==

Morning Session

Names:
Robert Kalescky
Marc Paterno
Henry Schreiner
Jeff Carver
Mike Sokoloff
Miron Livny(ML)
Matt Zhang (MZ)
Benedikt Riedel (BR)
Dick Greenwood
Kaushik De (KD)
Don Petravick.

S2I2 HEP/CS Workshop Questions

Please write your ideas here for discussion questions for the Thursday sessions. (Including your name is optional.)

How to align the CS research mechanisms (3 year grants, student developers, conference pubs) with the longer term needs of big science (30 year projects, production software, journal publications)? How do align such collaboration with the life-cycle of Phd thesis in CS? (*ML*)
(is it not substantially about networks of people and incentives -- DLP)

What role common data formats play in fostering collaboration with computer scientists. I.e. moving away from ROOT formats to open formats, those used e.g. in other data driven sciences? (R. Gardner)

- What makes a format "open"?
    - Should be independent of a framework
    - ROOT is very HEP-centric (BR)
    - HDF5 (already used in some experiments for higher level data), netCDF (popular in meteorology), zipped BSON, boost archive formats, etc. are used across disciplines and in industry (BR)

Which of the many specialities in CS is most useful for HEP? (consider: machine learning, software engineering, computer vision, programming languages, networks, databases, complexity theory, robotics, human computer interaction, systems, architecture, ...) (N Ernst)

- This isn't an answer in full, but some examples of where applications of the above are currently being used in HEP (M. Feickert)
    - Machine Learning (DOI's and e-Prints): 10.1038/ncomms5308, arXiv:1609.00607, arXiv:1612.01551
    - Machine Learning and (some) Computer Vision: 10.1088/1748-0221/11/09/P09001, arXiv:1611.05531
    - Programming languages (incomplete, others please add): C++ (ATHENA, ROOT), Python (PyROOT, applications of scikit-learn)

How do we reward grad students/post-docs/more senior physicists for writing good (sustainable) code and for cleaning up the existing code base (M. Sokoloff)?

Some tools are common, others are specific. How do we identify the common set of tools for the HL-LHC for us to focus on while keeping requirements late (just-in-time/agile)? (KD)

- Discussion - do we need incentives, to promote better/common code development?
- Discussion - maybe we should let good ideas rise to the top? The best ideas and practises should be encouraged, and they will win.

How are the Software Engineering needs in HEP distinct from the Computer Science needs? (JC)

What are examples of successful CS-HEP collaborations, and what properties have driven their success? How do we measure success of such a collaboration? (*ML*)

What CS research challenges exist within HEP where CS researchers could contribute to HEP but also receive recognition for their work in the CS community? (D. Katz)

How can a software institute provide "consulting services" to help experiments review their architectural decisions, the state of their legacy code, their documentation and code review processes, etc. in the context of "best practices" identified by the computer science/software engineering community (M. Sokoloff)?

How can we make sure code is modular and well-documented enough, where a new developer (grad student), when given a specific task to work on, can immediately make useful additions without having to understand the entire code framework? (MZ)

- What if we add "And without further complicating the software" to this question

How do we formulate the HEP challenges in terms of CS principles? (*ML*)

How to align the CS research mechanisms (3 year grants, student developers, conference pubs) with the longer term needs of big science (30 year projects, production software, journal publications)? How do align such collaboration with the life-cycle of Phd thesis in CS? (*ML*)

How specifically is the current software and its development strategy deficient? There has been discussion of new data and compute time requirements and broad estimates, however a better understanding of these requirements and the compute capabilities that will be available may help to guide the direction that solutions can take in fulfilling those requirements. (R. Kalescky)

What role common data formats play in fostering collaboration with computer scientists. I.e. moving away from ROOT formats to open formats, those used e.g. in other data driven sciences? (R. Gardner)

Could HEP describe some long-term challenges that don't need to be solved immediately, but that CS people could go off and think about? (D. Katz)

How to engage a broader slice of the CS community and make scientific computing more respectable within CS circles? (A commonly heard complaint in CS: scientific computing is a “niche" research area.)

What CS technologies, techniques, and trends could the HEP community adopt, rather than doing everything internally? (Keeping in mind the long time scales and production needs of HEP.)

How could an HEP software institute facilitate interactions between the CS and HEP communities?

What are the incentives for such collaboration for HEP people? For CS people? For non-CS people? E.g. recognition, funding, publications, students, new problems to solve, new places to apply technologies, new solutions to current problems, pride in working on a global-scale problem.

How can we create “crystallization points”, shared artifacts that allow the encoding of tools and practices of the two communities and that can be improved over time? (Successful examples are wikipedia, linux kernel, docker registry)

- Along these lines DIANA HEP (in particular Kyle Cranmer and Lukas Heinrich) is working on some of this in the form of preserving analyses with use of Docker (c.f. RECAST). Though Docker has some problems with HPC envs(?) (M. Feickert) This article has useful pointers to efforts making software containers viable for HPC (C. Maltzahn).

Re: data "privatization" in Frank's presentation (commentary here from R. Gardner):

- "Public" and "private" have different meanings in collaborations. In Frank's talk "public" meant datasets available collaboration-wide, e.g. public to the collaboration, and private meaning the end-stage datasets specific to an analysis and not necessarily registered in the experiment's official catalogs, even after publication (Frank correct me if this is wrong).

- The implication of this if true is that it prevents full reproducibility of a published result; there's a little "black hole" of data (and potentially software) between the collaboration datasets and the final plots and figures data.   (n.b HEP typically runs two experiments that probe the same phenomena, is this not a kind of substitute for the topic)
- In the future we want "published" results to come with published data, and software, allowing for reproducibility by future analysts.
- How can CS help here? (many projects out there - are they addressing problems relevant to the scale and timeframe of HL-LHC?)
    - Check out the Popper convention -- this effort views reproducibility as a software engineering problem (the dev/ops community has already sophisticated tools to reproduce behaviors in a continually evolving software artifacts) and is partly funded by the Big Weather Web NSF SI2-SSI project. It's a convention, not a particular tool set (although tools need to be "scriptable"). So it should be applicable to a wide variety of domains. It's also scalable because it uses git for provenance and the git repositories include large resources by reference.
- (D. Katz: see https://mpsopendata.crc.nd.edu for some work in this area)

How can CS help build frameworks/organization/processes that incubate software from the S2I2 into open source projects? Do organizations like AMPLab – UC Berkeley which build tools that have strong industry-coupling and support apply? How do we avoid building HEP unicorns, but technologies that are potentially of broad interest and with large, open development communities? (R. Gardner)

- Check out Center for Research in Open Source Software (CROSS) at UC Santa Cruz. The research project portfolio is currently skewed towards storage systems but the goal is to create a career path for Ph.D. students to become open-source software leaders. The membership agreement and the bylaws are strongly inspired by NSF's U/ICRC concept.
- Red Hat also provides great resources for open source in education.

What open source tools supported by industry can the HEP community use to solve its problems? Some good examples are OpenStack and LLVM (Spark, Tensorflow, also various commercial "AI as service" offerings, see IBM Watson for example), are there more out there? (L. Sexton-Kennedy)

What are the challenges of today's HEP software, and its adoption and scalability on emerging hardware or OS virtualization software that one has to think beyond those? What pieces of this software CS-HEP collaboration can be sliced for CS community to work on with a clear definition of expectations? (Amit K.)

There are many different types of HEP software. How do our issues differ, depending on whether we are talking about "infrastructure code" (e.g. event-processing frameworks) or “physics code” (e.g. the implementation of a tracking algorithm) versus “analysis code” (often ntuple analysis)? How does “offline” differ from “online”? How do we deal with different timescales for “life” of such software? (M. Paterno)

Can documentation and training be included as a component that can be enforced by a design, review, and/or reward system, to lower the barrier for those that are not as familiar with the specific software? (H. Schreiner)

Can the software stack be modularized (libraries instead of monolithic tools, e.g. LLVM’s strategy) so that multiple groups can pick and choose what to use? The institute could promote and reward efforts in this direction. (R. Kalescky)

Can the institute promote standards (data, libraries, etc.)? Either specific to HEP or otherwise. (R. Kalescky)