BREAKING THE WALL BETWEEN OPERATIONAL AND EXPERT TOOLS

Delphine Jacquet on behalf of BE/OP /LHC, CERN, Geneva, Switzerland

Abstract

Most of the time, for the daily operation of the LHC, dedicated operational applications are used, and the expert have their own applications to control their equipment, do specific machine developments and studies. In some cases, these expert applications need to be used by the operation's team, even if not really adapted. This presentation will demonstrate why this systematic split between "operational" and "expert" applications implies a lot of disadvantages. In addition, commons problems and requirements which are easily identified in all groups are shown. Finally it will propose an alternative approach to overcome these issues.

INTRODUCTION

To control the different systems and equipment of the accelerators, two types of applications are usually available. The expert applications are designed to give a very detailed view of the system, displaying information for experts for a complete diagnostic of the system. They are most of the time developed by the equipment software experts. The choice of development's language and practices is the responsibility of the equipment groups only. Very often, beside the expert applications, simpler operational applications are developed by the operations group, exclusively in JAVA. The purpose of this applications is to give at one glance the general state of the system. The information displayed is filtered to be relevant and meaningful to an accelerator operator, and the possible actions are limited.

EXPERT AND OPERATIONAL APPLICATIONS

For a lot of accelerator systems, both expert and operational applications are available. The expert application is perfectly adapted to the experts needs, and the independent operational application is designed for OP. This situation looks perfect for everybody.

Nevertheless, this is far from ideal, the main drawback being the development and maintenance of much more applications while every group lack of manpower. Often it implies a lot of code duplication between expert and operational code. In addition, in case of low level software modification, all the involved operational applications will have to be modified. It happens that the modification is not well communicated and it breaks the operational application, and can cause accelerator downtime.

For some accelerator systems, the expert application itself is used by operators. Sometimes because there is no manpower in OP to develop a new application that would be more adapted, or because the expert tool is useful as such for some diagnostics or measurements that OP wants to perform. Some actions that were done only by experts at the early commissioning of the LHC are with time delegated to OP, and the operators use the existing expert application for that.

In one hand, having a single application presents advantages: only one application to maintain, possibility to control the actions authorized for OP by using RBAC.

In the other hand, the expert applications are not designed for OP's needs:

- They can be difficult to use for non-expert even sometime simple things as finding the relevant device name are not straightforward
- The expert applications often present too much details, this is confusing and leads to wrong diagnostic, and at the same time the information interesting for OP is lost.
- In some cases the expert applications have a direct access to the devices. It happens that the multiplication of clients in the control room is not managed properly and creates performance issues. The parameters are not always in LSA and therefore the application doesn't benefit of LSA settings management features.
- Most of the time no real training on how to use the application is provided, and there is a tendency to try every buttons to see if it solves a particular problem.

WHAT IS NEEDED TO IMPROVE THE SITUATION

Common problems

It is easy to see that all the equipment groups, accelerator physicist or operations team, have commons problems with the software development.

Flexibility and independency: for equipment experts and physicist, the applications have to be flexible and easy to update and improve. They want to choose the programming language that is best adapted, for example they use the powerful mathematic libraries of Mathematica or Python when developing software for complicated analysis.

Accessibility: being independent doesn't mean that the software is isolated from the rest of the accelerator control systems. It has to be able to access theses control systems, and the control systems have to be able to use the expert libraries even if not written in Java.

Maintenance and evolution: In all groups there is a lot of software, but not enough developers. The code needs to be maintained and evolved, either to adapt to a new hardware or simply to follow the control system upgrades.

Sustainability: often the developers are non-professional and only temporarily at CERN, and when they leave their software as to be taken over or die.

Different solutions

To face the problems, this is very common that different groups have their own solution and develop their own tools.

For example, RF expert applications are developed in LabView, because it's easy to create nice graphical user interfaces. In operations group, a GUI framework called Inspector was developed to create simple user-interfaces without programing any Java, configured only with xml files.

In the ABP group, a library was developed to access the logging database from Python (pyTimber), and in BI a library was developed to access FESA devices via JAPC from Python scripts.

Such individual initiatives are very useful and other groups are interested to use the same solutions, but face several limitations:

- The products are not scaled or adapted to be used intensively
- The developers did not foresee to maintain and evolve their libraries for everybody's needs.
- Difficult to make sure that the products are sustainable

Eventually these products will need to be taken over for long term support, with the usual problem of resources.

Common needs that should find a common solution

GUI framework

The production of user interfaces in Java is always painful and difficult and takes a lot of resources. In addition, all the applications of the control system are using different look and feel, there is no uniformity in the GUI used in the CCC. With a common framework used by all the developer, lot of time could be gained in the equipment groups, a lot of code repetition could be avoided making the code easier to maintain.

High level software layer [1]

Some features and functionality are needed in many applications, for example reading the actual tune of the LHC beam, combining some timber data into meaningful information, subscribing and combining java parameters etc… As it is done now, the same code is re-written or copy-pasted each time an application has to implements such or such functionality. Instead, all these common functionalities should be identified and implemented in a product accessible by everybody.

Consolidation of the interfaces with the control system.

The use of other languages than Java for the expert or operational applications can't be avoided, developer should keep the freedom to use what is most adapted for their need. It should then be easy to access the control system from these other languages, typically Python. The existing libraries should be consolidated and given a long term support in order to be usable by everyone.

HOW TO FULFIL THESE NEEDS EFFICIENTLY?

The experience so far with a client/provider approach between the control group and the equipment's or operations group has shown limitation in term of efficiency: CO group may be overloaded with work and unable to fulfil all requirements, while unsatisfied clients start their own development in an attempt to unblock their situation. |Experiences of more collaborative approach across groups like the LSA project have demonstrated the interest and efficiency of working together.

Therefore, knowing that groups can agree on needs and priorities, creating teams across groups to develop tools and frameworks needed by everyone would be the way to go. In this approach, free contributions to improve and evolve the software has to be allowed, providing that the code quality is respected. All groups have to agree on common principles and practices, like the way the software is structured, the tools and the rules for testing and reviews.

The benefits of working together are many. The products would have more chance to fit the needs of everybody. The code would be re-used instead of duplicated or rewritten for every application, this will save resources and time. The software could evolve in a more dynamic way if everybody contributes and it will enforce a standardisation of the code, the tools and the software structure.

A central place to get help, advice and information for any new development is needed, also for sharing experiences and expertise. This would benefit for

experienced software programmer as well as physicist developing their first application.

CONCLUSIONS

Commons problems and commons requirement concerning the software development can be identified for all the groups in the BE department. Nevertheless coming to commons solution is not as straightforward as it should. It has been already demonstrated, thanks to some projects like LSA, that a well organised collaboration across groups is very beneficial: better efficiency and quality of the delivered product, knowledge sharing and mutual trust and respect. While being aware of the organisational difficulties and the need of commitment of all the developers in every groups, efforts in this direction should continue to be encouraged and valued by the management. It will pay at longer term and mitigate the difficulties faced by everybody thanks to a common effort.

REFERENCES

[1] K.Fuschsberger “Controls – an OP perspective” 6th Evian Workshop December 2015