Evolution of HEP Processing Frameworks

Christopher D Jones and Kyle Knoepfel
Fermi National Accelerator Laboratory

Paolo Calafiura and Charles Leggett
Lawrence Berkeley National Laboratory

Introduction

HEP data-processing software must support the disparate physics needs of many experiments. For both collider and neutrino environments, HEP experiments typically use data-processing frameworks to manage the computational complexities of the following processing categories:

- **Triggering**, which uses digitized responses directly from the detector to quickly decide if the information is worth storing for later processing,
- **Calibration**, which is performed to provide proper understanding of the detector responses,
- **Reconstruction** of the detector responses using the calibrations to provide insight into the underlying physical quantities,
- **Simulation**, which includes the generation of an underlying physical interaction, its detector response, and the consequent reconstruction to mimic physics data, and
- **Analysis**, which uses the reconstructed information to obtain a physical result.

To implement all the categories above can require an experiment to develop and maintain several millions of lines of software involving tens of thousands of different interrelated algorithms. Such a challenge is made tractable by breaking apart a large-scale system into separable components, which can evolve independently of each other. This flexibility also means components can be specialized to deal with the processing needs of a (e.g.) computing site whose storage interface differs from that of other sites.

By stipulating a set of rules and APIs, each framework imposes a uniformity in how components are written, initialized, and executed, as well as how they communicate with each other. This uniformity enables framework users to (a) apply one programming approach to a myriad of data-processing tasks, and (b) more easily grasp how many different processing configurations function. A framework can thus assemble a workflow simply by consulting a user-supplied configuration that specifies which components to use, how to initialize them, and how to schedule them to process data.

Given their historic success, frameworks will continue to be critical software systems that enable HEP experiments to meet their computing needs. As in the past, framework evolution will be driven by such needs and by both software and hardware changes in the computing landscape.

Computing landscape

For decades, it was sufficient for a framework process to execute on only one CPU core. Computing clusters and large-scale batch systems were deployed and have achieved impressive data-processing throughput using only commodity CPU hardware. However, as CPU clock rates have plateaued, computing nodes now include multiple integrated CPU cores, performing computations in parallel to keep up with Dennard scaling.

HEP data-processing frameworks have transitioned (or are transitioning) to being able to exploit multiple cores on a computing node. This can be achieved by simply executing multiple processes on the same node, or by using multi-threading, where a single process accesses multiple cores through threads. Whereas multiprocess programs are simpler to handle due to distinct memory spaces, multi-threaded programs have the advantage of reducing memory usage at the cost of adopting a more complicated programming model.

Further evolution, both in frameworks and workflow management, will be needed to exploit computing resources from non-x86 CPUs as well as non-CPU hardware, for example GPUs or FPGAs. In addition to hardware changes, HEP frameworks will need to evolve to effectively utilize computing at non-traditional HEP computing sites such as commercial Cloud computing or High-Performance Computing centers, which support preemptible, distributed, and heterogeneous applications. The DOE's leadership computing facilities, in particular, are constructing extremely parallel systems, which favor non-CPU computation to achieve their stated performance within an affordable energy budget. Significant effort within the HEP community will be required to develop and deploy framework solutions that meet these computing challenges.

Conclusion

We invite the HEP community to acknowledge the importance of data-processing frameworks, their successes, and the computing challenges such frameworks face. Frameworks have weathered computing revolutions in the past; they will do so again with support from the HEP community.

This LOI serves as a preface to a white paper that will expand on the motivation for data-processing frameworks, how the changing computing landscape will affect those frameworks, and what endeavors will be necessary to meet upcoming framework challenges.