MANAGE THE MAX IV LABORATORY CONTROL SYSTEM AS AN OPEN SOURCE PROJECT

Vincent Hardion, Mirjam Lindberg, Antonio Milan Otero, Darren Paul Spruce, Andreas Persson, Julio Lidon-Simon, Jerzy Jan Jamroz, MAXIV Laboratory, Lund, Sweden
Piotr Pawel Goryl, SOLARIS, Jagiellonian University, Poland

Abstract

Free Open Source Software (FOSS) is now deployed and used in most of the big facilities. It brings a lot of qualities that can compete with proprietary software like robustness, reliability and functionality. Arguably the most important quality that marks the DNA of FOSS is Transparency. This is the fundamental difference compared to its closed competitors and has a direct impact on how projects are managed.

As users, reporters, contributors are more than welcome the project management has to have a clear strategy to promote exchange and to keep a community. The Control System teams have the chance to work on the same arena as their users and, even better, some of the users have programming skills. Unlike a fortress strategy, an open strategy may benefit from the situation to enhance the user experience.

In this topic we will explain the position of the MaxIV KITS team. How “Tango install party” and “coding dojo” have been used to promote the contribution to the control system software and how our projects are structured in terms of process and tools (SARDANA, GIT...) to make them more accessible for in house collaboration as well as from other facilities or even subcontractors.

INTRODUCTION

MAX IV Environment

The MAX IV Laboratory was created in 1987 under the name Maxlab. It consists today of three existing storage rings - MAX I, MAX II and MAX III. A new facility, the MAX IV, is being constructed in the north east of Lund, Sweden[1].

The different professions involved in the development of the original facility were separated into very few areas of expertise. Conversely the scientists and engineers were generalists with a wide knowledge in the fields of science, vacuum, mechanical engineering, electronics and software (among many others). Many Maxlab people know or have already developed a piece of software for their own needs.

The new MAX IV project will grow both physically and humanly and the organisation has been changed to follow this escalation. The jobs have been divided into several areas of expertise and the responsibility for each area has been assigned to different teams. The software development of the future control system is one of these areas.

In cases like this there is always a risk that the new division of responsibilities will conflict with the old way of getting things done. The time needed to develop a new feature may have been less in the previous organisation since there is no communication latency when the developer is also the user. A dedicated team on the other hand has to maintain standards and take care of integration and deployment strategies, which brings overhead but in the long term will bring tangible benefits for the maintenance work. Managing the control system software environment like an Open Source project is a way to reconcile both user and developer expectations.

The Solaris Case

The MAX-IV Laboratory has tight collaboration with the Solaris Project. The Solaris is the synchrotron radiation facility in construction in Krakow, Poland[2]. The Solaris machine will be a replica of the smaller MAX-IV storage ring and a linear accelerator built of the same components as the MAX-IV one. The concept of replicating the hardware makes source code sharing significant for both projects. Most of software development is done by MAX-IV and the code is being provided to the Solaris. The Solaris is participating in development, too[3][4]. Open software repositories: GIT at the MAX-IV and the SVN (as it is today) at the Solaris simplify the cooperation. It is expected that open source formula will make commissioning of the both facilities more efficient.

FREE OPEN SOURCE SOFTWARE

The model of Free Open Source Software (FOSS) is based on transparency and contribution. To stay alive projects need to attract users because testing and feedback is the first contribution sought by the project’s authors. Among the users some developers will contribute directly with source code to reproduce a bug or with a patch for new features. At this point a virtuous circle is created and from a certain critical mass of persons involved in a community an ecosystem is formed.

“Not only does this provide an entry point for other developers to get involved, but it also means that people will take you seriously, and recognise you're capable of delivering the goods as time goes on.”[5]

At the beginning a piece of software offers a unique feature which makes it the choice of the users. Then the user support represented by any documentation, maybe a wiki or a mailing list, is essential to keep the users drawn to the project. A low learning curve can help to accelerate the appropriation of the software by the users.

Keeping the Community & the Users

Here the image matters! To keep the users interested and committed they need to be aware about the key features, news and updates of the software. Quality is a factor to take into account for the image. The users expect

ISBN 978-3-95450-139-7

to have a minimum of stability when they download the latest stable release version of the software. Also the authors of the software need to communicate clearly to avoid forming any misconceptions.

In the same time they must be careful not to over-communicate. Imagine a project on a famous source repository that describes a killer feature on its main webpage but where the source code is just a generated skeleton! It is simply a question of balance: a good open source project communicates on the things that are done and usable but not necessary feature-complete, "release early, release often"[6]. The best way to manage a community is to walk in the user's shoes.The author of Linux itself , Linus Torvalds, considers his users as co-developers [6].

Transparency

Nowadays the transparency of the open source model is so appreciated that many licenses in different domains have copied it, like Creative Commons for the publishing domain and Open Hardware for electronics. Transparency brings the developer to the community. The ability to debug or to enhance a piece of software is the first quality that makes the developers want to use it. Then they will have a great interest in publishing their modifications if the way to contribute is clear enough. More contributions means more features, probably more users and by consequence more developers.

Project Management

Open source doesn't mean no management. Meritocracy is often applied when a community begins to appear. The developers who contribute the most can enter into the management of the project, acquiring the right to commit modifications directly into the main stream. In their turn they participate to filter out the requests of the new contributors.

MAX IV STRATEGY

The MAX IV control system is almost entirely based on FOSS components like Tango (LGPL) for the middleware, CentOS for the operating system, Taurus for the GUI layer. It seems natural to also apply the FOSS strategy to the project management.

Origin of Control System Contributions

- Staff (users with programming skills): the users of the MAX IV control system are the beamline and machine scientists. They have varying programming skills and interests. Some are able to write their own control system components for their domain, using C ++ or Python while others are satisfied with writing their own high level scripts for running experiments, using Matlab, Python etc.
- Collaboration :the software backbone of the MAX IV control system relies on collaboration with other synchrotron facilities. The TANGO distributed control system relies on contributions from all the facilities in the TANGO collaboration. MAX IV is also a contributor to Sardana and Taurus[7] and benefits from its community.
- Third party contributors: collaborating with subcontractors and suppliers requires a somewhat different effort. As far as possible MAX IV strives to maintain ownership of the code base so that it can remain open source.

User Autonomy

At Maxlab there is a strong culture of fixing it yourself – solving any problems that prevent you from getting your work done. Since there was no formal controls group in the previous organisation the users had to build most of their own software. It is a challenge to take care so we can benefit from this spirit - we must avoid squashing it or making it our enemy.

We need to show the users that what we deliver is going to solve their problem or they will go off and do their own stuff, outside the standard frameworks. Conversely what is developed by the users needs to be easy for us to maintain and integrate. The attitude we need to maintain is that competition is good for us as control system developers – it challenges us to provide good solutions and tools that really benefit the users. We must show that the Tango framework and overall insight we can offer is going to solve their issues, and that we can support them to do their own developments inside this framework. When the control system developers work together with the users our expertise in development and their domain specific knowledge will make the control system efficient for the users needs and robust for us to maintain.

Open Source vs Fortress Strategy

Of course all these advantages that can be gained from collaboration become a huge constraint in the case of a fortress strategy where you want to control absolutely how the project is managed. Any external input is seen as a negative criticism on how you direct the progress.

IMPLEMENTATION

Communication with the Users

Everything concerning the control system is open to the users, the information is available both for reading and for contribution. There is a wiki containing project information, howto:s and documentation. We aim to be supportive of users who want to contribute, by providing documentation, training and support. This is achieved through traditional training sessions, the Kits Café effort, and by providing continuous support during development with an open office policy.

Open the Market

The users should be able to choose their level of commitment. We want to supply a positive user experience, giving them a sense of influence over the solutions that are deployed. The users have the best knowledge of the systems we want to steer together and we want to pool all available resources to achieve a good control system.

The tools and programming frameworks that we use for the control system development are made available to everyone. We provide a standardised development platform to anyone with a Maxlab user account, this is achieved by hosting an NX Server[8] with the standardised MAX IV environment. It is a sandbox where the users can play with the control system and develop their own scripts and device servers. The environment includes a Tango database that can be used for experimentation, as well as all the standard control system components in use at MAX IV. Tango devices for simulation exercises are also installed, like a SimuMotor, SimuCounter etc.

Open the Source

The git software repositories are made available through Gitorious[9], a webserver that allows any user to manage their own repositories. They can decide who is able to participate in their projects and who has to send a merge request (a kind of patch from another git repository). Due to the simple branching strategy of git it is easy for anyone to begin to contribute to the source code of the control system. In the end the users use the same tools as the developers providing better support opportunities and making it simple to contribute.

The internal policy of creating unit tests for all software developed by KITS plays an important part in the contribution system by:

- providing a kind of documentation. The unit test provides an example of how to use the software. We are using PyTango to write the tests so the user is able to copy a test to create their own script. In the same spirit the user can describe a bug with a unit test.
- checking the contributions. To integrate an external contribution we need to check so that it doesn't bring any regression.

The code review of the contributions is part of the job of the KITS software team. Some practices can make that job easier:

- Clearly defined guidelines are invaluable, documenting the coding standard and the best practices that we want to enforce. These guidelines need to be easily accessible to the users, for example on the wiki.
- Planned code reviews together with the contributor.
- Focussing on the user's needs, empowering them to develop and helping maintain their code in repositories with version control.

In the end we will use the Continuous Integration system to automate the tasks related to accepting contributions. The goal is to deploy a RPM package of every piece of software to make it easier for us to maintain or change the configuration[10].

Open the Usage

The default software delivered by the control system group can be interacted with in different ways, from a graphical user interface(GUI) to a command line interface (CLI). In the same terms of collaboration and user autonomy Sardana provides tools that allow the user to create and manage their own resources, like controllers, macros and GUIs.

For the creation of new controllers, Sardana provides an API that simplifies the work and makes it easy to expand the standard catalog. The language used for the development of these controllers is Python, which means that the learning curve is quite short.

Regarding the macros, there are two possibilities to expand the standard set and collaborate with the project:

- Create a sequence of standard macros.
- Develop a new macro using the Python API.

In the case of the graphical user interfaces, Sardana provides a framework called Taurus which gives to the user a set of tools to make the development of new widgets or GUIs easier.

These GUIs can be created without writing a single line of code, just using the tools provided by Taurus, for example, a user can create a new GUI simply by configuring the different elements that the user wishes to include in the GUI.

In all the cases exposed above the user can include their resources under the terms of continuous integration.

KITS CAFÉ

Since October 2012 the staff of MAX IV Laboratory benefits from a regular and very informal event set up by the Control team. The purpose is to meet each other and to share the knowledge about the control system and any activities related to the software. The different actions described below are directly inspired by the ones used by the Linux User Group and the Agile movement.

Install Fest

The “install fest”[11] was the first event we organised to help our colleagues to install Tango on their laptops. The goal was to break the first barrier to start using the control system. This method had the advantage that we could manage the installation of plenty of computers at the same time while the users could learn and ask about the process.

Coding Dojo[12]

During this session the main objective is to involve the users in the development of some simple exercises in order to make them familiarise themselves with the subject proposed for the workshop. Also, they will learn useful lessons about the control system that will help them to contribute to the project in the future.

The workshop consists of a well defined pair programming problem that the whole audience must solve together. To arrive at that goal, there is a main workstation shared by two people, one “driver” and one “co-pilot”. The driver is the one in charge of the execution of the exercise and the only one allowed to touch the keyboard. The co-pilot is the guide of the driver and provides another point of view because he is more focussed on the result appearing on the screen.

After 5 minutes, the driver must leave his place to the co-pilot and another person from the audience must take the place of the co-pilot. This 5 minutes iteration is repeated during the two hours of the session.

As part of the audience there could be an expert that from time to time can act, just to help the attendees to progress towards the final result. But we must be clear about the goal of the session, it is not to finish the exercise, it is to exercise.

CONCLUSION - FUTURE PLANS

Achieving a control system that fulfils the users' needs of stability and flexibility with limited resources is a challenge but the model of FOSS is a strong foundation. Transparency and collaboration will make the work easier for us and for the users. Our choices for the control system software platform have been made with these qualities in mind. Tools may be evaluated and changed if needed but we are trying to promote a standard.

The implementation of the project management strategy that has been outlined above is driven by the Control team. Keeping our tools for version control, continuous integration and our documentation in good shape is easy to prioritise since it has immediate effects on our daily work. But time must also be allotted to the user focussed activities described in this article.

The KITS Café should be a reliable source of learning and support for the users. We have plans for other activities as well, such as hackathons[13] where we code together, working on a project brought by a user. These activities will then be evaluated to see what benefit they bring to the users and to the control system as a whole.

We are learning by teaching.

REFERENCES

[1] Julio Lidon-Simon, et al.; Status of the MAX IV Laboratory Control System; MOPPC109; ICALEPCS 2013, San Francisco (2013).
[2] Bocchetta, Carlo, et al.; Project Status of the Polish Synchrotron Radiation Facility Solaris, 3014 - 3016. IPAC'11 (2nd International Particle Accelerator Conference). San Sebastían, Spain (2011).
[3] P. Goryl, et al; An implementation of the virtual accelerator in the Tango control system; MOSB3; Proceedings of the ICAP2012, Rostock-Warnemünde, Germany (2012).
[4] L. Zytniak, et al.; GeoSynoptic Panel, this conference, TUPPC112 (2013).
[5] M. Saunders, Linux Format Issue 155, March 10th 2012 http://www.techradar.com/news/software/how-to-manage-an-open-source-project-1067846/2
[6] Eric S.Raymond; http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/ar01s04.html
[7] Tango, Sardana and Taurus http://www.tango-controls.org/
[8] No Machine http://web10.nomachine.com/
[9] Gitorious https://gitorious.org/
[10] Vincent Hardion, et al.; Configuration Management of the control system; THPPC013; ICALEPCS 2013, San Francisco (2013)
[11] Install fest http://www.tldp.org/HOWTO/Installfest-HOWTO/introduction.html
[12] Coding Dojo; http://codingdojo.org/
[13] Hilmar Lapp, et al.; The 2006 NESCent Phyloinformatics Hackathon: A Field Report; MEETING REPORT; Evolutionary Bioinformatics 2007:3.