The Fermilab Physics Class Library

Fermilab Physics Class Library Task Force (FPCLTF):
M. Fischler [a] D. Adams [b] W. Brown [a] I. Gaines [a]
R. D. Kennedy [a] J. Marraffino [a] L. Michelotti [a] M. Paterno [c]
E. Sexton-Kennedy [a] J. Yoh [a]

[a] *Fermi National Accelerator Laboratory, Batavia, Illinois*

[b] *Rice University, Houston, Texas*

[c] *University of Rochester, Rochester, New York*

The Fermilab Physics Class Library Task Force has been formed to supply classes and utilities, primarily in support of efforts by CDF and DØ toward using C++. A collection of libraries and tools will be assembled via development by the task force, collaboration with other HEP developers, and acquisition of existing modules. The main emphasis is on a kit of resources which physics coders can incorporate into their programs, with confidence in robustness and correct behavior. The task force is drawn from CDF, DØ and the FNAL Computing and Beams Divisions. Modules—containers, linear algebra, histograms, etc.—have been assigned priority, based on immediate Run II coding activity, and will be available at times ranging from now to late May.

1 Intentions, Structure and Standards of the FPCLTF

The Tevatron Run II physics program of the two major collider experiments at Fermilab (which discovered the top quark in 1995), is scheduled to begin in 1999. For several years this will be among the leading-edge experiments in the world. CDF and DØ have each (independently) decided to base a major portion of their respective Run II software on C++. These decisions were motivated by a desire for greater usability and long-term maintenance. For instance, a Run I event reconstruction program was somewhat "brittle," in that introducing (and testing) minor improvements became an ordeal. The C++/object-oriented approach, which might alleviate such problems, is considered important in designing programs for Run II and beyond.

Preprint submitted to Elsevier Preprint *15 May 1997*

To support this move, Fermilab must develop a resource base of relevant, easy-to-use, well-designed library modules to serve key HEP needs. There are problems common to CDF, DØ, and others at the lab. Providing common solutions will leverage the solving effort, and also develop a strong local pool of C++ design expertise, which software teams from the experiments can draw upon. These considerations have led to the formation of the Fermilab Physics Class Library Task Force (FPCLTF), a joint CDF, DØ, and FNAL Computing and Beams Divisions group.

Under the guidance and input from CDF/D0/Computing-Division managers and physicists, and in concert with similar HEP efforts such as BABAR[1] and LHC++[3], we intend to provide these libraries through acquisition (commercial or freeware—though likely with modifications or adjustments) or creation (often in collaboration). Some local development must take place—the effort to ensure that modules are available when needed, and to create high-quality products, will generate the experience essential to providing a local pool of expertise. Information about the class library is disseminated from the FPCLTF web page at `www.fnal.gov/docs/working-groups/fpcltf`.

A uniform structure of distribution, headers, libraries, documentation, coding styles, and testing for the modules in the FPCL is described in a Scope and Standards document[2]. "Borrowed" products may require some massaging into this structure. For example, the task force may create distribution packages and port documentation into various formats. User documentation in the form of postscript and *indexed* html will be required. The format and information appropriate for each form is specified in the standards. Coding style and code management standards are provided to facilitate maintenance and future enhancement of modules. A collection of user interface rules, such as prefering public methods to Fortran-looking global functions—`v1.cross(v2)` rather than `cross(v1,v2)`—is emerging. However, where commercial products and established HEP libraries come with acceptable interface, documentation and structure, they may be incorporated into FPCL without filling in every "missing" piece.

Quality, both in correctness and robustness, will be enforced by imposing design and testing standards. Basic tests define the functionality of a module, but a suite of stress tests to expose any defects must be created and reviewed for range of coverage. These stress tests are applied *after* delivery of supposedly correct implementations. Thus this testing both exposes defects and provides a measure of confidence in the quality level of the module. In addition, module releases are checked on a set of reference platform environments (currently IRIX, AIX, DEC Alpha, Solaris, Windows NT, and Linux), and "blind tested" by a physicist without expert assistance.

2 Contents and Priorities for the Class Library

In January, the base "customers" of FPCL—software developers primarily on the CDF and DØ experiments—met to determine what the task force should do first and how the contents of FPCL should evolve. After discussion of what HEP class resources were being developed elsewhere (including a presentation on CLHEP and LHC++ activities from the BaBar perspective[1]), a list of modules desirable in a class library for Fermilab users was formed. This provided a framework for setting priorities, based on presentations by CDF and DØ software management: Software development groups are beginning to design code, and need to know what library resources they can count on. Immediate needs of these groups dictate the modules targetted for delivery by April/May.

This first phase of activity, decribed in more detail below, includes: **Container classes** and STL; **C++ stdlib** implementations; **Linear Algebra** efficient for small and large matrices; **Space and Lorentz Vectors** and transformations; **Histograms**; and **Fixed-length data types**. To design these, FPCLTF also needs to develop technical policies concerning error logging and hooks for persistence mechanisms, and a tool to produce user documentation in the FPCL formats.

Slated for development on a 6–9 month time scale are: a **Random Number** framework which can accomodate a variety of generators and distribution functions; **Concrete Data Types** including numbers with associated error-correlation matrices; **Minimization** in a C++ version based on MINUIT; **Code Patterns** (e.g. reference-counted objects); **Particle Data Book** data, including parton distribution and structure functions; **Documentation Tools** packaging some valuable HEP-developed tool sets; and **Timing Routines** and specialized assertions.

Other anticipated modules include: visualization and event display, geometry, error handling, accelerator simulation, a user interface package, interactive graphic input, shared memory tools, specialized fitting, and simulation/tracking tools.

Work is proceeding on several urgently needed "Phase I" modules:
FPCL must provide suitable container classes and stdlib. STL must be included, but while it is well-designed, the learning curve for safe usage by non-expert physicists is steep—additional functionality is desired. A leading prospect for supplying a unified structure that may satisfy container class needs and augment stdlib is `Tools.h++` (by Rogue Wave Software). We are providing that library to developers to examine usage patterns; barring serious deficiencies we will follow the HEP trend toward `Tools.h++` for both contain-

ers and stdlib coverage. This will be augmented by provisions for fixed-size types, already available. A package should be in place for the Fermilab community by May 1.

Linear Algebra packages available are generally inadequate for HEP needs. As pointed out at the workshop, commercial packages tend to have unacceptable overheads for small-matrix cases important to HEP. Home-brew HEP packages often can't take advantage of fixed-size matrices, lack some matrix operations and specialized forms, and/or are not robust and numerically stable in instances when they should be. The underlying code and algorithms of these packages, however, can be used to "jump start" our class development. A design strategy review in early March will lead to implementations covering all the immediate needs, delivered in May.

Several HEP efforts have produced 3-vector and 4-vector classes. Some of these represented early strivings to attain expertise in C++ and are flawed in various ways. The design of Vector, Rotation and Lorentz Transformation classes is dominated by deciding what the proper interface and set of supported concepts are. An e-mail design review of the concepts, interface, and header files took place in February; the results, headers, and a feature guide can be accessed off the FPCL web page. The complete package should be released for Run II developers in April.

There exist several Histogram packages tailored for HEP work. The effort of FPCLTF in this area is to define what our users want in a histogram interface, and try to provide that interface, with hooks so that the underlying work can be done by HBOOK, Histoscope, or whatever package is favored. The difficult issues will be how to cope with desired functions which may not be supported by each package.

We thank CDF and DØ software developers for key input and guidance. Researchers external to FNAL, including Bob Jacobsen and Joe Boudreau, were influential in determining technical directions. Finally, M. Mengel, S. Snyder, and F. DeJongh have provided additional technical design guidance.

References

[1] The BaBar web page, which includes links to B. Jacobsen's presentaton, may be found at `www.slac.stanford.edu/BFROOT/doc/www/Computing.html`.

[2] The FPCLTF Scope and Standards Document, a FNAL Technical Memo, accessible via links from the FPCL web page.

[3] The LHC++ web page is at `wwwcn1.cern.ch/asd/lhc++`. This page has a link to CLHEP, which has been incorporated into LHC++.