# Art Build: Unit Tests and CMake Tools

Ben Morgan

THE UNIVERSITY OF WARWICK

LBNE Code

LArSoft

LArCore LArFoo LArBar ...

FNAL SDK : art + (fhicl, …, cpp0x)

FNAL UPS : ups, gcc, ROOT, etc

Linux/BSD OS

# LBNE Software Stack

Working on “FNAL SDK”, Patrick on LArSoft and “UPS” layers

# Part 1: Art Status and Unit Tests

The Foundation of LArSoft

# Art: Repository Status

- GitHub copy of Redmine Repo :
  - https://github.com/LBNE/fnal-art/
- Github master == Redmine master (manual pull/push)
- Dedicated branch for removal of UPS/cetbuildtools:
  - https://github.com/LBNE/fnal-art/tree/remove-ups-1.11.3/
- Maintaining this with stable code from upstream v1.11.3

# Art Build Status

- Since last report (Slides on Indico):
- Dictionary generation complete
- Bug fixes (thanks Patrick!) from first steps at worch integration
- Addition of unit tests

# Art Unit Tests

- Mixture of Boost.Unit and CPPUnit (Redmine #7029)
- Added “FindCPPUnit.cmake” module
- Most tests added as of commit 6aa0a34
- CMake option to enable build of tests, then run via

```
$ make
$ make test
```

# Art Unit Tests Status

- Tests running for everything except
  - Integration, Framework/IO/Root, Framework/Principal
  - Latter two depend on Integration, which is very large
- Majority pass except for
  - GroupSelector_t: Appears to be path/naming issue with plugin lib
  - StateMachine_t_XY: Due to missing cetbuildtools supplied shell script, which must also be on the PATH

# Art Build: Next Steps

- Finish addition of tests
- Resolve failing tests
- Patch as needed from worch install experience
- Provide CMake convenience tools for building plugins
- White Paper for detailed criticism of FNAL system and proposal of these fixes.

# Part 2: Whither cetbuildtools?

If we need a CMake layer, how to do it properly.

# Why use cetbuildtools?

- “Common” settings for builds - compiler flags etc
- “Convenience” CMake functions for building/installing stuff, e.g. “art_make”
- UPS integration - “easily” locate needed packages.

# What's wrong with cetbuildtools?

- Build settings only considers GCC/Linux
- "Convenience" functions enforce a specific layout of source code, names and UPS install.
  - Heavy singular interfaces with many switches to work around exceptions
- Propagates itself into client projects - so difficult to wall ourselves off from it
- Often contains functionality that is really Art-specific, or replaces perfectly good CMake builtins

# Common CMake Functionality

- Nevertheless, having common CMake functionality in a base project can be useful
  - Extra FindXXX.cmake modules, e.g. FindTBB.cmake
  - Build settings and code generation macros (c.f. ROOT_GENERATE_DICTIONARY from ROOTConfig.cmake)
- **Idea:** Replace cetbuildtools, keeping needed functionality and implementing it properly.
- **Inspiration:** KDE extra-cmake-modules

# “HEPCMake”

- Slightly overblown name, but intended to be general
  - Get it at: https://github.com/drbenmorgan/HEPCMake
- Partitions modules into task areas (finding stuff, enforcing a particular lab policy)
- Can be directly included in any project, or for sharing, installed as any other package.
- Feedback to upstream packages, e.g. ROOT dictionary generation via CMake not optimal? Speak to ROOT devs.

# Using cetbuildtools…

```
#My CMake script…

SET(CETBUILDTOOLS_VERSION $ENV{CETBUILDTOOLS_VERSION})

IF (NOT CETBUILDTOOLS_VERSION)
    MESSAGE (FATAL_ERROR "ERROR: setup cetbuildtools to get the cmake
modules")
ENDIF()

set(CMAKE_MODULE_PATH ${CMAKE_CURRENT_SOURCE_DIR}/Modules
                      $ENV{CETBUILDTOOLS_DIR}/Modules
                     ${CMAKE_MODULE_PATH})
```

# Using HEPCMake…

```
#My CMake script…
find_package(HEPCMake REQUIRED)
```

# "artTools.cmake"

- Module used/installed by Art to provide better convenience wrappers following CMake style:

```
art_add_service(my_service my.h my.cc)
install(TARGETS my_service DESTINATION lib)
```

- Instead of

```
art_make(EVERYTHING UNLESS SIDE EFFECTS)
```

# Next Steps

- Comments/Questions?

- Easy to add HEPCMake (or whatever name) into the worch build.

- Easy to locate

- Easy to use - but needs decision on use and feedback!