Software Training in High Energy Physics

Michel H. Villanueva [1], **Sudhir Malik** [2], **Meirin Oan Evans**[3]

[1]Deutsches Elektronen-Synchrotron (DESY), Notkestraße 85, 22607 Hamburg, Germany
[2]Physics Department, University of Puerto Rico Mayagüez, PR 00682, USA
[3]Department of Physics and Astronomy, University of Sussex, Brighton BN1 9QH, UK

E-mail: sudhir.malik@upr.edu

Abstract. Among the upgrades in current high energy physics (HEP) experiments and the new facilities coming online, solving software challenges has become integral to the success of the collaborations. The demand for human resources skilled in both HEP and software domains is increasing. With a highly distributed environment in human resources, the sustainability of the HEP ecosystem requires a continuous effort in the equipment of physicists with the required abilities in software development. In this paper, the collective software training program in HEP and its activities led by the HEP Software Foundation (HSF) and the Institute for Research and Innovation in Software in HEP (IRIS-HEP) are presented. Experiment-agnostic, open, and accessible modules for training have been developed, focusing on common software material with ranges from core software skills needed by everyone to advanced training required to produce high-quality sustainable software. A basic software curriculum was built, and an introductory software training event has been prepared to serve HEP entrants. This program serves individuals with transferable skills that are becoming increasingly important to careers in the realm of software and computing, whether inside or outside HEP.

1. Introduction

Software skills are an integral part of the toolkit of any successful HEP experimentalist. Maximizing the science from the hardware investments in current and future HEP projects relies critically on them. These skills are also transferable in the case of non-HEP career evolution of people trained in HEP. Although they play a critical role in many research fields, most users learn software skills only after joining a research program. Universities do not uniformly provide such training to students prior to the beginning of their Ph.D. research. Many domain-specific aspects further complicate the learning process. Embarking on a HEP-specific path presents its own experiment-specific software environment difficulties. The HEP community must play a role in bringing together a focus on these efforts, incursioning into sustainability and scalability, and initiating learning early in the process of preparing a software-equipped future particle physicist. There is no one size that fits all when imparting software training. A possible solution in HEP is to exploit our large-scale community structure and organize training within our research domains. Efforts like HEP Software Foundation (HSF) [1], IRIS-HEP [2], FIRST-HEP [3] and SIDIS [4] have taken strong and effective steps in this direction and impart training [5] to those around the field.

2. Training for Software in HEP

Trained software developers are critical resources for the success of current and future HEP experiments. It is therefore essential that the community as a whole design and develop tools, methods, and resources to advance software skills, which in turn will lead to solutions for software challenges. Developing a workforce early is required to address the emerging needs and unresolved bottlenecks in software and computing development. Undergrads, Master and PhD students may leave the field of HEP after completing their degrees. Imparting software skills while they pursue their academic goals can help jump-start contribution to the experiment, as well as prepare for another STEM degree or a career outside or inside academia. Scientists and physicists who choose to stay in the field always need to update their skills with new tools that disrupt the field, like Machine Learning. As shown in Figure 1 the HSF/IRIS-HEP vision of the evolution in training spans from early graduate students to mentors and has extended this fabric to undergraduate and master student levels. Furthermore, the training outreach activities are reaching STEM aspirants in pre-college students and their teachers.

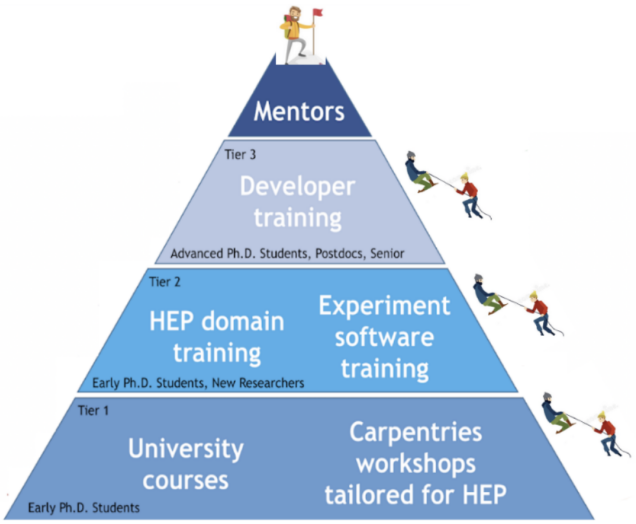

Figure 1. Evolution of HEP Education and Training

HEP software training is a cross-experiment, multi-domain activity and poses its own challenges. Each experiment has a big chunk of its own specific software. However, many HEP tools like ROOT and GEANT, coding languages like Python and C++, Operating Systems like Linux, version control systems like *Github*, and algorithms like Neural Networks and Boosted Decision Trees are common across the experiments. Some of the common HEP tools are common to the entire software community across all STEM and Computing disciplines. Our training and outreach events can be found here [6, 7]. A rough estimate shows that we have about ∼10K people in our community to train every year, ranging from Undergrad to Scientists, with many of them holding non-permanent positions and staying in the collaborations for significantly less time than the life span of the experiments. This implies that keeping a critical mass of developers requires that training activities scale and sustain with time, so that the most updated technologies and practices quickly reach all. What adds to this challenge is that software training is a completely voluntary activity for mentors and participants. This requires the continuous motivation of new participants or experienced users to volunteer as mentors.

3. Community Building

Community building is central to HEP-wide software training. It is essential to build and improve material, incorporate new ideas, make connections and acquire new mentors for sustainability and scalability. The HSF/IRIS-HEP training community of mentors is growing [8, 9] and is voluntary-based, reflecting the vision that participants become mentors at different levels in the pyramid and feedback into the system. It has also expanded the mentor community to related fields like Nuclear Physics and Computing. However, keeping the community growing and engaged needs constant nurturing, coordination, motivation, and involvement of several key and senior people, especially, from labs and institutions. The mentors should also be able to see the value in continuing to remain a part of the community, at least for some reasonable period of time. The mentors' profiles (see a snapshot in Figure 2) are advertised on our web page, showing also links to their further professional details.

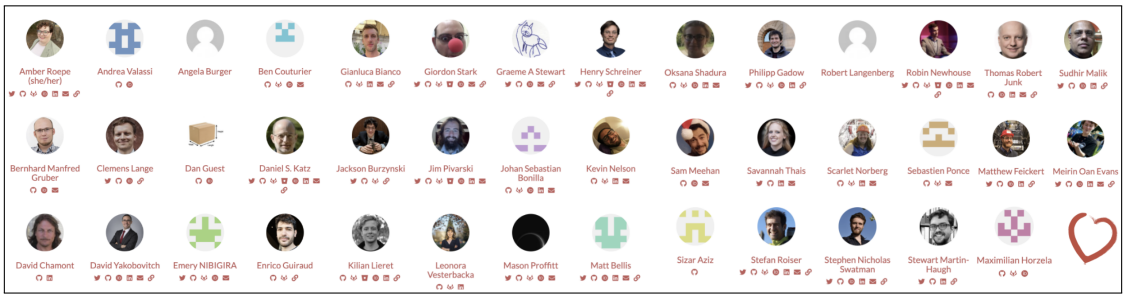

Figure 2. Community of Mentors

4. Training Content

Most HEP experiments have similar basic prerequisites that are actually common with other related areas like Nuclear Physics or Computing. We are building a training curriculum that consists of independent modules. This is necessitated by a view that students should be able to prioritize certain skills before others, as in many cases like HEP Physics Analysis it is expected to jump-start directly towards scientific results, with minimal time given for acquiring software knowledge or best practices. The current spectrum of modules consists of a basic skill set (*Unix, Shell, Python, Git*) that serves newcomers, software engineering topics (Continuous Integration) with some specific to HEP data analysis and techniques, to advance topics (Parallel Programming and Machine Learning). All Modules are open-source, which is in line with our view of Open Science. Figure 3 shows a snapshot of our training curriculum [10].

We have organized training events [11] in the last 3-4 years and trained about 1500 people. Due to COVID-19, virtual training events were held in 2020. Some advantages and disadvantages of in-person and virtual training, based on our training experience, are described in [5]. Software Carpentry events populated with a basic skill set and some HEP-specific training topics (like *PyROOT, Uproot*) are being organized 2-3 times per year, supported by our community along with *The Carpentries* [12], with which we currently have a membership, supported by IRIS-HEP. As our community and training topics mature, we aim to scale up the frequency of training on different topics.

5. Impact of Training

Assessing the impact of training is essential to its continued effectiveness, success and future. Every training format involves a pre-training and post-training survey. While the pre-training survey includes demographic and "How much do you already know" questions, the post-training

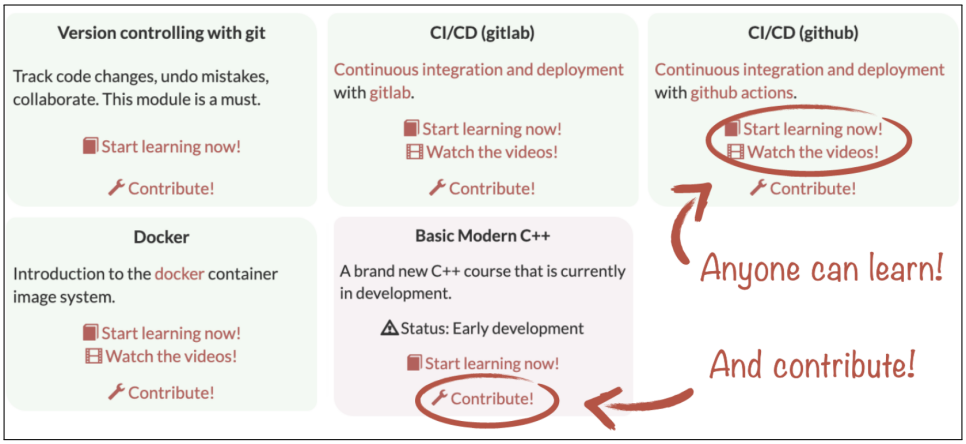

Figure 3. HEP Software Curriculum

is focused on what has been learned and compares the outcome of the skills before and after the training. One such example is shown in Figure 4, where a comparison is made of knowledge in *Uproot* and *Git* modules before and after the training. Note the difference in the statistics of people who took the survey before and after the training. Getting everyone to take surveys, especially after the training, has been a challenge ad we are trying to address that. However, it is clear that the training is transforming the skill set of participants. The number of people who know more confidently about the module moves from left (before the training) to the right (after the training) for both the example modules shown.

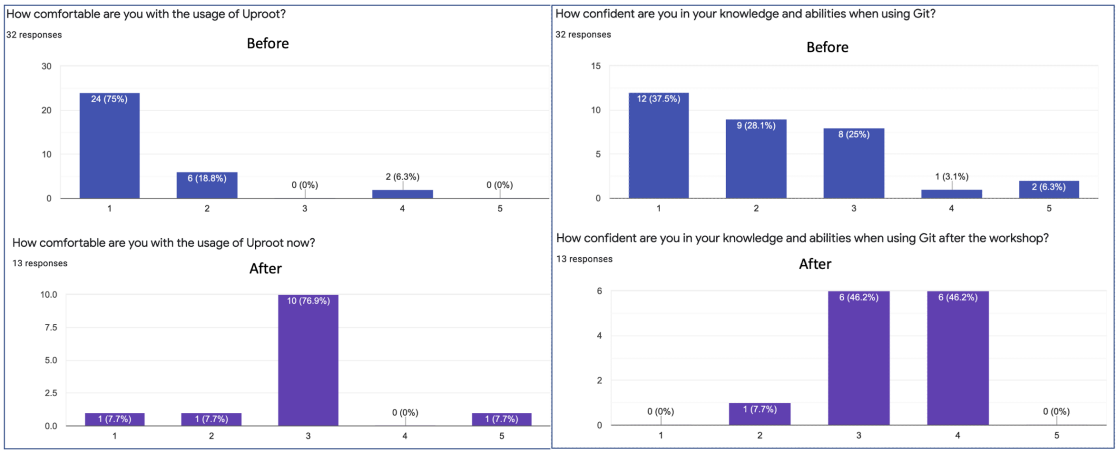

Figure 4. Survey Feedback for *Uproot* and *Git* modules

6. The Software Training Challenge

One of the key things to succeed in software training is to have an up-to-date website [13]. As a training community, regular brainstorming sessions are organised every few months to have experts re-visit our training material and surrounding pedagogy. In addition, there are weekly meetings to discuss the progress in the planning of events and development of materials [14]. A rough estimate shows a target HEP community of around ten thousand Postdocs, PhD students and Undergrads that work on projects in any given year. There are similar people from related

fields like Nuclear Physics, Astrophysics and Computer Science whom we constantly try to reach out to. There are also scientists and faculties who may want to learn or update their software skills. Our goal is that every HEP graduate student should attend the basic training to the very least. This translates into an increase in the frequency of our events and size in the community of individuals - facilitators, participants, instructors, experts and hosts. In the recent past we have succeeded in expanding collaboration with related communities like Nuclear Physics and Neutrino. We want to explore ways to incentive new trainers to continually contribute. We are especially committed [15] to Diversity and Inclusion where everyone feels welcome to participate. We also host Outreach events to promote early software awareness and skill development among high school students. Some of our events can be accessed via [16].

Most of our training events have been in the COVID-19 era. As we come out of it, we aim to look into more creative ways to scale and sustain. Evaluating curriculum receptivity, building a course around a basic curriculum for HEP beginners, giving course credits and certificates of participation as an incentive, are some of these ways. In the long term, having a financial model that builds regional and local training capacity is essential to sustain and scale. Another key part of this model is to establish equity, diversity, inclusion, and accessibility. This participation will span across HEP communities, under-resourced institutions and communities in different geographical regions. Opportunities to grow professionally and have career paths for the core team and volunteers are essential as we plan for the future.

7. Conclusions

We discussed the key feature of the HSF/IRIS-HEP software training program that aims at ensuring sustainability of software in HEP for years to come. Our training material allows open-source access. This process enables continual feedback to improve the curriculum. The program promotes a culture within HEP of valuing software skills and teaching of those skills to others. A growing community with a special focus on inclusion and diversity in science is essential to sustain and scale the program.

We thank NSF grants OAC-1836650, OAC-1829707 and OAC-1829729 for support of the training program.

References

[1] High Energy Physics Software Foundation `https://hepsoftwarefoundation.org/`
[2] Institute for Research and Innovation in Software for High Energy Physics (IRIS-HEP) `https://iris-hep.org/`
[3] Framework for Integrated Research Software Training in High Energy Physics `https://first-hep.org`
[4] Software Institute for Data-Intensive Sciences `https://sidis.web.cern.ch/`
[5] Malik S, Meehan S and Lieret K *et al* 2021 *Springer: Computing and Software for Big Science* **5** 7
[6] HEP Software Training Events `https://hepsoftwarefoundation.org/Schools/events.htmll`
[7] Training, Education and Outreach `https://iris-hep.org/ssc.html`
[8] The HSF Training Community `https://hepsoftwarefoundation.org/training/community.htm`
[9] Lieret K 2021 Community building `https://indico.cern.ch/event/941278/contributions/4084356/` HSF WLCG Virtual Workshop
[10] HSF Software Training Center `https://hepsoftwarefoundation.org/training/curriculum.html`
[11] HSF training events `https://hepsoftwarefoundation.org/Schools/events.html`
[12] The Carpentries `https://carpentries.org/`
[13] HSF Training `https://hepsoftwarefoundation.org/workinggroups/training.html`
[14] HSF Training Meetings `https://indico.cern.ch/category/10294/`
[15] Host, Request and Organize a Training Event `https://hepsoftwarefoundation.org/training/howto-event.html#do-i-need-to-consider-aspects-of-accessibility-and-diversityinclusionequity`
[16] HSF/IRIS-HEP Outreach Events `https://indico.cern.ch/category/11386/`