Developers' meeting – minutes
MXCuBE project meeting, virtual Elettra,
October 2021

Participants (only those commenting, in order of appearance):

Peter Keller, Roberto Borghes, Marcus Oscarsson, Lais do Carmo, Antonia Beteva, Jordi Andreu, Ivars Karpics, Olof Svensson, Rasmus Fogh

The main focus of the meeting was on discussing needs and priorities with a 5-10 year time frame, including new features, scientific developments, improving workflow incorporation and sharing , the queueing system, and centring.

Peter Keller raised the problem of handling the directory structure of acquired data. Roberto Borghes seconded the point – The system at Elettrra moved data from temporary to long term storage, and there had been problems with files being overwritten, since there was no way of knowing, e.g. after restart, if a given file/directory name had already been used but moved away. MO and LdC said that run numbers should take care of this problem, but RB pointed out that this would require a database check. It was agreed that the way directories were handled should be clarified and made more powerful.

The discussion moved to the general problem of preserving state (including run number) when the link to the server drops out or you close down. IK had done some unfinished work on this using REDIS some years ago; storing e.g. centring points was easy enough, but storing the queue state was difficult. His work included a hardware object common for both Qt and Web versions, and could be resurrected. There was also related work on handling the beamline as a state machine. The problem of confidentiality would, however, have to be dealt with. MO suggested it would be desirable to develop a persistence layer and move it into the MXCuBE core. This would raise the problem of duplicating information in ISPyB, but storage in MXCuBE would be more explicit and straightforward. IK noted that acquisiton times for a 360º sweep are now as short as 10-20s, so that persistence is not a major problem for MX. For SSX, however, you may have a 20min acquisition run that continues even as MXCuBE crashes, so that persistence becomes quite important.

There was further discussion around adopting a messaging or message broker system for interprocess communication. This would be expected to clarify the handling of signals, but the disparity between Qt and web user interfaces might make the job difficult. PK noted that GDA worked with ActiveMQ. This is Java-based, but a good way of decoupling, it is however quite

heavyweight to deploy as soon as you go beyond the basics. OS has also bee looking at GDA, and is considering porting ZOCALO to the ESRF.

A related problem is that there are currently three different ways of triggering processing, via a script (characterisation), a web services call, or the autoprocessing hook. This might be worth simplifying. A good approach might be to have this as a service, so you could easily swop in a different protocol if necessary. There was a need for fast, while-you-wait processing in some cases, but it was pro[posed that this should not be too hard to handle if the service had access to the MXCuBE persistence layer.

RF asked about harmonising the use of signals between UI and hardware object layer; this is a different situation from the interprocess communication discussed above, and is subject to a separate uniformation effort.

For SSX there is a need for continuous feed-back as you collect, like for the characterisation plug-in. We should begin to think about the needs of SSX across the board (JA). Since SSX is actively worked on across multiple synchrotrons, one would have to either allow each site to add their own parameters for both queue entries and ISPyB storage, storing up for a very difficult merging task down the line, or make some kind of generic SSX objects. IK had been adding parameters one by one, and at one point stopped committing the results to Github. A generic keyword-value system with an upper limit on the number of parameters might be an idea? It was proposed to set down a working group to try to harmonise the parameters as they were created.

The ore general implementation of workflows should eb a goal, and another workgroup was proposed for this. PK pointed out 1) that there was an abstract beamline interface that could serve as a starting point for the data that would need to be transferred. 2) that any standard chosen should take care to avoid language dependencies. PK had found that XML-RPC could nt fully handle Python-Java data transfer. OS noted that ideally there should be a workflow solution as part of MXCuBE. It was noted as a complication that workflows often depend on third party software, e.g. X-ray centring depends on DOZOR-M. There are alternative programs (e.g. the older DOZOR), bur DOZOR-M seems to clearly the most powerful. Could these third party programs be accepted as standard and shipped with MXCuBE?

It is agreed to think of which architectural changes would be needed for incorporating both SSX and workflows, since it is expected that a lot of work will be needed on both.