Contribution ID: **11** Type: **not specified**

From ROOT to BOOT, from BOOT to ROOT

Monday, 26 March 2007 10:40 (45 minutes)

The ROOT system is currently undergoing an important evolution. The aim is to provide a very small kernel, called BOOT, easy to install on all systems and very stable. Using the BOOT executable module, all the other ROOT libraries (and later on users and experiments code) will be automatically imported via the web from central source repositories. This should greatly facilitate the installation of the system on a new computer and provide automatic updates too. The following 3 phases of the project have been defined:

-Phase 1: Restructuring of the CORE ROOT framework such that only a small subset will be required to start a running ROOT application (including user libraries). This phase is well advanced and will be presented in details. Thanks to this first phase, the virtual address space required by the interactive version has been reduced by a factor 2.

-Phase 2: Instead of generating (via rootcint) the code for the dictionaries that can represent a substantial fraction of the total code to be compiled and linked, the dictionary information will be written to small ROOT files stored together with the shared libraries. In particular the CINT interface stubs to call compiled code will be replaced by generic inline calls. The dictionaries will be loaded from the ROOT files only when really necessary. This will also minimize the amount of information to be stored in memory and it will decrease substantially the size of the shared libraries, in particular in case of templated classes.

-Phase 3: Autoloading of the code on demand from a central source repository and online compilation of the code with local storage(cache) of the compiled code. This will facilitate considerably the installation of new versions. This feature will become important in the future when the percentage of code really used in a running application will be a very small subset of the total code in the source repository (or currently shared libraries). I will discuss the implications of this Phase when considering the C++ language evolution or interface with other languages.

Presenter: BRUN, Rene (CERN/SFT)