New Experiments Control for ESRF Beamlines

Vincent Michel, Antonia Beteva, Tiago Coutinho, Marie-Christine Dominguez, Matias Guijarro, Cyril Guilloud, Alejandro Homs, Jens Meyer, Emmanuel Papillon, Manuel Perez, Sébastien Petitdemange

Grenoble, France

Bliss is the new control system for ESRF beamlines, with full deployment aimed for the end of the EBS upgrade program in 2021. It provides a global approach to run synchrotron experiments with centralized configuration services, support for many kinds of equipments and an advanced scanning engine.

Central configuration and services

A server called Beacon centralizes all the configuration and synchronization services.

- Static configuration for the different piece of equipments. This configuration is stored in YAML files.
- User scripts and session setup. They are python files that can be imported in the user sessions.
- Include a tango database for interoperability with existing tango servers.
- Provides a cache for different settings. The cache is stored in a central redis database.
- Provides transient data store used to keep the data acquired during a scan. This is also stored in the redis database.
- Act as a message broker to implement shared state and distributed lock.

Hardware control

The Bliss project supports many kinds of equipments.

- The controller objects are loaded from the configuration when initializing a user session. Controllers are python objects exposing the different features of the corresponding equipments.
- Direct hardware access is implemented when possible, in order to have a precise control over the corresponding equipment and schedule the hardware requests in a rigorous way.
- Concurrency is achieved using gevent, a coroutine-based python library. Cooperative multitasking within a single thread greatly decreases the chance of a race condition while the coroutine interface allows for simpler sequential code.

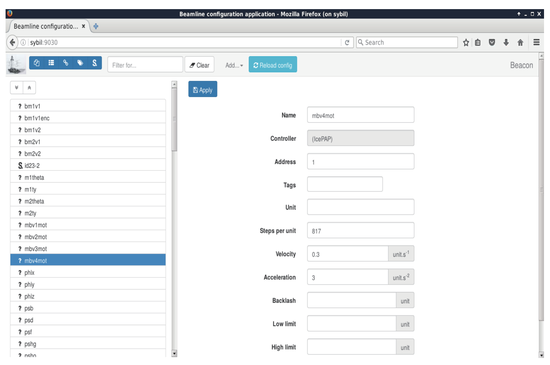
Scanning and data management

Bliss provides an advanced framework to write and perform many kinds of scans.

- Higher-level functions are provided for different step scans (so called ascan, dscan, timescan, etc.). Arbitrary counters can be provided to those standard scans in order acquire data.
- A lower-level interface allows for developers to write more complex scans (typically, continuous scans). This interface provides so-called acquisition devices to be organized as a chain.
- The tree representation of the chain allows for possible decoupling between the different subsets of equipments involved (i.e. using different trigger signals running at different rates).
- The data is stored temporarily in the redis database and published through channels. Any application can subscribe to those channels to get the data on the fly. The default exporter is an HDF5 file writer.

User interfaces

The Bliss ecosystem provides several user interfaces for the different part of the project.

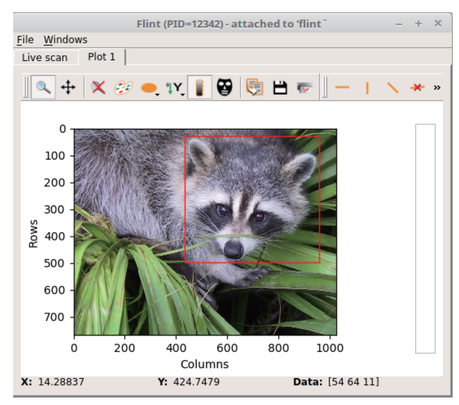



```
matias@kashyyyk:~ % bliss -s test_session
test_session: Executing setup...
Initializing 'heater`
...
Initializing 's1hg`
Done.

>>> ascan(m1, 0, 10, 30, 0.1, diode, save=False)
Total 30 points, 3.0 seconds

Scan 4 Mon Sep 11 11:58:03 2017 <no file> test_session user = guijarro
ascan m1 0 10 30 0.1

  #  timestamp  m1  diode
  0  1.50512e+09   0  499.112
  1  1.50512e+09  0.345  500.799
...
 28  1.50512e+09  9.655  505.622
 29  1.50512e+09  10  499.883
```



The beacon configuration application is a web application that allows user to manage and edit the static configuration and the user scripts from their web browser.

The Bliss shell is the main interface to the framework. It is an enhanced python interpreter based on ptpython that can be customized to include beamline specific information.

Web application are also being developed to provide monitoring capabilities such as this synoptic view. This prototype also features some plotting tools and an embedded shell.

Flint is a plotting application based on silx. It is attached to a bliss session and plot the data from the different channels as it is being acquired. It also provides offline plotting capabilities, allowing for interactive user routine (i.e. the selection of region of interest).

Tools and technologies

The Bliss project relies heavily on python and its wide ecosystem. Compatibility between the project and the external libraries is maintained through a large test suite and continuous integration. Most of the dependencies are available as conda packages on the official conda channels, and the missing ones are built and distributed on a private channel.

Current status

- The development version is already deployed on several beamlines.
- The first release is planned for the beginning of 2019.
- The full migration is planned for the end of the EBS program in 2021.
- Official repository - https://gitlab.esrf.fr/bliss/bliss

ESRF-EBS