Improvements in the iLCSoft build system

Jan Engels

Desy, 2nd March 2011

Outline

- Overview on the current iLCSoft build system
- Motivation for improvements
- Deprecated features
- Introducing new features (to be released in v01-11)
- Summary
- Outlook

Overview on the current iLCSoft build system

- **iLCSoft**
  - **Core packages**
    - LCIO, GEAR, LCCD, CED, ...
  - **Marlin plugins**
    - MarlinReco, LCFIVertex, MarlinPandora, CEDViewer, ...
  - **External dependencies**
    - ROOT, GSL, CLHEP, CERNLIB, ...
  - Currently, dependencies in iLCSoft packages are treated individually between all packages, i.e.:
    - If A->B and B->C and D->A
    - Then: D->A, D->B, D->C
    - Not: D->A->B->C, where D depends on A and therefore also gets B and C
    - More on this topic later in this talk

Overview on the current iLCSoft build system

- **ilcinstall**
  - python script for installing an iLCSoft release
    - core packages + marlin plugins + external dependencies
      - few exceptions (mysql, java, geant4)
    - supported platforms: SL4_32, SL5_32, SL5_64
      - Also works under other platforms: mac osx, ubuntu, … (best effort to support)
    - used for **AFS reference installations** (next slide)
    - generates **init_ilcsoft.sh** environment script (mostly required for external dependencies)
      - export ROOTSYS=/path/to/ROOT
      - export LD_LIBRARY_PATH=/path/to/ROOT/lib:$LD_LIBRARY_PATH
      - export PATH=/path/to/ROOT/bin:$PATH
      - (…)
    - generates **ILCSoft.cmake** cmake script
      - SET(LCIO_HOME "/path/to/LCIO" CACHE PATH "Path to LCIO" FORCE)
      - SET(Marlin_HOME "/path/to/Marlin" CACHE PATH "Path to Marlin" FORCE)
      - SET(CMAKE_MODULE_PATH "/path/to/CMakeModules" CACHE PATH "CMakeModules" FORCE)

Overview on the current iLCSoft build system

- **AFS reference installations**
  - **/afs/desy.de/project/ilcsoft/sw/(ARCH)/**
    - x86_64_gcc41_sl5 – SL5_64 and compatible
    - i386_gcc41_sl5 – SL5_32 and compatible
    - i386_gcc34_sl4 – SL4_32 and compatible

  - **external tools (shared between releases)**
    - /afs/desy.de/project/ilcsoft/sw/(ARCH)/
      - ./mysql/5.0.45/{bin,lib,include}
      - ./root/5.28.00/{bin,lib,include}
      - ./cernlib/2006/{bin,lib,include}
      - (...)

  - **ilcsoft releases**
    - /afs/desy.de/project/ilcsoft/sw/(ARCH)/**v01-10-01/**
      - ./CED/v01-01-01/{bin,lib,include}
      - ./lcio/v01-51-02/{bin,lib,include}
      - ./MarlinReco/v00-19-01/{bin,lib,include}
      - ./root/5.27.06 (symlink to ../root/5.27.06)
      - init_ilcsoft.sh
      - ILCSoft.cmake
      - (...)

Overview on the current iLCSoft build system

- **CMake**
  - CMakeLists.txt
    - text files containing instructions used to build iLCSoft packages
  - CMakeModules
    - utilities and macros used within the CMakeLists.txt files
  - Configure and build packages using cmake:
    - . /path/to/ilcsoft/init_ilcsoft.sh
    - mkdir build
    - cd build
    - cmake -C $ILCSOFT/ILCSoft.cmake ..
    - make install

Motivation for improvements

- The iLCSoft framework is continuously growing (~25 packages + external dependencies)
- Initial versions of ilcinstall exclusively implemented for building packages using "traditional makefiles", later on *upgraded* for using cmake as well
- Initial versions of CMake scripts for the iLCSoft packages
  - were designed to be backwards-compatible with *older* "makefiles"
    - led to use non-standard cmake constructs
  - checking of package versions was not high priority
  - BuildSetup.cmake including hard-coded paths proved to be a pitfall for most users
  - Other pitfalls, e.g. -DStreamlog_HOME=... -DIcio_HOME
- Inter-package dependencies is a very tricky subject and needs careful handling, otherwise it can easily get into a "dependency-hell"
  - why do I need to specify LCIO_HOME when Marlin anyways depends on LCIO?
- **Simplify usage of cmake build tools for both, developers and users!**

Deprecated features (as of next release v01-11)

- Deprecated BuildSetup.cmake files
  - pitfall for most users due to hard-coded paths

- Deprecated use of PKG_HOME cmake variables
  - cmake -DLCIO_HOME=/path/to/lcio ..
  - use standard cmake PKG_DIR (or CMAKE_PREFIX_PATH)

- Deprecated use of BUILD_WITH cmake variables
  - cmake -DBUILD_WITH="RAIDA LCCD ..." ..
  - use standard cmake FIND_PACKAGE

- Deprecated macros: LoadPackage.cmake and CheckDeps.cmake
  - not flexible enough to handle dependencies properly
  - use standard cmake FIND_PACKAGE

FIND_PACKAGE usage examples

- **Typical FIND_PACKAGE usage**
  - FIND_PACKAGE(Marlin REQUIRED)
  - INCLUDE_DIRECTORIES(${Marlin_INCLUDE_DIRS})
  - LINK_LIBRARIES(${Marlin_LIBRARIES})

- **Version checking**
  - FIND_PACKAGE(ROOT 5.28 REQUIRED)
  - FIND_PACKAGE(LCCD 1.2 EXACT)
    - IF(LCCD_FOUND)
      - INCLUDE_DIRECTORIES(${LCCD_INCLUDE_DIRS})
      - LINK_LIBRARIES(${LCCD_LIBRARIES})
      - ADD_DEFINITIONS("-DMY_CODE_USES_LCCD")
    - ENDIF(LCCD_FOUND)

- **Using COMPONENTS**
  - FIND_PACKAGE(ROOT 5.28 REQUIRED COMPONENTS Gdml Geom XMLIO)
  - LINK_LIBRARIES(${ROOT_LIBRARIES})
  - LINK_LIBRARIES(${ROOT_COMPONENT_LIBRARIES})
  - LINK_LIBRARIES(${ROOT_GDML_LIBRARY})

  - FIND_PACKAGE(ILCUTIL REQUIRED COMPONENTS ILCSOFT_CMAKE_MODULES streamlog)
  - INCLUDE_DIRECTORIES(${streamlog_INCLUDE_DIRS})
  - LINK_LIBRARIES(${streamlog_LIBRARIES})

Configuring with cmake

- CMAKE_PREFIX_PATH
  - Search path for FIND_PACKAGE (Similar to typical PATH variable on *nix systems)
    - export CMAKE_PREFIX_PATH=/path/to/LCIO:/path/to/Marlin:/path/to/ROOT
    - cmake ..
  - alternatively may be specified directly on the command line
    - cmake -DCMAKE_PREFIX_PATH="/path/to/x;/path/to/y;/path/to/z" #(1)
    - cmake -DCMAKE_PREFIX_PATH=/path/to/x\;/path/to/y\;/path/to/z #(2)
  - note the different use of ';' and ':' !!!
  - also note the use of "" in (1) or escaping of ';' in (2)

- Alternatively PKG_DIR variables may be used (not PKG_HOME)
  - cmake -DLCIO_DIR="/path/to/lcio" -DMarlin_DIR=/path/to/Marlin" ..
  - PKG_DIR variables always overwrite CMAKE_PREFIX_PATH settings

Reduced dependencies

- Dependencies have been reduced
  - First Slide:
    - D->A->B->C
  - Not:
    - D->A and D->B and D->C

- Building simple Marlin plugin is now much easier:
  - FIND_PACKAGE(Marlin REQUIRED)
  - INCLUDE_DIRECTORIES(${Marlin_INCLUDE_DIRS})
  - LINK_LIBRARIES(${Marlin_LIBRARIES})
  - cmake -DILCUTIL_DIR=/path/to/ilcutil -DMarlin_DIR=/path/to/Marlin
  - **No need to know about LCIO, GEAR, streamlog or CMAKE_MODULE_PATH !**

New package: ILCUTIL

- ILCUTIL is a new utilities package for the iLCSoft software framework
  - Requires CMake >= 2.8.2
- "meta-package" grouping together a set of small utility packages needed by most iLCSoft packages
  - streamlog
  - ILCTEST
  - ILCSOFT_CMAKE_MODULES (previously known as CMakeModules
- Further packages can be added as needed

Summary

- CMake scripts have been improved and significantly simplified
  - Reduced dependencies
  - Use more standard CMake-way of doing things helps reducing maintenance costs
  - Finding and linking against other packages improved
  - Added Version checking (using FIND_PACKAGE)

- Introduced new “meta-package” ILCUTIL
  - Easily expandable
  - Contains cmake macros and core utility packages

Outlook

- Split ilcsoft releases into 2 stages
  - core-tools (+ external dependencies)
  - physics-tools (mostly Marlin plugins)
- Use common installation prefixes for ilcsoft tools (core/physics-tools)
  - simplify cmake configure steps (setting of CMAKE_PREFIX_PATH, PKG_DIR variables)
- Use new ExternalProject feature available in more recent cmake versions (>=2.8.2)
  - Possibility to use a single CMakeLists.txt to install a whole iLCSoft release!
  - Optionally allow packages to install their dependencies
    - CED->freeglut
    - Marlin->LCIO+GEAR (optionally RAIDA+LCCD+CLHEP)
    - LCCD → CondDBMySQL
    - RAIDA → ROOT