Attachment A

Statement of Work (SOW)

HEP Reconstruction and Analysis Software for Exascale-era Computing

Dated March 11, 2016

Overview

The primary objective of this project is to produce a prototype software system suitable for moving HEP experiment event data through multiple processing stages in an exascale-class computing facility. In particular, work will concentrate on demonstration of two critical components of a complete event-processing system: (a) high-performance I/O to a parallel filesystem, and (b) communication of event data through high-performance node interconnects (rather than through the filesystem), between processes of one application running across hundreds or thousands of nodes - *a distributed program*.

Fermilab requires professional services to advise, recommend, and provide the following:

1. Provide a plan to design a file schema using HDF5, suitable for the storage of experimental HEP data.

Experimental HEP data are typically richly structured (many nested types, not dominantly fundamental types), and are dominantly variable-size data structures (such as C++ vectors). Part of the project's research work is to understand, using real-world use cases from some of the experiments we currently support, how to use HDF5 most effectively to store the data. We will together select three specific use cases for this development.

2. Determination of what, if any modifications of the organization of HEP-specific data structures are needed.

The current software design practices in experimental HEP are heavily influenced by the technology (ROOT) that dominates the field. You shall investigate and recommend changes that would be appropriate in the design of our data structures to allow us to make better use of HDF5 and the high-performance parallel storage systems that it supports. This task explicitly does not involve modification of HEP algorithms to make use of these new data products.

An important part of experimental HEP software development is the ability of physicists to invent new persistable data types, without the physicist needing deep understanding of the I/O system, without having the physicist writing any code to support I/O for the new type,

and without having to extend the event-processing software framework to know about the new type. Based on what is learned from the other work on this project, you shall write a whitepaper discussing how a system (for example, a data definition language and code generator) might be devised to allow both the C++ classes for data products and the I/O code to support those classes to be automatically generated.