Decision to move to CMT v1r18p20061003

April, 2007

We have decided to move to CMT v1r18p20061003. This newer version is available via the GLAST installer and all developers are encouraged to upgrade. Note that when using the installer, CMT will be copied to the extlib directory you specify.

Note: MRvcmt v0r24p2 is the recommended version to use in conjunction with this version of CMT. You should use MRvcmt v0r22p2 at least.

Note2: Navid and Joanne discovered that CMT now prepends the working directory to the CMTPATH. This is a change from the original behavior where CMTPATH has been purely user defined. This may be due to CMT's move towards CMTPROJECTPATH where CMTPATH is derived rather than user defined. However, the CMT documentation states that when CMTPROJECTPATH is left undefined, CMT should revert its original behavior. A note has been sent to the CMT mailing list to inquire.

Shared Library Support

The big push to get us to move to this new stable release of CMT was the need to support shared libraries on Windows. Jim Chiang was continuing to have trouble with shared libraries in SciTools. Initially, we intended to deploy Tracy's fix to CMT v1r16p20040701 - but Joanne found that v1r18p20061003 included the fix as well. We then set forth to test this new version on Linux, Windows and Mac and there were no ill effects. Then came time to demonstrate that shared libraries could really export their symbols! Joanne and Tracy worked with CalibData and Jim and Tracy checked out SciTools. Here's the summary:

When using CMT v1r18p20061003, there is no need to use dllimport and dllexport. From Tracy: "Clearly, the mods I had for cmt v1r16 only found part of the solution, the cmt folks have found the full solution so that the shared_library and linker_library patterns really do behave as advertised in exporting all symbols correctly." So we would do something like, which is now valid for both Linux and Windows:

```
# if defined(_CalibData_CalibModel_cxx)
#  define  _EXTERN_
# elset
#  define  _EXTERN_ extern
# endif
```

Apparently the CMT developers achieved this feat on Windows by dispensing with the DATA qualifier in their *.def file...this may not be absolutely technically correct but it's ok and simplifies life:

Ok, one only need a little bit into the linker documentation to find:
Note that when you export a variable from a DLL with a .def file, you do not need to specify *_declspec(dllexport)* on the variable. However, in any file that uses the DLL, you must still use *_declspec(dllimport)* on the declaration of data.
so, with the "new" cmt we are perhaps not being as politically correct as we should be but it means we don't need to insert dllimport into the code. With my fix we would need to insert the dllimport stuff.
Tracy

This apparently works for Jim in ST, but we had to ultimately revert to Joanne's original workaround in CalibData that links in the object file on windows - so v1r18 was not as grand as we had hoped.