in order to maintain a secure and robust approach to the underlying control and data acquisition systems.

This approach proved in time to be impractical on the FERMI FEL, where experimental setups and procedures were observed to change sometimes even within a single beamtime. Furthermore, even after the commissioning of the machine was officially concluded a lot of internal research has been carried out by the machine physics and beam transport groups, which is logistically usually based on beamline software infrastructure. This type of work often requests a rather different workflow than user oriented research.

The secondary motivation for this work derives from a different problem we encountered in day-to-day operations. The Fermi control infrastructure, which is based on the TANGO framework [3], comprises different software devices that have different origins and usually different development histories. This means in practice that different devices of the same type (e.g. motor controllers) have different command names for the same action (e.g. "Move" instead of "MoveTo"), requiring specific code to be written for each different device even if the actions to be performed are the same. A uniform upper level interface for every type of "abstract" device in the experiment control and data acquisition would therefore be a welcome tool.

OVERVIEW AND IMPLEMENTATION

Two main lines of development are currently being merged in this project. The first regards the creation of elementary "building blocks" with which experimental procedures can be built with greater ease and simplicity. The second concerns producing a uniform interface and the possibility of grouping different devices performing similar tasks. The goal is to have a model that follows experimental flow rather than the control system organization. Figure 1 shows a schematic view of the proposed library and its interaction with existing components.

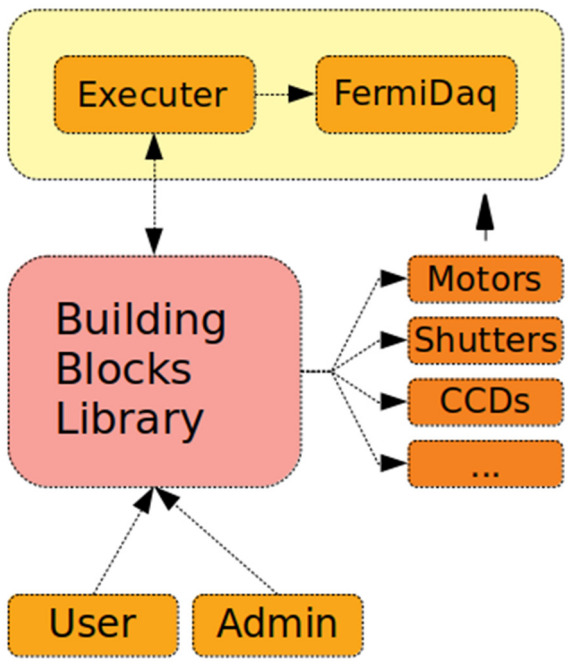

Figure 1: Library interactions.

Reusable Experimental Sequences

After the workflows on FERMI beamlines became more stabilized we realized that the abstraction level where every script is the equivalent of an experiment was too high for many common cases of beamline operation. The natural solution in order to decrease the level of abstraction was to find the common elements in the workflow and codify them as Python objects.

† martin.scarcia@elettra.trieste.it

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOSH4001

A common element in experimental scripts on FERMI beamlines is the synchronization of different devices with the bunch rate of the machine. The bulk acquisition is handled by the FermiDAQ at the machine frequency (10Hz or 50Hz), but many devices cannot operate at this high frequency. Furthermore, pump-and-probe experiments frequently require that different devices be synchronized with a single FEL shot, including mechanical shutters, optical delay lines, etc.

Originally each experimental sequence was handled separately, but this proved to be inefficient, due to frequent variations in the fine details and the addition of new devices. The core functionality was however the same, so this was a natural choice for the first "building block" to be implemented.

The concept of a "FEL sequence" was therefore coded as a Python class, with appropriate methods allowing the prompt creation of standard FEL shot sequences and trigger sequences for experiment control devices or acquired devices. All the currently used sequences can be rapidly instantiated as objects and there are ample possibilities of evolution because most of the objects properties can be customized.

The software was deployed and tested during an in-house beamtime performed by the beam transport group early this September. The challenge of controlling and synchronizing the acquisition of different devices across two different beamlines was met with success demonstrating the improved versatility of this new approach as the software routines needed to control this "unusual" beamtime were essentially the same that could control standard experiments.

Device Abstraction and Grouping

A common need that has arisen during beamtimes on FERMI is the possibility to group devices dynamically into sets according to different operational needs. The same motors for example in one context may be grouped to represent the movement of a sample holder, while in another context they could represent the movement of a CCD camera. The two sets usually though do not overlap completely, so a dynamical management of these sets is important to achieve efficiency when controlling the experimental environment.

A library of classes representing the different type of devices found on the beamlines was developed including motors, shutters, power supplies, valves, etc. These abstract "devices" allow to have a uniform interface to different software implementations. Derived classes allow to group objects according to the required experimental setup. In this way the perspective is shifted from the control system point of view (the devices are grouped by driver/implementation) to the beamline scientist point of view (the devices are grouped by their experimental function). In the first case the organization is fixed, in the second we have the desired flexibility.

This library has been used in the past year proficiently on most of the FERMI beamlines.

FURTHER DEVELOPMENTS

The next step in the development of this software will be to integrate the software written until now in a single library with a more uniform interface and the appropriate documentation. On top of this we plan also to provide a higher level of functions based on this library, that will replace the currently used "experimental scripts". Software maintenance will be simplyfied and beamline scientists or users will be allowed to write their own scripts if they wish to do so, while still providing a simple user interface for standard operation. In this way we will provide advanced users with a stable and controlled interface to the more sensitive parts of the software infrastructure such as the Tango servers controlling the end-station devices or the machine control systems.

We plan also to integrate this library with a quick script-prototyping tool, which is currently used mainly in the data post-processing phase.

CONCLUSION

With this library we can now provide the scientists with a flexible tool for implementing highly customizable control software to run experiments. Furthermore, the object-oriented structure makes the development of experiment control scripts easier and quicker also for software developers decreasing software maintenance costs.

This flexibility does not come at the expense of stability as we are able to protect the most sensitive structures such as the control system infrastructure beneath a strong and trusted software layer.

REFERENCES

[1] L. Giannessi and C. Masciovecchio, "FERMI: Present and Future Challenges", *Applied Sciences,* vol. 7 – 6, 2017.

[2] R. Borghes *et al.*, "A Common Software Framework for FEL Data Acquisition and Experiment Management at FERMI", in *Proc. 14th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'13)*, San Francisco, CA, USA, Oct. 2013, paper FRCOAAB06, pp. 1481-1484.

[3] TANGO, http://www.tango-controls.org