The Marriage of Trybos and Root

Stage 1 (and Little Beyond) for the CDF Run 2 Event I/O System

CDF Week: Dataflow from L3 to Physics
19-May-1998
Rob Kennedy

- **Goals for Stage 1**
- **Constraints, Changes, etc.**
- **Deployment and Other Tasks**
- **Stage 1 Staff and Status**
- **Looking Ahead**

Trybos is a 100% original C++ software package capable of reading/writing Ybos files, manipulating Ybos banks, etc. Trybos has always been intended to serve as an interface to events in data formats other than Ybos. Stage 1 of the Event I/O portion of Data Handling will involve extending Trybos to support a simple sequential Root format.

Goals for Stage 1

➪ By 01-Jun-1998, Trybos will be able to read/write Ybos and/or Root format *sequential* datafiles. This is not a firm "deadline". The first release is not going to be optimally efficient for the Root format.

➪ Trybos will only use that part of the Root product which is required for event I/O, roughly 2 to 4 of Root's 28 libraries.

➪ User code in modules within AC++/Framework will not have to change in order to read/write Root format files.

➪ Existing Banks classes will continue to be used to access data elements. Root object-browsing of Banks will not be supported.

➪ Stage 1.5: A restricted F77 API for Trybos will be developed afterwards which will allow F77 user code in AC++ modules to manipulate banks, access data in banks, etc. *This F77 API heavily constrains the model of a Bank within Trybos.*

Constraints, Changes, etc.

- F77 API: the data section of each BLOCAT'd bank must be available in a single contiguous integer array. Data may have to be copied to/from an IW emulation which is visible to C++ code.

- AC++ job scripts (like Run I .uic files) may have to specify the format of files used for I/O. This greatly simplifies dealing with remote "net" files, for instance. The data format can change from file to file within an AC++ job, but it must be the same within a file.

- Root does not now compile with KCC --strict, which we use in Cdfsoft2. Trybos will try to hide this from users as much as possible.

- Must support event mixing, multi-file input and output, etc.

- None or few user code changes: no well-defined bank header size.

- Some changes to AC++ I/O modules to switch on file formats....

Deployment and Other Tasks

- The complete Root product will be made available as an external package within Cdfsoft2. Only those portions of it used for Event I/O will be supported locally; All else will be supplied "as-is".

- The changes to the Trybos API should be insignicant enough that a separate CVS branch should *not* have to be set up. The revised Ybos format support will be released separately and before the Root format support to minimize disruption.

- A SoftRelTools arch_spec_Root.mk file may be needed to simplify building software that depends on the Root libraries.

- A compile-time switch will be defined to allow users lacking the Root package to continue to use Trybos with the Ybos format.

- As work progresses, we should pursue closer technical and managerial contacts with the Root development team....

Stage 1 Staff and Status

➪ Rob Kennedy is dedicated ~100% for May 1998 to work on this, ~50%? in June 1998 (flexible).

➪ Pasha Murat is consulting on Root concepts (event model) and working on Root product deployment.

➪ The design for Trybos w/Root is essentially complete. Some redesign of the Ybos format portion of Trybos was necessary to generalize the concepts of Generic_Bank, Record_Iter, and so on.

➪ Implementation has already begun. The Ybos portion is planned for release on or about 22-May-1998, Root portion by 01-June-1998.

➪ Root 2.00.05 has been built, UPS-productized, but not yet deployed for all of IRIX6, IRIX6-KCC, Linux2, and Linux2-KCC. This required "invention"; no other UPS products vary by C++ compiler(?). Also Root is maintained in CMZ, no GNUmakefiles.

Looking Ahead

➪ After initial deployment, and F77 API completed, we can consider a more optimal implementation of Trybos using Root Event I/O. For instance, can we drop the bank type description info from the Root format data stream?

➪ The event model chosen for Stage 1 is a strictly sequential model. We can later experiment in a more realistic fashion with composite events, multi-branch events, proxied branches, and other non-sequential data access patterns.

➪ The Banks package may be expanded to include Root-oriented versions of Bank classes to permit object-browsing, etc. These classes would have a data member for each field in the bank, rather than current Ybos array model of bank data (which works well with F77).

➪ Miscellaneous: Root little-endian file format (Level 3), improved diagnostics, low-level optimization of Trybos and Root, ...