HEP SW Collaboration: a few ideas...

Michel Jouvin[*1], Sébastien Binet[†1], David Rousseau[‡1], and David Chamont[§2]

[1]LAL, IN2P3/CNRS & University Paris-Sud, Orsay, France
[2]LLR, IN2P3/CNRS & Ecole Polytechnique, Palaiseau, France

May 26, 2014

This contribution represents the personal opinions of the authors, discussed with a few other collegues. It doesn't represent an official point of view of IN2P3/CNRS, University Paris-Sud, Ecole Polytechnique or France.

The source of this contribution can be accessed at https://github.com/jouvin/HEP-SW-Collab. Comments and contributions (pull requests!) are welcome.

1 Context

Next runs of LHC experiments and new generation of HEP experiments are challenging HEP software with an unprecedented data deluge. At the same time the budget constraint everywhere gives no choice but impoving HEP software performance by a factor of magnitude in the next 5 to 10 years. HEP is not unique in facing such a challenge but has a handicap: most of its computing problems are sequential in essence when most of the performance improvement in new processor architectures comes from parallelism (many cores, vector instructions. . .).

HEP benefits from a rich but fragmented software ecosystem made of many different types of packages covering simulation, analysis, frameworks... Some of them are produced and maintain by large collaborations (GEANT4, ROOT), others are developed and maintained by experiments, sometimes in common like GAUDI, and many packages have been started by an individual or a very small team. This is both a strengh and a weakness for adressing the challenges ahead of us. This is a strengh because we have a lot of people involved in SW development, covering a wide range of expertises, and this diversity is fostering innovation. On the other hand, this is also a weakness because of the risk of effort duplication at a time where manpower is limited if not in shortage.

[*] jouvin@lal.in2p3.fr
[†] binet@lal.in2p3.fr
[‡] rousseau@lal.in2p3.fr
[§] chamont@llr.in2p3.fr

Some sort of coordination between projects is the only possible answer to get the benefit of our diversity without paying the price of the fragmentation.

HEP has a rich tradition of software development, assessed by its two flagship projects, ROOT and GEANT4, now used outside the community. At the same time, we have to learn from this history that almost all the major software products now in use in the community have been started by some individuals or groups to fullfil user/experiment needs but never by a management decision. Sometimes, projects have been started as "innovation" despite the management "hostility". At the same time, those projects, in their fight to be recognized, were not always open to new ideas and took time to recognize them. With this history, HEP is not unique. The open-source model that has been so successful in the last 10 years was invented as an answer to the same problems: every project needs innovation and new ideas to evolve but a top-down managed project has difficulty to integrate new contributors and hardly benefits from new ideas. In this sense, the proposed HEP SW collaboration is a way of recognizing that HEP learnt from the open-source experience and its success to foster innovation and to get many different parties involved in the same project.

The two main challenges we are facing are an efficient access to large volume of distributed data and parallelization. A lot of expertise in these areas exists outside HEP. Even though computing models may be different, we can benefit from these expertises if we are able to liaise with these other communities that include commercial actors, in particular for Big Data. Several existing collaborations at the local level also shown that the computer sciences are interested to work with us as we are both a demanding use case and a community with already a significant expertise allowing real collaborations.

2 User stories

To have a better understanding of how the HEP software collaboration could work, we propose below a few user stories. It is not meant to be complete.

2.1 "hosted" project

I'm developing, alone or with a few others, a piece of software which I believe could be interesting to other groups. I apply to the HEP software collaboration with little more than one paragraph describing the project. The HEP software collaboration gives me its green light : at this point, the only criterion is that the project is somewhat relevant to HEP computing needs and challenges. From that point my project becomes a project *"hosted"* [1] by the HEP software collaboration. I have access to and benefits from a number of services provided by the HEP Software Collaboration. A non exclusive list could be:

- a software forge à la GitHub or Bitbucket providing both source code repositories, an issue tracker and a blog

[1] This word might not be the best choice.

- a continuous integration infrastructure (nightly build and test infrastructure in particular)
- other support for collaborative development, like mailing lists
- documented choice of recommended best practices. I don't have to abide to these best practices but I know that they can help the adoption of my project by other groups and experiments. For example:
  - choice of open source licenses
  - choice of coding rules and QA in general
  - development model
  - interfaces
  - packaging practices
- limited access to test machines (real or virtual) with different architectures

For me, the real incentive to participate to the HEP SW collaboration was the software development infrastructure that I can use for free and the visibility gained by my project. I do not expect any direct support or funding. I hope that if my project is successful in raising interest for others, it will at least help to build a more sustainable community around it or may be apply to some funding program for this kind of project. I appreciate that I'm the main person responsible for project roadmap and strategy and that I am not at risk of seeing my project killed at some point by some decisions of the HEP SW Collaboration management. I accept that if my project is really successful and becomes a component that several HEP packages rely on, I may be asked to become an "endorsed project" with a greater control exerced by the collaboration.

The threshold to become a "hosted" project is very low to promote the creativity.

2.2 "endorsed" project

I am or we are running a project that has become a corner stone of many other packages or HEP user activities or a specialized package of interest for some of the computing challenges faced by HEP communities. I have already a robust software process in place and a governance model for my project. I am ready to be more engaged with the HEP community but I want my project to continue its own life and have the ability to develop relationships with other communities, even with different/competing needs. HEP Software Collaboration infrastructure is not the main incentive for me but I recognize that what is offered is fulfilling my needs and may save some resources in my project spent to maintain our own infrastructure. I appreciate that the HEP SW Collaboration allows me to use its infrastructure even though I am not a pure HEP project.

For applying to be labeled as an *"endorsed"*[2] project, I understand that there is a formal process, conversely to "hosted" projects. This requires to demonstrate to the

[2] This word might not be the best choice.

HEP Software Collaboration Board why my project is of interest to the HEP community, addressing in particular the following topics:

- relevance
- performances
- compliance with the best practicies defined by the collaboration
- support (at least short and mid term) (e.g. the main developer is not at the end of her PhD)

I understand that becoming an endorsed project may need some adaptation in my software process to comply with the collaboration best practies and will require to integrate my documentation and user support in the collaboration framework. Being an "endorsed" project by the HEP Software Collaboration, the collaboration will be invited to participate to my project governance, without taking a full control of it. The collaboration will act as some sort of "sponsor". Even though I'm aware that the collaboration will not be a direct source of funding, it will give my project more funding opportunities, either by applying to some funding programs or by discussing with the funding agencies relevant to me to help getting the adequate level of support. In return, I expect a much increased visibility, with the positive impact on my project sustainability and my ability to attract new contributors.

Becoming an "endorsed" project may become a natural evolution of some "hosted" project but should not be the goal assigned to "hosted" projects. And it should not be a requirement to be a "hosted project" before becoming an "endorsed" project, even though it will probably the case for many projects started as R&D activites: they will becomoe an "endorsed" project when reaching maturity. In particular in the initial phase of the collaboration, it is envisionned that several existing projects have vocation to become immediately an "endorsed" projects.

2.3 Marketplace

I am a physicist and I am about to write an application for my own analysis. This application has no objective to become a standard package or even a package of general interest. I don't intend to reinvent the wheel and I'd like to find good packages for the math and the visualisation I have to do. I'm interested in finding a "marketplace"[3] where I can easily identify available packages that may fit my needs with some guarantee about their quality, their support and how well they interoperate between each others.

I heard about the HEP Software Collaboration and I found it was a good place to start my search. I was very positively surprised and I particularly appreciated:

- the marketplace area: it was very easy to search through all the projects hosted by the collaboration, well organized in relevant categories and with a rich set of tags.

[3] This word might not be the best choice.

- the overview documentation for each project that provided a clear description of the package features, who was developping it, the level of support I could expect, all in a consistent format.
- the single point of contact for all projects through the collaboration issue tracker.
- easy access to the source code.
- a clear, open-source, licensing policy.

I appreciate that if at some point there is some interest for my application, I may apply to become a "hosted" project but that there is no obligation to do it.

2.4 Computer Scientist

I am a computer science researcher, specialized in computational statistics. I would like to confront my ideas with a real physics use case. I heard about HEP analysis challenges, with both an attractive data volume (a real challenge!) and a strong expertize inside the community.

I heard about the HEP Software Collaboration web site. Thanks to the description of each project, I could find one which is both not too small and not too big with a manageable amount dependencies, that could be a good testbed for my innovative tools and methodologies.

In my search for identifying the killer project, I found particularly useful:

- the directory of the projects (marketplace) and its advanced searching capabilities, including good descriptions of the project architectures.
- ability to download some standalone demonstration code and datasets (provided by most projects), so that I could privately give a first try, before contacting the project team.
- the single point of contact for all projects through the collaboration issue tracker.

3 Goals

Based on the previous user stories, we propose to define the following main goal for the HEP Software Collaboration:

- An umbrella organisation offering a lightweight coordination between projects, promoting collaboration between them with the objective of improving software quality and visibility and of reducing duplication when it is not motivated by innovation. The HEP Software Collaboration will in particular focus on fostering innovations that will help to meet the two challenges mentioned in the introduction: the big data challenge and the ability to use efficiently the new processor architectures.

- Set up a framework that will act as an incubator for new projects, offering them the necessary infrastructure to adopt from the beginning a robust and sustainable development model, in line with HEP standards, and giving them the visibility for the project to mature and expand beyond its original initiators.

- Be a point of contact between potential funding sources, in particular HEP experiments funding agencies, and the projects endorsed or hosted by the collaboration. In particular, the HEP Software Collaboration will offer a framework to facilitate consistent, complementary and non competing projects when applying to European funding and to help with getting a coordinated funding from several sources from different countries/continents (e.g. European and US sources).

- Establish contacts with other scientific communities or parties that could be interested to contribute to HEP software or to use it, widening the scope of certain projects. This seems particularly important to liaise with the Computer Science and Data Science community that already showed interest for our challenges. Leveraging contacts existing at several institutes, the HEP Software Collaboration could allow for a more formal and wider collaboration.

There is probably a lot to learn from the successful large software foundations that emerged in the last 10 years, like the Apache or Eclipse foundations, even though we think that, at least in the beginning, direct funding of the projects by the collaboration will be marginal if not unlikely.

4 Development Model and Tools

Software development models evolved dramatically in the last decade as a result of two different processes:

- Emergence of Agile methodologies: breaking from traditional waterfall metodhologies where the iteration cycle is very slow, agile methodologies put user needs ("user stories") at the center of the development process with short iteration cycles and demonstration at the end of each development cycle. The result is a user-driven evolution of the product, one of the characteristic of the most successful software packages both in HEP and outside the community. The HEP software inventory recently made by P. Elmer pointed out that this was an important feature shared by all successful tools and packages in our community. ROOT has been an early adopter of this methodology and demonstrated it could be successful at a large scale.

- Social coding as implemented by successful platforms like GitHub and BitBucket. These platforms allow an easy aggregation of external or occasional contributors and provide tools helping the communication between project members, making the management of a project reasonnably easy even with a large number of contributors.

Based on these recognized evolution and on our current practices, the HEP Software Collaboration should define best practices regarding development model (e.g. public code access, importance of unit testing), documentation, user support. . .

To help projects benefit from these successful practices without wasting resources operating/maintaining duplicated software development infrastructures, the HEP Software Collaboration will set up and operate an infrastructure open to every project part of the collaboration, either as a "hosted" or as an "endorsed" project. Using every component of this infrastructure should not be mandatory, even though the HEP Software Collaboration could think about some incentive (gamification?) to use it.

- mailing list both for intra-project communication and for public communication (release announcements...)
- hosting of source code repositories with the largest sustainable choice of VCS. Being prescriptive for the VCS to use is generally a source of resistance... On the other hand, we could restrict the hosting to the most popular DVCS (Git and Mercurial in particular), as probably centralized VCS like SVN don't really allow to implement seamlessly the proposed models.
- continuous integration with the most popular tools (e.g. Drone, Travis, Jenkins). Again, the collaboration will have to find the right balance between the available manpower to setup and maintain the infrastructure and the diversity required to be attractive for projects without being too prescriptive.
- an infrastructure to build nightlies with the appropriate dashboards to easily identify problematic components, source of errors...
- a service to host project documentation in a wiki-like (lightweight) format. Ideally, this service should make easy collective contribution to the documentation with a peer-review (lightweight) process. A good example is GitHub Pages service based on Jekyll, based on the contents of a Git repository where contributions can be done through pull requests. Each project will have to maintain at least one page describing the project goals, who is participating to its development, and providing links to the project documentation if hosted elsewhere.
- a "marketplace" that will allow easy searching/browsing into all the projects "hosted" or "endorsed" by the collaboration. All the projects will be classified into main categories and will have tags attached to them to help identifying them. Search criteria will also include authors, level of support...

All this infrastructure will be set to foster inter-project compatibilities and enabled an efficient cross-pollination of the projects in the HEP Software Collaboration. The collaboration may want to define, as proposed best practices, the different approaches possible for project interoperabilities (see P. Mato's talk at the kickoff meeting), with a particular focus on data compatibility (a C-compatible binary layout of data structures would be the best option to ensure cross-language compatibility and interoperability.)

5 IPR

At least in the initial stage of the HEP Software Collaboration, the collaboration should not be too prescriptive about IPR policy. The only real requirement with "hosted" or "endorsed" project should be that they use an open-source license compatible with a copyleft model à la BSD or Apache2, as a too restrictive open-source license could prevent reusing the package in others.

The HEP Software Collaboration may discuss the opportunity to create its own open-source license, derived from one of well-known one, and propose it as the license for the collaboration projects, without being prescriptive again. It is expected that several projects "hosted" or "endorsed" by the collaboration will have to take into account constraints from other parties and the collaboration should make it possible rather than difficult/impossible...

Also, for some flagship projects or when request by projects, the HEP Software Collaboration could take ownership of the IPRs. This is probably a difficult topic, with potentially conflicting interests and legal difficulties, and this should not be considered as a prerequisite. This could be a topic discussed by HEP Software Collaboration governance after it has been established.

6 Governance

The HEP Software Collaboration shoud start with a governance as lightweight as possible. We should avoid at all price to build a bureaucracy or to give the impression that the collaboration governance will take control of the projects. This governance should really help to implement this bottom-up, agile approach to software development and make it clear that there is no attempt to build a prescriptive governance that will impose choices and kill innovation at a time we need it.

The collaboration governance model should be inspired by the existing software foundation, like the Apache foundation, where projects retain a strong personality and their own technical/political governance. If we want to foster the collaboration spirit between projects, there is probably no choice but decisions by consensus (or quasi-consensus) on important matters.

The collaboration should be agile in building its governance! We should not try to setup the definitive, almost-perfect, one! But rather be as minimalistic as we can at the beginning and evolve/refine it, based on the experience. Our proposal is to start with 2 boards:

- Technical Board: this should be the main board of the collaboration. It should be seen as a forum between the projects that are part of the collaboration and be open to both "endorsed" and "hosted" projects. Its main focus will be discussion and strategic/technical choices for the development infrastructure run by the collaboration and the identification of potential commonalities between projects. Its main objective should be to reach consensus in decisions but in controversial circumstances, if needed, the "endorsed" project consensus will prevail. Any really controversial issue that cannot benefit from further discussions before decisions will have to be taken or at least endorsed by the Scientific Board.

- Scientific Board: it will have the responsability to define the long term strategy, the funding implications and discuss evolution of the governance if needed. It should include representatives of the main institutes contributing to the collaborations, including small/medium size ones, representatives of the main user communities (e.g. HEP experiments) and a few representatives from the Technical Board. Consensus should be the rule for decisions.

One of the most important challenge for the HEP Software Collaboration in this first stage is to be really HEP-wide and not to appear as an appendix of CERN or even the only LHC experiments. For example, connections should be established with the Linear Collider community and the AIDA-2 project. We should really ensure that the members these boards reflect this objective to embrace the whole HEP and possibly in the future to extend to other communities interested. We should also ensure that small/medium size institutes participate to this governance.

7 Funding

In its initial stage, the HEP Software Collaboration should not aim to directly funding the projects. Projects funding will come either from the existing source for "hosted" projects or from applying to different national or continental funding programs. In particular for European funding, the HEP Software Collaboration will ensure that there is no destructive competition for funding between the different collaboration projects and will help to get the right consortium set up for building proposals and to emphasize the community interest in the project.

Parties involved in projects or in the computing activities of HEP communities will be encouraged to apply for complementary national funding when there are opportunities. This may not be necessarily for one particular software project in the collaboration but could be the occasion of a "mini-collaboration" at the national level, covering several software projects from the HEP Software Collaboration. In this particular case, the HEP Software Collaboration will ensure the appropriate coordination between the HEP-wide activities and the national ones.