Web-Based control system for the QUATI beamline at Sirius

Igor Ferreira Torquato[1], Santiago Figueroa[1], Alexey Espíndola[1], Eduardo Coelho[1], Amélie Rochet[1]
[1] Brazilian Synchrotron Light Laboratory (LNLS), Brazilian Center for Research in Energy and Materials (CNPEM)

Introduction

On the Quati beamline[1], which is the X-ray Absorption Spectroscopy (XAS) beamline of Sirius[2], the experiment control system is designed to offer a simple yet robust web-based interface to scripts and scan engines. The system architecture is composed of a centralized control server designed to serve multiple clients. The chosen architecture enables the use of asynchronous communication with the client, client-server decoupling, and non-simultaneous availability, offering greater flexibility and fault tolerance.

Main technologies used

The main technologies used in the system include RabbitMQ[3] for decoupling the GUI application from the executors (consumers), MongoDB for storing metadata related to scripts and runs, Flask[4] as the web framework and Bootstrap + JS as the font-end framework.

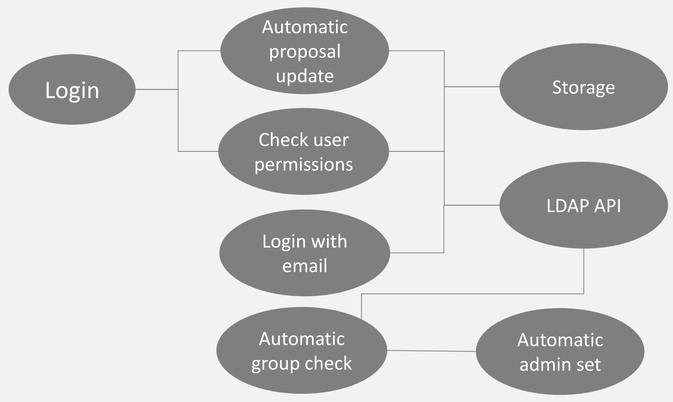
RabbitMQ MongoDB Flask Bootstrap

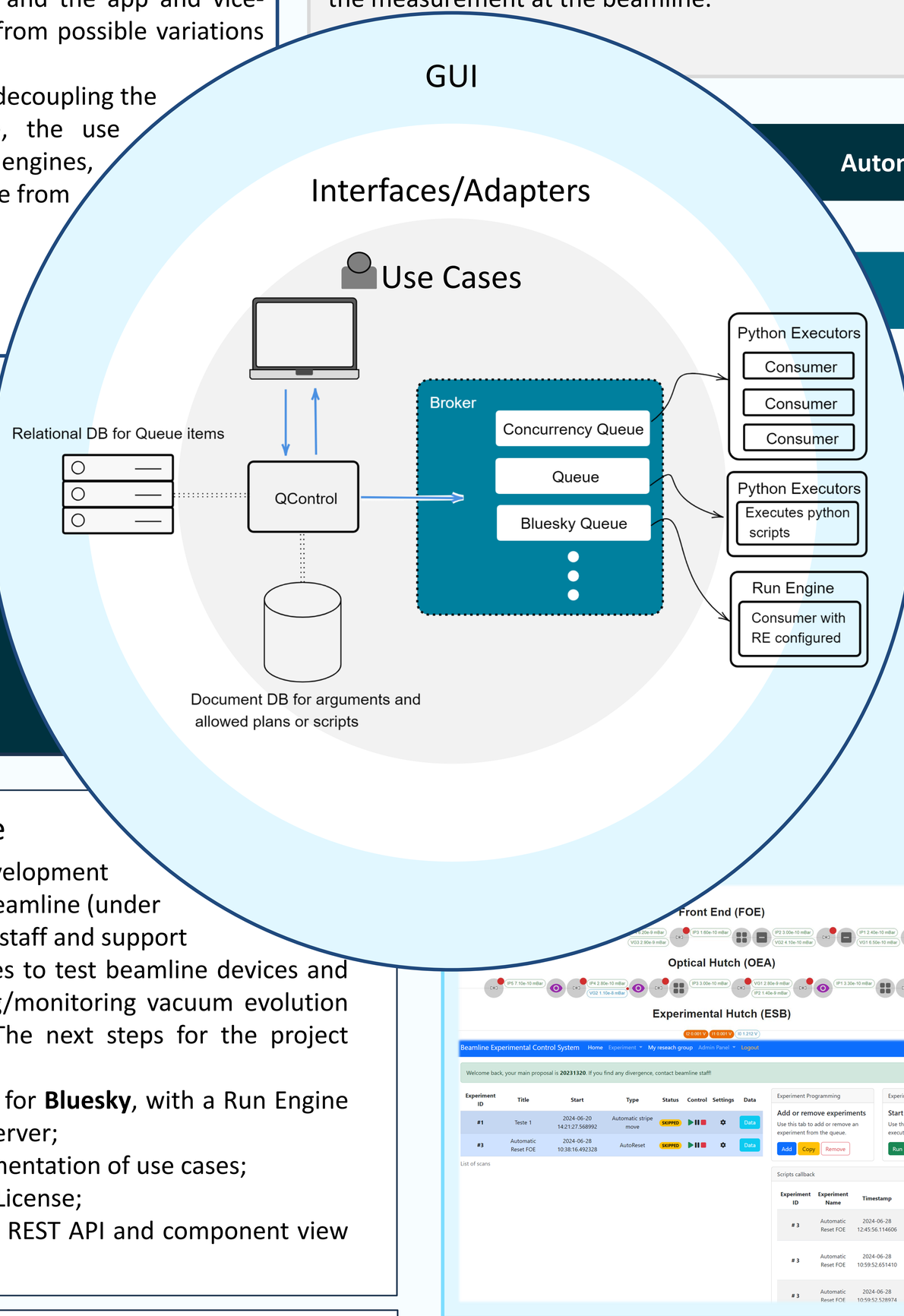
System architecture

The system architecture is designed with an application that is decoupled from the control system and scan engines. It maintains its own databases to store useful information for the beamline user, independent of the control system's requirements. The outermost layers of the application, along with interfaces, handle translating actions between the Graphical User Interface (GUI) and the app and vice-versa, isolating the core use cases from possible variations in the system requirements.
By reversing the dependencies and decoupling the applications using a Message Queue, the use cases are isolated not only from the engines, but also gain a level of independence from the infrastructure, enabling a fully testable system, from unit to integration tests.

Use Cases

Use cases were employed to design the entire system. They consider that common metadata, such as proposal number and facility information are "transparent" to the user. Therefore, the user/scientist focus will be only on the measurement at the beamline.





Automations for speedup user's productivity

Containerized application with Docker Compose

Decouple system with Message Queue

Consumers

Multiple Queues can be created to meet user requirements, with each queue linked to one or more consumers.
An abstract class for consumers is implemented, allowing for the integration with specific control systems.

Web

- Integrated with internal login;
- Checks for user proposals;
- Has user roles and admin features;
- Uses SSL for secure connection;
- Enables remote access;
- GUI can be Web or Desktop;
- Uses WebSockets for real-time updates.

Conclusions and perspective

The QControl is currently under development and is implemented on the Quati beamline (under commissioning). It has enabled the staff and support teams to synchronize their activities to test beamline devices and has also been used for controlling/monitoring vacuum evolution during beamline commissioning. The next steps for the project include:

- Develop a consumer for **Bluesky**, with a Run Engine or consulting HTTP server;
- Complete the implementation of use cases;
- Documentation and License;
- Possibly evaluating a REST API and component view of the system.

References

1.Figueroa, Santiago JA, et al. "QUATI beamline: QUick x-ray Absorption spectroscopy for TIme and space-resolved experiments at the Brazilian Synchrotron Light Laboratory." Radiation Physics and Chemistry 212 (2023): 111198
2. Liu, Lin, et al. "The sirius project." Journal of synchrotron radiation 21.5 (2014): 904-911.
3. RabbitMQ Documentation | RabbitMQ. (n.d.). https://www.rabbitmq.com/docs
4. Flask Documentation. https://flask.palletsprojects.com/en/3.0.x/

NOBUGS 2024
23- 27 September 2024
ESRF, Grenoble