WARWICK
THE UNIVERSITY OF WARWICK

Update on Geant4 CMake, Modularization and Data

Ben Morgan

2: CMake Updates for Geant4 10.4

- Mostly bug fixes and tidying, apart from
- Removal of obsolete `FindX` modules: ROOT, XercesC
  - *Either provided by CMake, or* `XConfig.cmake` *files*
- Removal of `GEANT4_BUILD_EXAMPLES` option
  - *Not used and not useful!*
- Simplification of Windows DLL builds with symbol auto-export (*still in testing but expected to reach release*)

3: Library Modularization: Reminder

- **Aim: single scheme of organising (sub)category source code into binary libraries**
- Balance:
  - **usability** (fewer libraries the better)
  - **modularity** (only build/use what you need)
  - **minimisation of dependencies**
- Improve developer and user interfaces to CMake

4: Library Modularization: Status

- CMake implementation close to “beta”
  - Requires Windows DLL improvement from earlier
  - Support for category level unit tests to be added

- **Zero impact on your code/files** - CMake interfaces (variables/commands) are fully backward compatible, and library “scheme” is identical

- Use of old/new library modularisation implementation chosen by a CMake option - allows comparative testing

5: Library Modularization: Next Steps

- **Due to time constraints, propose to merge new implementation immediately after 10.4 release**

- Default remains old implementation, start testing new implementation to validate that it builds the current global libraries identically and without loss of performance

- Once validated, switch default to new implementation and begin studies/discussion on splitting/merging current global libraries

- **To reiterate: You will not be impacted or required to change any CMake files (e.g. sources.cmake) during this period!**

6: Library Modularization: Tasks for 10.X in 2018

- When the new CMake implementation becomes the default, you *can* start simplifying the sources.cmake files for your categories/modules
  - You'll be able to remove things like `include_directories` commands, and only have to state which other Geant4 modules yours uses
  - Full documentation will be provided of course!
- **Splitting/Merging of libraries will be discussed with WGs during preparation of 2018 workplans**
- Anticipate that EM/Hadronic/Processes, Persistency, Run/Event/Digits and UI/Vis will be the main WGs involved
  - Based on code size or use of external dependencies

7: Additional tasks now delayed to 2018

- Improve ability for an install of Geant4 to be moved ("relocatability")
  - *Requested by ATLAS/FNAL, and helpful for binary packagers in general (Homebrew/Spack/vcpkg)*
- Move `-D` flags to `#define`s in "G4Config.hh" header
  - *Guarantees consistency between build/use time*
- Build/Install of Geant4Py as optional component of Geant4's main build/install

8: Data Libraries in Geant4

- Current usage of data libraries (e.g. G4EMLOW) has several issues:
  - **Usability**: full usage requires setting of 10 environment variables, some are mandatory
  - **Versioning**: not always clear or checked that data library versions (10 of them) are compatible with used version of Geant4
  - **Performance**: Use of many small text files causes issues, amongst others

9: Proposal: Data Library Task Force

- Feel it is needed because issue crosses several working groups (EM/Hadronic Physics, Particles, Materials, Software/Testing)

- **Please discuss this week! A possible remit would be:**

  - *Design/Propose a C++ interface for accessing data to minimise user configuration, allow developer override*

  - *Design/Propose a versioning scheme and C++ interface to ensure compatibility between a running Geant4 program and loaded data*

  - *Identify performance issues in data libraries, and propose alternative formats and/or access methods to resolve these*

10: Questions?