Contribution ID: **110** Type: **Oral**

Optimizing ROOT's Performance Using C++ Modules

Monday 10 October 2016 14:30 (15 minutes)

ROOT version 6 comes with a C++ compliant interpreter cling. Cling needs to know everything about the code in libraries to be able to interact with them.
This translates into increased memory usage with respect to previous versions of ROOT.

During the runtime automatic library loading process, ROOT6 re-parses a set of header files, which describe the library; and enters "recursive" parsing. The former has a noticeable effect on CPU and memory performance, whereas the latter is fragile and can introduce correctness issues. An elegant solution to the shortcoming is to feed the necessary information only when required and in a non-recursive way.

The LLVM community has started working on a powerful tool for reducing build times and peak memory usage of the clang compiler called "C++ Modules". The feature matured and it is on its way to the C++ standard. C++ Modules are a flexible concept, which can be employed to match CMS and other experiments' requirement for ROOT: to optimize both runtime memory usage and performance.

The implementation of the missing concepts in cling and its underlying LLVM libraries and adopting the changes in ROOT is a complex endeavor. I describe the scope of the work and I present a few techniques used to lower ROOT's runtime memory footprint. I discuss the status of the C++ Modules in the context of ROOT and show some preliminary performance results.

Primary Keyword (Mandatory)

Analysis tools and techniques

Secondary Keyword (Optional)

Tertiary Keyword (Optional)

Author: VASILEV, Vasil Georgiev (Fermi National Accelerator Lab. (US))

Co-author: CANAL, Philippe (Fermi National Accelerator Lab. (US))

Presenters: CANAL, Philippe (Fermi National Accelerator Lab. (US)); VASILEV, Vasil Georgiev (Fermi National Accelerator Lab. (US))

Session Classification: Track 5: Software Development

Track Classification: Track 5: Software Development