Case Studies of Scientific Software Collaborations in High Energy Physics and Beyond

Peter Elmer

Department of Physics, Princeton University, Princeton, NJ 08540, USA

Abstract.

This is a partial draft of a document in preparation. The case studies are still being written, but already one can see some of the common elements.

[Draft version of 3 April 2014]

1. Introduction

Software is a critical component of the success of High Energy Physics (HEP) experiments. HEP has a long tradition of developing powerful analysis toolkits and libraries. The fact that experimental activities are mostly centered on a handful of large laboratories and the global scale of the scientific collaborations has facilitated the propagation of these toolkits/libraries to the entire HEP community.

2. HEP software packages

2.1. ROOT

[Contributions from Rene Brun (CERN) and Fons Rademakers (CERN)]

Presentation Keywords: CERN, FNAL, User-focus, Unique, Champion, Collaboration

Description: ROOT is a data analysis framework and program library, providing a large range of functionalities including Object I/O, histograms, math libraries, graphics, detector geometries, etc.

Origin: The ROOT project began in late 1994 as a collaboration between Rene Brun (CERN staff) and Fons Rademakers (private? HP?) in the context of the NA49 experiment. It built on the experience of a similar previous tool (PAW) from CERN.

Evolution: Over the first 5-7 years, there was a slow increase in contributors. Most were temporary contributors in specific periods and areas, e.g. Nenad Buncic (???), Valeri Fine(???) and some longer term, e.g. Masaharu Goto (HP/private). An important step came when ROOT was adopted by CDF and D0 and Philippe Canal (FNAL staff) became a long term collaborator. [Alice adoption of ROOT. Eventual adoption by BaBar, LHCb, CMS, Atlas.]

Notes: ROOT gained a larger user community and initially spread largely by word of mouth. Eventual adoption by large experiments cemented

Important elements of the success were: the team had previous experience with a similar tool (PAW), being innovative, being very motivated due to competition, relentless user support, letting user feedback drive the development process, reacting quickly to user feedback and a "release early, release often", being available everywhere (Unix, Linux, Windows), being Open Source, embracing user contributions, being agile and having no management overhead.

One notable aspect of the ROOT project is the fact that there was significant competition in the areas in which it was ultimately the winner. In particular several projects and technology choices (LHC++, Moose, Objectivity/DB) received significantly more institutional "top-down" support than ROOT in the early days (through about 2003).

2.2. Geant4

[Contributions from Makoto Asai (SLAC) and John Apostolakis (CERN)]

Presentation Keywords: CERN, KEK, SLAC, Unique, Collaboration, Experiment

Description: Geant4 is a toolkit for the simulation of the passage of particles through matter.

Origin: In late 1980s, there were several independent research activities for the new generation of HEP software targeting to SSC and LHC, and some of them were addressing to the detector simulation. Since then we met time to time at conferences and other occasions and exchanged results, findings and even future plans. At the CHEP conference at San Francisco in 1993, CERN group and Japanese group agreed to propose a joint R&D project to CERN/DRDC, which resulted in the launch of RD44 in 1994. This should be noted that, by these pilot projects of more than five years, we had already been confident of the choice of language (C++), tools (Roguewave), methodologies (Booch), etc., when we launched RD44.

Since the beginning we aimed for general purpose toolkit, with immediate targets to both LHC experiments (SSC had already dead at that time) as energy frontier experiments and BaBar (Belle didn't show their interest at that time) as luminosity frontier experiment. We also collected requirements from space engineering and medical communities.

After one year from the launch of RD44, we made the proof-of-concept alpha release and re-approved to proceed through the DRDC review. In 2.5 years after the first alpha release, we made the first beta release with most of the geometry modules and full set of what we currently call "standard" EM physics package. After 4 years from the start of RD44, we made the on-schedule first production-ready release with reasonable coverage in hadronic physics, which BaBar immediately adapted and started their mass production in early 2000.

We started RD44 with about 15 original authors and the number was doubled at the alpha release after one year, and went up to more than 40 when we made the beta release after 3 and 1/2 year. When we made the first public release Geant4 version 1.0 and shifted from R&D phase to the international collaboration for further development and user-support, we were about 50.

Evolution: It must be stressed that we were helped very much by BaBar experiment, which had already decided to use OO/C++. We had lots of discussions and iterations between BaBar developers (in particular the BaBar simulation framework developers and Database system developers) and Geant4 developers. Without real-time (or let me say synchronized) developments in BaBar side, we couldn't make Geant4.

At around the time of the first public release of Geant4 version 1.0 in December 1998, ATLAS, CMS and LHCb launched their internal projects of shifting their simulation engines to Geant4, and key Geant4 developers had direct involvements in these experiments and contributed to their transitions. Such direct involvements were one of the key ingredients of the rapid and wide spread of the use of Geant4. Also, extensive user supports, e.g. HyperNews, Technical Forum, user workshops and tutorials at various location/occasion, etc., are all essential part of the rapid spread of Geant4.

Notes: [To add addition notes from Makoto/John]

2.3. RooFit

[Contributions from David Kirkby (UC Irvine), Wouter Verkerke (Nikhef)]

Presentation Keywords: User-driven, Collaboration, Unique, Migration, ROOT-Distro, Champion

Description: The RooFit packages provide a toolkit for modeling the expected distribution of events in a physics analysis. Models can be used to perform likelihood fits, produce plots, and generate "toy Monte Carlo" samples for various studies. The RooFit tools are integrated with the object-oriented and interactive ROOT graphical environment.

Origin: The first version of the RooFit package (called RooFitTools) was developed in the BaBar experiment at SLAC, and initially used in 1999-2000 as part of the $sin(2\beta)$ analysis. The inital author was David Kirkby (then a postdoc at Stanford). Although it was independent by design from the rest of the BaBar software, the code was initially included in the BaBar CVS code repository.

Evolution: Wouter Verkerke (then a postdoc at UCSB?) began using the code as part of the $sin(2\beta)$ analysis in 2000 and he began contributing code in the fall. In winter 2000-2001 David and Wouter worked together to do a complete overhaul of the class design, taking in the experience of using RooFitTools to do a really complex fit the ICHEP 2000 BaBar $sin(2\beta)$ fit. This new design was finalized in March 2001, at which point it was also renamed it from RooFitTools to RooFit.

In 2002 the code repository was also split in a "core" part and a "models" part, where the latter contained classes representing functions and pdf, and the former contains all the infrastructure classes. Wouter effectively took over leadership of the project over the next few years, and the "core" part has largely stayed under his control in the past 10 years. In the "models" part there have been frequent contributions of model classes that are of general use. The user community within BaBar grew mostly by diffusion as experienced users moved into different areas of analysis.

The project was made open source in ∼2002 on SourceForge, but that probably had limited impact outside of BaBar. Integration of the RooFit package into ROOT in 2003 was likely the most important enabler in bringing ROOT to a larger user community. Initially code bundles were imported from the SourceForge repository to the ROOT software distribution. In 2005 the code was moved from SourceForge into the ROOT code repository itself. Due to the integration into ROOT, Lorenzo Moneta (CERN PH-SFT) became involved in bug fixes, etc.

More recently Manuel Schiller (Nikhef/LHCb) has been contributing to core code that is "closer to the machine" (arena-style memory allocation, improved inter-process communication).

At LHCb use of RooFit largely followed migrations of B-physics users from the B-factories. RooStats has never played a significant role here. RooFit has been used for various physics publications of D0, CDF, Belle, ATLAS, CMS, LHCb and BaBar. From forum interactions with users it is also used in many other smaller experiments, and also outside HEP (e.g. the Marine Fishery labs in Aberdeen was one surprising example).

Another big step came in 2007 when Wouter began to work with Kyle Cranmer (then BNL/Atlas), who had been asked by Rene Brun to overhaul ROOT's statistics classes. From this collaboration came the idea to base future statistics tools in ROOT on likelihood models formulated in RooFit, leading to the development of RooStats (described below).

Notes: A key choice was that it was never designed as "throw away" software or to be limited to $sin(2\beta)$ or BaBar. (No BaBar-specific dependencies, only ROOT.) Beyond this there wasn't an explicit roadmap for building a big user community.

In terms of things that would be done differently, the C++ API makes the learning curve steeper than necessary so including a python layer to decouple the API from the algorithmic layer would have been a good choice.

2.4. RooStats

[Contributions from Kyle Cranmer (New York University)]

Presentation Keywords: Experiments, User-Driven, Unique, ROOT Distro, Champion

Description: RooStats is a high-level statistical toolkit based on RooFit, and distributed in ROOT.

Origin: The project concept came about in Sept. 2005, and first code was written in 2007. The original authors were Kyle Cranmer (then a XXX at BNL?), Wouter Verkerke (Nikhef staff), Lorenzo Moneta (CERN PH-SFT staff) and Gregory Schott (???).

Evolution:

Notes: The high-level interface design was mainly done by a few people. There were a few other parallel activities, and there was a strong effort to fold them into the RooStats project even though that meant a total re-write for most of those contributions. We found a few more people to implement some of the core tools that we needed, some of them stayed with the project and some moved on afterwards. We had a lot of user feedback, and some of those people turned into contributors (either directly, or indirectly via one of our existing contributors).

Initially, there was a strategical choice that the tool would not back a specific statistical approach, but would implement all the approaches being considered within a common framework (this is similar to TMVA not choosing neural networks, but begin a common framework with common interfaces).

Secondly, we built on top of RooFit, which had a large userbase given the influx of people from BaBar into Atlas and CMS at the time. RooFit was very good for modeling, but did not have interfaces for high-level statistical tests. So that matched well.

Thirdly, Bob Cousins (UCLA/CMS) had the bright idea to setup a Steering committee for the project with members from ATLAS and CMS statistics committees to advise the developers on priorities. This gave the project some official standing and was key to broader uptake.

Next, we added some tools like HistFactory which substantially lowered the bar for entry into the framework (because it took a while for people to learn RooFit, but HistFactory presented a straight-forward user interface that was less general than RooFit, but covered 90% of the cases).

Lastly, we added two killer features:

- the workspace concept [technically in RooFit, but a product of the RooStats project] that allowed us to share/publish statistical models, and
- we had the only implementation for the new LHC Higgs Working group statistical procedure, which became the de-facto standard for ATLAS and CMS

We were aiming all along for at least the LHC experiments, and generally broader given its role in ROOT.

Simulatenously discussions between ATLAS and CMS were going in 2007 on about joint procedures for statistical analysis for (mainly) the Higgs searches, and given the existing user base of RooFit in both ATLAS and CMS at that time (largely due to an outflux of BaBar members into ATLAS and CMS) it was decided that RooFit/RooStats would be the 'endorsed' tools for statistical prescriptions agreed upon by both experiments. RooStats was also officially a collaborative project between ATLAS, CMS, the ROOT team and myself.

This development, together with the 'effective user base' that came from BaBar has lead to a quite widespread adoption of RooFit at the LHC.

While RooFit is used in LHCb, RooStats has not been used.

2.5. FastJet

Presentation Keywords: User-driven, Unique, Champion

[Contributions from Matteo Cacciari (LPTHE Paris), Gavin Salam (CERN) and Gregory Soyez (IPhT Saclay)]

Description: FastJet [10] is a software package for jet finding in pp and e^+e^- collisions. It includes fast native implementations of many sequential recombination clustering algorithms, plugins for access to a range of cone jet finders and tools for advanced jet manipulation.

Origin: The package began in 2005, with the first public version released in Feburary, 2006. The original authors were Matteo Cacciari and Gavin Salam (both LPTHE Paris at the time). Gregory Soyez (LPTHE Paris, on leave from Liege, with FNRS funding) got involved after collaborating with Gavin in developing SISCone, joined the development team in 2007, and has given a hugely significant contribution since then. The aim of the package was to provide better (and better implemented) clustering algorithms. The package was a vehicle to deliver them to the phenomenological and experimental communities.

Evolution: The package gained a larger user community as experiments bundled it into their software, and it also became the default jet clustering package used by other codes, e.g. Rivet, Delphes, etc. The work has been supported, from 2006 theough 2013 by two successive grants from the French Agence National de la Recherche (ANR) for work on jets that meant, though not exclusively, FastJet development.

Notes: The authors gave numerous talks about the algorithms and the code to highlight the associated physics issues. Adoption was probably driven by acceptance of the need for better and more standardised algorithms, ease of use, and marked advantage with respect to previous software in terms of speed. The only change in strategy that developers noted is that they should have perhaps tried to get one or two more people involved in the development in the past few years, so as to be able to be more reactive in terms of maintenance, development and user support, in a context where the pace of activity has picked up markedly after the LHC turn on.

Used by: Phenomeonlogical community, CMS, ATLAS, ALICE, LHCb, H1, ZEUS, STAR and also, more marginally, CDF and D0

2.6. EvtGen

[Contributions from Anders Ryd (Cornell) and David Lange (LLNL)]

Presentation Keywords: User-driven, Unique, Migration

Description: EvtGen (originally Evt) is an event generator for B-physics, implementing many detailed models important for the physics of B-mesons. This includes semi-leptonic decays, CP violating decays and it produces correct results for the angular distributions in sequential decays, including correlations.

Origin: The earliest version of EvtGen began in 1993 or 1994, when Anders Ryd (Institute at the time? UCSB?) was a graduate student in Ithaca working on CLEO II, and was motivated by the need to model semileptonic decay backgrounds for his thesis. This code was eventually completely written and David Lange (UCSB at the time), also a graduate student became involved in 1995.

Evolution: The code was brought to BaBar at SLAC when Anders switched experiments. In CLEO it had only handled semi-leptonic decays, but Anders put in significant effort once in BaBar to turn it into a full event generator for B-decays. Once it became the only reasonably modern/C++ B-physics decay tool, other experiments were interested, too. Contributions came from people working in CLEO, BABAR, CDF, and LHC-B people primarily via new physics models. The EvtGen framework makes it easy to contribute independent pieces of code given some basic information on how to do it. Perhaps o(20) peole contributed in total in this period (which period?). [Something about how Anders/David no longer supported it and it wandered in the woods for a while? I can follow up with the Warwick people to get there info as to how/why they picked it up.]

Notes: The main thing that the authors would do differently would be to look for a more formal support (funding) model once the user base was established.

Used by: CLEO ??, BaBar, CMS, Atlas, LHCb, CDF, Others?

2.7. FroNTier

[Contributions from Barry Blumenfeld (JHU), Dave Dykstra (FNAL), Liz Sexton-Kennedy (FNAL)]

Presentation Keywords: FNAL, Experiment, Collaboration, Unique, Migration, Standards, Adoption

Description: FroNTier [5] is is a simple web service approach providing client HTTP access to a central database service. It is most typically used as a framework to deliver conditions (i.e. calibration, alignment, etc.) data to processing clients worldwide. Because of the read-only nature of the data, Squid proxy caching servers are maintained near clients and these caches provide high performance data access.

Origin: FroNTier started as a project within the CDF Offline project in 2003. The goal of the project at the time was to provide a front end interface to multiple DB backends through network protocols such that the client code would not even have to know what backend service was being used. The initial design and development was done by developers from FNAL and JHU, driven by CDF use cases. It was intended as an explicit collaboration between D0 and CDF, although CDF was the lead.

Evolution: After the initial developments for CDF and D0, most of the original development team dispersed. The FroNTier software was then brought into CMS by Lee Lueking (FNAL) when he moved from D0 to CMS in 2005. The current developer, D. Dykstra (FNAL), joined the project in 2006 who has been largely responsible for the evolution for the LHC experiments (CMS and eventually Atlas). Atlas originally had a conditions model built on Oracle streaming, but chose in 200x to adopt FroNTier because YYYY. (Who can answer this?)

Notes: Choosing standard (not HEP-specific) web technologies (Tomcat, Squid) was a key choice made early in the project and is felt to be an important ingredient to gain acceptance. Very responsive developer and user support is also seen as a key to the success.

2.8. Gaudi

[Contributions from Pere Mato (CERN)]

Presentation Keywords: CERN, Experiment, User-Driven, Adoption

Description: The Gaudi project is a open project for providing the necessary interfaces and services for building HEP experiment frameworks in the domain of event data processing applications.

Origin: The Gaudi project was begun in 1999, with the aim of developing the even processing framework for LHCb. The initial contributors were G. Barrand () , I. Belyaev () , P. Binko() , M. Cattaneo (CERN staff) , R. Chytracek () , G. Corti () , M. Frank (CERN staff), G. Gracia () , J. Harvey (CERN PH-SFT staff) , E. van Herwijnen (), P. Maley () , P. Mato (CERN PH-SFT staff), S. Probst () and F. Ranjard ().

Evolution:

Notes: ATLAS contributors came later after ATLAS made the decision to adopt Gaudi (as Athena): David Quarrie (LBNL), Paolo Calafiura (LBNL), Charles Legget (LBNL), et al. Other LHCb contributors Marco Clemencic (CERN), etc. New GaudiHive contributors Danilo Piparo (CERN), Benedikt Hegner (CERN), Daniel Funke (CERN),...

How did the additional contributors get involved? People moving to other obligations that were replaced, new functionality added, etc.

- How did the package gain a larger user community? (Or become used by additional experiments/labs/groups?) The software is there and does something useful for them without having to re-invent everything from scratch. They provide feedback and contributions in terms of code.

- What do you see as the key choices/features/strategies that enabled the software to be used by a larger community? Did you aim for a larger community intentionally or did it "just happen"? The initial main goal was not to 'sell' a product to a larger community. The goal was to develop the applications needed for the LHCb experiment. Obviously making the software flexible and 'well architected' helps on the initial goal but also enables the possibility of re-using the code. The re-use always came from the client side getting interested. We had users that we were not even aware (eg. BES3, Daya-Bay).

Used by: LHCb, Atlas, Daya Bay (?), Glast/Fermi, BES3(?)

2.9. FairROOT

[Contributions from Mohammad Al-Turany (GSI)]

Presentation Keywords: GSI, Experiment, Adoption, Champion

Description: The FairRoot framework is fully based on the ROOT system. The user can create simulated data and/or perform analysis with the same framework. Moreover, Geant3 and Geant4 transport engines are supported, however the user code that creates simulated data do not depend on a particular monte carlo engine.

Origin: The project began in 2003 as CbmRoot as software for CBM experiment at GSI, in very close collaboration with the CBM collaboration. Mohammad Al-Turany (GSI/IT) and Denis Bertini (GSI/IT) as the primary developers.

Evolution: People from the CBM collaboration start using and contributing to the software, Florian Uhlig (GSI/CBM Postdoc), Volker Friese (GSI/CBM, Staff member) and Ilse Konig (GSI/Hades/CBM, Staff member),all CBM collaborators, contributed to many features.

In 2006 Panda was still trying to use BaBar Software for more than two years (2003-3006). However as BaBar officially stops, the Panda collaboration realise that they have nobody who could maintain and develop the BaBar code. So they had to decide about developing everything themselves from scratch or search for synergy within the FAIR project and the HEP community. An agreement was achieved between PANDA, CBM and GSI-IT to take CbmRoot as base and move it to a more general framework: “FairRoot”. The core group was then extended by 2 people (Florian Uhlig for CBM and Radoslaw Karabowicz for Panda). Then more developers from Panda start contributing to major parts of the software (Tobias Stockmanns, FZJ, Staff member, Soeren Lange, Giessen, Staff member, Olaf Hartmann, SMI, staff member).

Notes: The fact that the two largest experiment at GSI/FAIR start Using the same framework and contribute to the manpower in the core group encourage the rest of the smaller collaboration at GSI/FAIR to look closely to the software and start negotiating the support. The R3B collaboration also join the effort and a new position for R3b/IT was created (Dmytro Kresan, Staff member). After that experiments from outside GSI start also considering the framework.

Key choices the enabled the use of the software by a larger community were:

- Release early, release often
- A stable core with restricted rules + Experiment code with less restricted rules (depending on the experiment!) guarantee a stable software with generic features and the experiments are still free to do their own stuff which could violate the rules at the beginning (because of died lines and so on). In many cases this is corrected later but in case of problems it only effect a single experiment.
- Variety of the experiments with different requirement on event sizes and rates push of a modular core that can handle many different cases
- The strong participation of the different experiment to implement own solution but keeping in mind that it would be used by others improve the quality of the software

Things that would be done differently:

- Clean APIs from the beginning
- Stronger coding rules and automatics checks to these rules
- Would go earlier to message queues and multiple languages

2.10. TMVA

[Contributions from Andreas Hoecker (CERN), Joerg Stelzer (CERN), Eckhard von Toerne (Bonn), Jan Therhaag (Bonn), Peter Speckmayer (CERN)]

Presentation Keywords: CERN, User-Driven, Unique, Collaboration, ROOT-Distro

Description: TMVA is a toolkit for multivariate data analysis with ROOT.

Origin: Development of TMVA began in 2005. The founding developers were Andreas Hoecker (CERN), Joerg Stelzer (CERN), Helge Voss (MPI Heidelberg), and Kai Voss (CERN at that time, subsequently left HEP). They were all members of ATLAS and LHCb, but TMVA was not part of dedicated SW development for an experiment. From the start it was developed as a tool for broader HEP application. It built on the idea of parallel training and evaluation of MVA-based classification in HEP that had been pioneered by the Cornelius package, developed by the Tagging Group of the BABAR Collaboration.

Evolution: The core TMVA team grew by Peter Speckmayer (CERN, joined end of 2006) and Jan Therhaag and Eckhard von Toerne (Bonn), who joined in Sep 2008. Throughout the years 23 other researchers (physicists, mathematicians, computer scientists) contributed SW to TMVA. Most contributed to the methods, some to infrastructure. Some got involved out of interest because they were direct colleagues of us, others were students (Summer students, parts of PhD projects), others heard about TMVA and wanted to contribute because they are interested in MVA techniques. Because external contributors were vital to the development of TMVA, we were very open to them from the beginning, acknowledged via co-authorship on the TMVA Users Guide [12].

We started the development and distribution on the Sourceforge platform, so there was no ROOT support at the beginning. Interest in the community rose slowly, but steadily via the word of mouth. Once TMVA was integrated in ROOT 5.11/06 (July 2006) and followers, the user base increased rapidly.

Notes: One key choice was was the intention from the start to develop TMVA for a large user base. Possibly the main reason for its success was the easy-to-use interface, paired with sufficient performance and good user support.

Another key choice was the declared goal to allow the community to move away from "religious struggles" over which MVA method performs best to having a tool that provides a fair comparison of all possible methods. This way, analysts do not emphasise anymore (at least not so much) which MVA is used, but rather whether or not MVAs are employed at all. In some sense one could argue that the seamless application of MVAs via TMVA has rendered the underlying technology invisible to those not interested, and helped make the use of MVA methods for classification and regression problems widespread and "normal".

As to what would be done differently a second time around, the main paradigm behind the development of TMVA was simplicity for both the users and the machine learning tools developers. These goals led to the development of a somewhat over-rigid framework. In particular, the complexity of the internal data handling makes developments difficult for non-experts. A profound internal redesign (mostly hidden to the users) became necessary with TMVA 4, which improved the internal structure.

2.11. IgProf

[Contributions from Giulio Eulisse (FNAL)]

Presentation Keywords: Experiment, User-driven, Champion

Description: IgProf is a tool for measuring and analysing application memory and performance characteristics.

Origin: IgProf was initially developed in the context of the CMS experiment in 2003. The authors were Giulio Eulisse (then NEU) and Lassi Tuura (NEU). [Funding?]

Evolution: The initial implementation was used by a small group of people developing code within CMS in the first years. After CMS began to rewrite its software in 2005, the code base began to grow much more quickly and monitoring and improving code performance became an urgent issue. This drove additional developments such as a navigable web-based report, which made it significantly easier for someone to share results with other people. This included the important capability to point at specific performance issues in the report (i.e. specific locations in the stacktraces) by URL. Later in 2008-2009(?) development was done to support efficient profiling of 64bit applications, both in IgProf and in an external stack unwinding library (libunwind).

[Mikko] Recent developments have included expanding support to cover the ARMv7 architecture.

Notes: Key technical choices included working entirely in userspace, not requiring modifications to code and the features of the web-navigable report.

Used by:

2.12. LCG AA Projects - POOL, CORAL, COOL, etc.

[Contributions from ()]

Presentation Keywords:

Description:

Origin:

Evolution:

Notes:

Used by:

2.13. dCache
[Contributions from Patrick Fuhrmann (DESY)]

Presentation Keywords: FNAL, DESY, Collaboration, Standards, Champion

Description: dCache [6] provides a system for storing and retrieving huge amounts of data, distributed among a large number of heterogenous server nodes, under a single virtual filesystem tree with a variety of standard access methods.

Origin: dCache began in ∼2000 as a collaboration between FNAL and DESY. A cache layer was needed between tape and workernodes, as at that time the tapes got bigger and faster and the workernodes got slower, blocking tape resources for too long. Charles Waldman (FNAL) and Patrick Fuhrmann (DESY) were the original authors. Charles came to DESY for some time, to facilitate collaboration and get up to speed quickly.

We started with the HERA experiments at DESY (Zeus, H1, Hera-b and Hermes) and experiments at FNAL. Initially with CDF followed by (I think) Minos and some time later the Sloan Digital Sky Survey if I'm not mistaken.

Evolution: Initially dCache wasn't seen a distributable software package but instead a service provided by FNAL and DESY. When dCache approached WLCG it became a software package. This was a painful process for the dCache collaboration and our customers. This was because a very small team had to support more and more sites with extremely diverging setups. About the support: DESY essentially was responsible for Europe and FERMIlab for the US. However, later we introduced a consolidated support process with headquarters at DESY, distributing requests according to the topic of the support request.

After dCache became popular within WLCG (mostly the Tier I's) NDGF joined the collaboration after they agreed to install a Tier I for WLCG in (∼2004). About 2008 SNIC (Swedish National Infrastructure for Computing joined) joined. 2011 the University for Applied Sciences (HTW) Berlin joined. They provide 1 - 3 students per year to implement smaller things in dCache. With HTW we usually launch German national projects funded by the Ministry for science and education.

Notes: Attracting additional contributors was quite simple. Whenever someone needed dCache and required features which were not in the plan of the core partners, they joined and implement what they needed. This is a bit different for the HTW Berlin. The computer science institute of the University was looking for a software project where students could work on real world issues.

The user community grew because:

- With having large WLCG dCache installations at the big labs (WLCG Tier I'1), it was somehow natural to use the already existing dCache installations for other data intensive groups. (Lofar at Juelich and SARA, small communities at PIC, etc)
- We joined German national (DGRID) and European projects (EMI, EGI) and with that the number of communities increased automatically.
- Communities and Companies found the dCache webpage as well as our presentations at Conferences and contacted us for dCache or dCache sub-components. (e.g. Credit Swiss, the Australian National Library and others)

Key choices in growing a larger user community were:

- We were actively aiming for other communities.
- The magic is to use standards wherever possible. This guaranties funding and attracts communities. With the implementation of NFS 4.1 and httpWebDAV, we now serve a lot more communities at DESY, FERMIlab and other sites. The difference is that they don't know that they are using dCache (Which is the idea of standards). Now, as we are implementing CDMI (Cloud Storage) we became part of the EGI "Federated Cloud"

working group and the German (Large Scale Data Management and Analysis) LSDMA project, which gets us in contact which even more communities.

- Moreover we participate conferences and User Forums (e.g. EGI) to learn about particular requirements from new communities.
- Another key feature is the integration of new authentication mechanisms (kerberos, User Password, SAML) and the possibility of mapping between the different identities of the different authentication mechanisms.
- On top, the ability to enable data migration (spinning disk, SSD, tape) makes dCache unique.
- Having two partners (and very soon three) was important to enable review of what was done by others, which was a great help. Partners in both Europe and the US was also a significant advantage over other projects.

2.14. Xrootd

[Contributions from Andy Hanushevsky (SLAC)]

Presentation Keywords: Experiment, Collaboration, User-driven, Champion, Adoption

Description: Xrootd [16, 17] provides high performance, scalable and fault tolerant access to data repositories of all kinds.

Origin: The Xrootd project originated in the BaBar experiment at SLAC. It traces its roots to initial work $\sim$ 2000 by Andy Hanushevsky (SLAC staff) to add load balancing capabilities to the Advanced Multithreaded Server (AMS) of Objectivity/DB, an object database technology. In 2002 the BaBar experiment decided to abandon Objectivity/DB in favor of a more scalable file-based solution based on ROOT persistence as part of the implementation of a new computing model. The Xrootd project was born, with an aim of providing the high performance, scalability and fault tolerance which the AMS and other technologies lacked. Although the Xrootd technology was developed from scratch, and has no specific dependencies on the ROOT toolkit or its file format, the name was chosen to indicate the intention to replace a much simpler file access daemon (rootd) which was then included with ROOT. Alvise Dorigo (INFN) and Fabrizio Furano (INFN) joined as developers.

Evolution: In late 2003, when a first working version of the Xrootd system was available and being tested at SLAC and elsewhere, discussions began with the ROOT team about replacing rootd in the ROOT distribution with Xrootd. (To facilitate migrating to "root://" protocol to Xrootd from rootd, a simple backwards compatibility was implemented to spawn a rootd from xrootd when an old-style client access happened on Xrootd.) Gerri Ganis (CERN staff, and part of the ROOT team) joined the team to work on the authentication libraries. BaBar commissioned the system and deployed it for production use in multiple computing centers (SLAC, CCIn2p3, RAL, as well as many smaller university sites) during 2003-2004.

(Something about additional collaborators, Castor2 adopting Xrootd as an alternative to to RFIO. Something about EOS and other systems like dCache/DPM adding Xrootd doors.)

In 20XY a lightweight collaboration was formed, which fostered additional collaborators. In 2011, a US NSF funding grant ("Any Data, Anytime, Anywhere") was awarded with an aim to further develop and deploy Xrootd systems that aggregate storage over multiple data centers into a "data federation". This further enhanced the collaboration by adding three additional institutions (UCSD, UNL, Wisconsin).

A lightweight collaboration mechanism (no governance) was formed in 201x.

Notes:

Used by:

2.15. CernVM-FS
[Contributions from Predrag Buncic (CERN)]

Presentation Keywords: CERN, R&D, Champion

Description: CernVM-FS (CVMFS) [4] is a network file system based on HTTP and optimized to deliver experiment software in a fast, scalable, and reliable way, using caching of both files and file metadata. It is used by several (all?) of the LHC experiments.

Origin: In 2007 CERN PH-SFT organized an official R&D project on virtualization (CernVM) which began in 2007 with Predrag Buncic (CERN PH-SFT staff) as the project lead. As the filesystem plays a key role, this was an important part of the project and thus CernVM-FS was a spin-off from the wider virtualization project. It went through several iterations:

- A first implementation was inspired by the GROWFS plugin for Parrot and was adapted for use in CernVM as a fuse module by Leandro Franco (CERN PH-SFT fellow)
- First complete rewrite that allowed nested catalogs and removed scalability issues of growfs was done by Jakob Blomer (CERN PH-SFT PhD student, later fellow)
- Jakob then reworked the code once again to address all requirements for deployment on the Grid (versions 2.0.x and 2.1.x)
- Currently, Rene Meusel (CERN PH-SFT summer student, technical student and later fellow) is working on server side of 2.1 version

Evolution: Originally, this was meant to be the key component of CernVM by which I wanted to distinguish our VM from everything else on the market. Our primary customers were ATLAS and LHCb. As R&D project we organized a few workshops and I made the presentations on various occasions such as GDB, CHEP.. In the process I got into arguments with Hepix people about suitability of this 'minimal approach' to virtualization. One day, Ian Collier (T1 manager at RAL), an active and respected member of Hepix and GDB communities, came to me and suggested that it would perhaps be a good idea to deploy CVMFS outside CernVM to solve their scalability problems with NFS shared software area at RAL. I was initially hesitating since giving out CVMFS would mean loosing the essence of what makes CernVM different but then I agreed under the condition that we call it officially CernVM File System. The solution worked like a charm at RAL and this was important in convincing other sysadmins and sites to adopt the tool.

Over time there have been substantial contributions from other parties. For example, libcvmfs library was contributed by Dan Bradley (Wisconsin), allowing CMS to use CVMFS via Parrot, to support use of opportunistic resources where local system configuration/modifications are not possible. Integrating this change required refactoring of the CVMFS code, but this was seen as beneficial to add more benefits for a wider community. The Sanger institute (https://www.sanger.ac.uk, genomic research) has contributed support for use of an alternative union file system on the installation boxes. The union file system is what provides write access to cvmfs repositories. Instead of relying on the AUFS kernel patches, this contribution allows to use kernel-maintained OverlayFS. In addition, contributions are currently being merged from CERN OpenLab (Seppo Heikkila) in order to use S3 as a storage backend for cvmfs data.

Notes: The original objective of CernVM was to provide a common solution for all experiments and thus CernVM-FS had the same objective. In addition to all of the LHC experiments, it was made to work early on also for NA61, ILC, H1, etc. Since the setup was general enough to start with, it was easy for other communities to adopt it.

There is nothing in particular that the developers would aim to do differently this time. It was in many ways this was very successful project where fortunately very few mistakes were made. The only difficult moments were when a large scale deployments on the grid started uncovering the rare bugs that we did not see in our usual VM deployment and we realized that we did not have good enough release testing procedure. This was then gradually improved and now there are strict performance and stress tests in place with staged deployment process that assures software quality. These are managed by CERN PH-SFT.

Another important ingredient was a great team in place to support CernVM and Jakob who offered excellent support for CVMFS. Essentially, every problem was quickly analyzed and fixed in matter of hours the patch release was available. Also, no e-mail was left unanswered which was very important to assure new users. Most important was maintaining a good relationship with the grid site administrators. We also agreed to carry out external security review which came out very positive and this all helped to convince people to adopt CVMFS as the tool for software solution. Still, the most important reason was performance - ATLAS did not need any more to restrict the number of jobs on many core boxes, LHCb jobs startup time was much faster with CVMFS then on any other file system.

2.16. Indico

[Contributions from ()]

Presentation Keywords:

Description: The Indico tool allows you to manage complex conferences, workshops and meetings.

Origin: Originally an EU-funded project, involving a collaboration of SISSA (Trieste), U. Udine, TNO (NL), UvA (NL) and CERN.

Evolution:

Notes:

Used by:

2.17. CLHEP

[Contributions from Bob Jacobsen(Berkeley/LBNL), Leif Lonnblad (Lund)]

Presentation Keywords:

Description: CLHEP [9] is a set of HEP-specific C++ foundation and utility classes such as random generators, physics vectors, geometry and linear algebra. CLHEP is structured in a set of packages independent of any external package.

Origin: The CLHEP project began in 1990 with Leif Lonnblad (then at CERN) as the primary author. In that period, there had been work using C++ and OOP on MC++ [14] (a toolkit for event generation) and Gismo [13] (a detector simulation toolikit). At the CHEP conference in 1992 in Annecy there were discussions about the need for a C++ class library, with the idea that a few general classes from MC++ could be the starting point for the library. A zereoth order version grew from there and the project was presented as such in 1994 [15].

Evolution:

Notes:

Used by:

3. Other scientific software packages
[See CHEP13 “Taxonomy of Scientific Software Applications” talk]

3.1. R
3.2. Gromacs
3.3. Blast
3.4. astroPy
3.5. HDF5
3.6. LAMMPS
3.7. CCP4

4. Conclusions

Acknowledgements

References

[1] Brun R, Couet O, Vandoni C and Zanarini P 1991 PAW users guide, CERN Program Library Q121.

[2] Brun R and Rademakers F 1997 ROOT - An object oriented data analysis framework Proc. AIHENP'96 WorkShop, *Nuclear Instruments and Methods in Physics Research* **A 389** 81-86.

[3] Agostinelli S et al 2003 Geant4 - a simulation toolkit *Nuclear Instruments and Methods in Physics Research* **A 506** 250-303

[4] `http://cernvm.cern.ch/portal/filesystem`

[5] FRONTIER: HIGH PERFORMANCE DATABASE ACCESS USING STANDARD WEB COMPONENTS IN A SCALABLE MULTI-TIER ARCHITECTURE. By S. Kosyakov, J. Kowalkowski, D. Litvintsev, L. Lueking, M. Paterno, S.P. White (Fermilab), Lauri Autio (Helsinki U.), B. Blumenfeld, P. Maksimovic, M. Mathis (Johns Hopkins U.),. FERMILAB-CONF-04-367-CD, Sep 2004. 4pp. Presented at Computing in High-Energy Physics (CHEP '04), Interlaken, Switzerland, 27 Sep - 1 Oct 2004

[6] `http://www.dcache.org`

[7] `http://proj-gaudi.web.cern.ch/proj-gaudi/`

[8] `http://fairroot.gsi.de`

[9] `http://proj-clhep.web.cern.ch/proj-clhep/`

[10] `http://fastjet.fr`

[11] `http://tmva.sourceforge.net`

[12] `http://tmva.sourceforge.net/docu/TMVAUsersGuide.pdf`

[13] W. Atwood et al, The Gismo Project, C++ Report 5 (MarsApril 1993, 38, SLAC-PUB-6135

[14] L.Lonnblad and A. Nilsson, Comput. Phys. Commun. **71** (1992) 1

[15] L.Lonnblad 1992 CLHEP - a project for designing a C++ class library for high energy physics *Comp. Phys. Comm.* **84** 307-316.

[16] Dorigo A, Elmer P, Furano F and Hanushevsky A 2005 XROOTD - A highly scalable architecture for data access *WSEAS Transactions on Computers* **4.3 (2005)**

[17] `http://xrootd.org`

[18] Brown D et. al. 2004 The new BaBar Analysis Model, *Proceedings of Computing in High Energy Physics (CHEP 2004)*, Interlaken

[19] Elmer P 2004 BaBar Computing - From Collisions to Physics Results, at Computing in High Energy and Physics (CHEP04) (Interlaken)