Sustainable Software for HEP Workshop

Indico link to meeting:
https://indico.cern.ch/event/930127/

Some stats:
# of registrants: 89
Peak # (T ~ initial start time of workshop) was ~78
Approximately 30 people stayed for the breakouts

Talks

IRIS-HEP Blueprint process

Q: Is there a connection between IRIS-HEP and IRIS (https://www.iris.ac.uk/)? [there's no connection] Or is it just a name clash? [yes]

- There's crossover in terms of experiments (LHCb, ATLAS, CMS) and scope (sharing knowledge around compute) so perhaps scope for collaboration if not already?
- IRIS was mainly infrastructure for scientific computing (they did manage to fund a little Rucio development). UK has funded ECHEP and ExCALIBUR projects now, with SWIFT-HEP proposal just submitted (which would be more like an IRIS-HEP in the UK).

Introduction to software sustainability 15m

Speaker: Daniel S. Katz (University of Illinois)

Experiment experiences 15m

Speaker: Edward Moyse (University of Massachusetts (US))

How does ATLAS find enough people trained in C++ to get reasonable shifters?

- Comment that most undergraduates these days come with Python, not C++ (Graeme) and modern C++ training seems hard to find in the wild (cf. Python)

What's the average length of time that people spend in junior software roles on ATLAS?

Is the issue around funding incentives an issue with funding agencies or science leads putting in the funding proposals? Or some other group? Surely funders only want to see good science being done?

Experiment experiences II 15m

Speaker: Danilo Piparo (CERN)

I really like the idea of explicitly thinking about making the analysis software sustainable - which is normally outside the thoughts of reconstruct writers, etc. I wonder if both experiments shouldn't really think that all the way through to see if there is any other low hanging fruit.

I don't see how LDMX's approach is different , don't they need software process to create the containers?

- DLange : I also decided I didn't understand the comment.

HSF: HEP Software Foundation 15m

Speaker: Graeme A Stewart (CERN)

DL : Contrary to Graeme's comment, CMS does not provide a cmake infrastructure

- Oops, yes, sorry - I forgot that! The larger point was that within an experiment there is an infrastructure that people can use, but outside projects need to set this up and there is a best practice for doing that.

PGartung: Not that we didn't try.

Community Software Successes & Failures 10m

Speaker: Elizabeth Sexton-Kennedy (Fermi National Accelerator Lab. (US))

Most things start as single institution and grow as they become interesting to a community

Institutional Support: what is the role of universities in this ecosystem?

- HTCondor is an example of this. Yes as Miron commented and demonstrated, it is possible for a single PI to carry a program of sustained support and development over decades but it takes great commitment from that PI.

There are multiple reasons that software might reach end-of-life. Not all are due to lack of institutional support. Some software products just get replaced by something better technically or more sustainable. True, however in the case of Geant and ROOT, it was the same institution that lead the charge in developing the replacement.

You made the point that language is important - does that mean everything will eventually go away? It was FORTRAN and now it is C++ and perhaps it will be something else in the future. Yes, all things will eventually go away if something better comes along to replace it, or the language becomes unavailable (Java from Oracle) and the maintainer decides that language

migration (Py2->Py3) isn't worth it. That is why lifecycle planning is important and if your VO is going to last longer then the project you are considering depending on, you have to ask what your exit strategy is. Fortunately C++ is a very actively developed language with additions to the standard every 3 years and a big commitment to backwards compatibility.

Re. the role of Universities, there are successful groups and careers around detector development (which has a much higher buy-in cost than software); I think we want to make the same arguments for software being key and something Universities should invest faculty in. (Making the high-level advocacy for software is necessary there to give the context.) [Graeme]

Software Sustainability Institute (SSI) 10m

Speaker: Neil Chue Hong (EPCC, University of Edinburgh)

Astropy 10m

Speaker: Adrian Price-Whelan (Flatiron Institute)

Comment: Lots that HEP can learn from (and envy, franky) with the AstroPy project.
Q: What is the HEP software project most closely analogous to AstroPy? Scikit-HEP?

- (Ben K): I think it's both the PyHEP and scikit-hep projects: scikit-hep for the code aspects, PyHEP for the community

The Carpentries 10m

Speaker: Erin Becker (The Carpentries)

HSF training 10m

Speaker: Samuel Ross Meehan (CERN)

Research Software Engineers 10m

Speaker: Ian Cosden (Princeton University)

Software maintenance and capacity building in HEP 10m

Speaker: Dr Ketevi Adikle Assamagan (Brookhaven National Laboratory (US))

- Software maintenance should start with good software documentation. Well-documented software provides a good example for the people that inherit such a software. They will not rewrite it, which often happens when we do understand or like a piece of software passed on to us. And rewriting is a waste

of time and effort; Good-documentation also serve as a excellent education platform when we pass on the software to the younger generation;
- We should pay attention to increase diversity and inclusion in the computing and software fields. Please join us in the Snowmass 2021 Community Engagement and Computational Frontiers where we aim to discuss these issues and make recommendations to HEPAP.

Various resources mentioned in chat:

since we're also talking about productivity (software engineering, etc.), folks might be interested in another workshop going on at the same time, with materials (recordings and white papers) online - https://collegeville.github.io/CW20/

“programming” is #1 “skill acquired at CERN that are deemed important for working outside the field” https://cerncourier.com/a/assessing-cerns-impact-on-careers/

This is a great short tutorial on how to light yourself online (aimed at theatre practitioners, but great for us too!): https://www.nationaltheatrescotland.com/latest/lights-for-home-filming

Breakouts Reporting - Recommended Actions

Group 1:

1. Expand the use CI or other testing - to as many platforms as feasible and reasonable
2. Figure out how to reward documentation
    + ++3. Figure out how to develop career paths for the people who work in this

Group 2:

1. Create a policy at experiment/community level that all software created as part of experiments must be written up in a software paper (something that gets a DOI) and that this must be cited by experiments
++++++++ 2. Promote recognition and financial support/career paths: Have HEP RSEs on the funding panels / funder advisory committees / hiring committees
3. Have software professionals and contribute their effort from your organization (pledge agreements)

Group 3:

+ +++++++++1. Repurpose nominal funding from in-person training to pay content developers
+ + 2. Repurpose nominal funding from in-person training to pay participants

++ 3. Incentivize training participant -> instructor transitions

Group 4:

See the three “Things” and their executive summary in:
https://docs.google.com/document/d/11RsIZo1Q2ltpnHuInQm07jPtUqEEvvTxU7DVxEinCqs/edit

+++ 1. C++ training for HEP (that is sustainable and scalable). Modern C++. This is now a specialism. Make this equal with detector development specialism.
+ +2. Use of containers (to support analysis; well developed in ATLAS). Sustainable and integrated with repo + CI. Develop best practices
+ +++3. Incentives and professional development, some certification? RSEs training incoming HEP students. Mentoring

Group 5:

+ +++ 1. Organize a workshop to explore and define common software and services, including the following. How does the community create standards that are sufficiently flexible to be widely accepted, while still concentrating our combined efforts?

a) libraries
b) data and computational services
c) gateways (VREs) and analysis services

+ + 2. Define 2 or 3 common development environments as community accepted standards.

a. make these available via application container technology (e.g. Docker)
b. include project templates for testing, documentation and more
c. proposed name (tongue in cheek): the HEP software development kit (HPSDK)

3. ran out of time