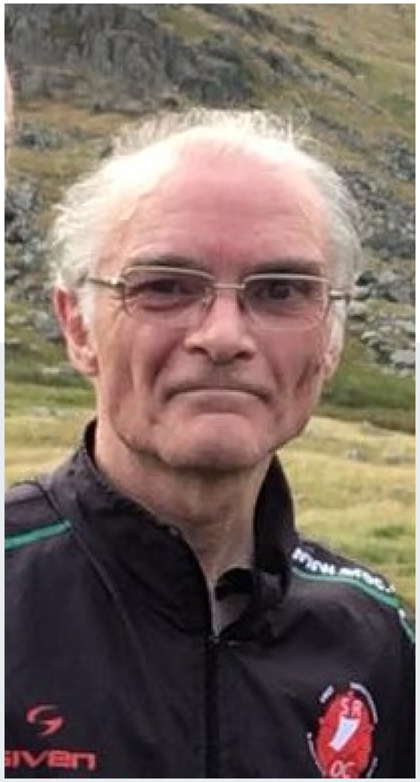
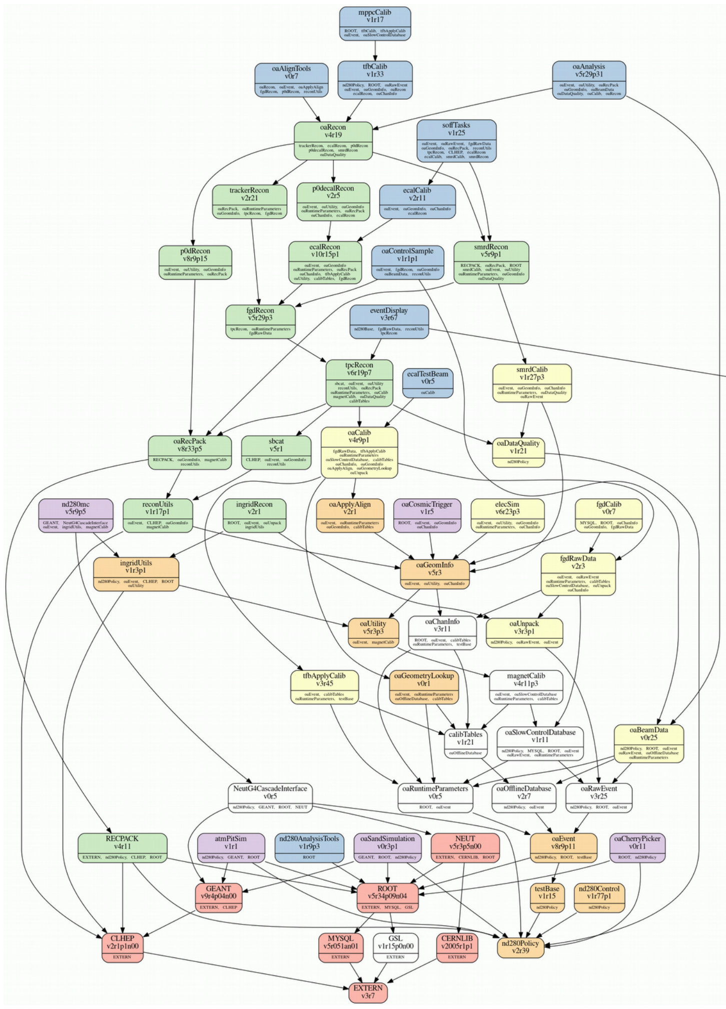
ND280 software - emulating CMT with CMAKE

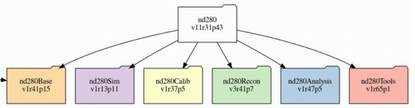
- Alex Finch



ND280 Software



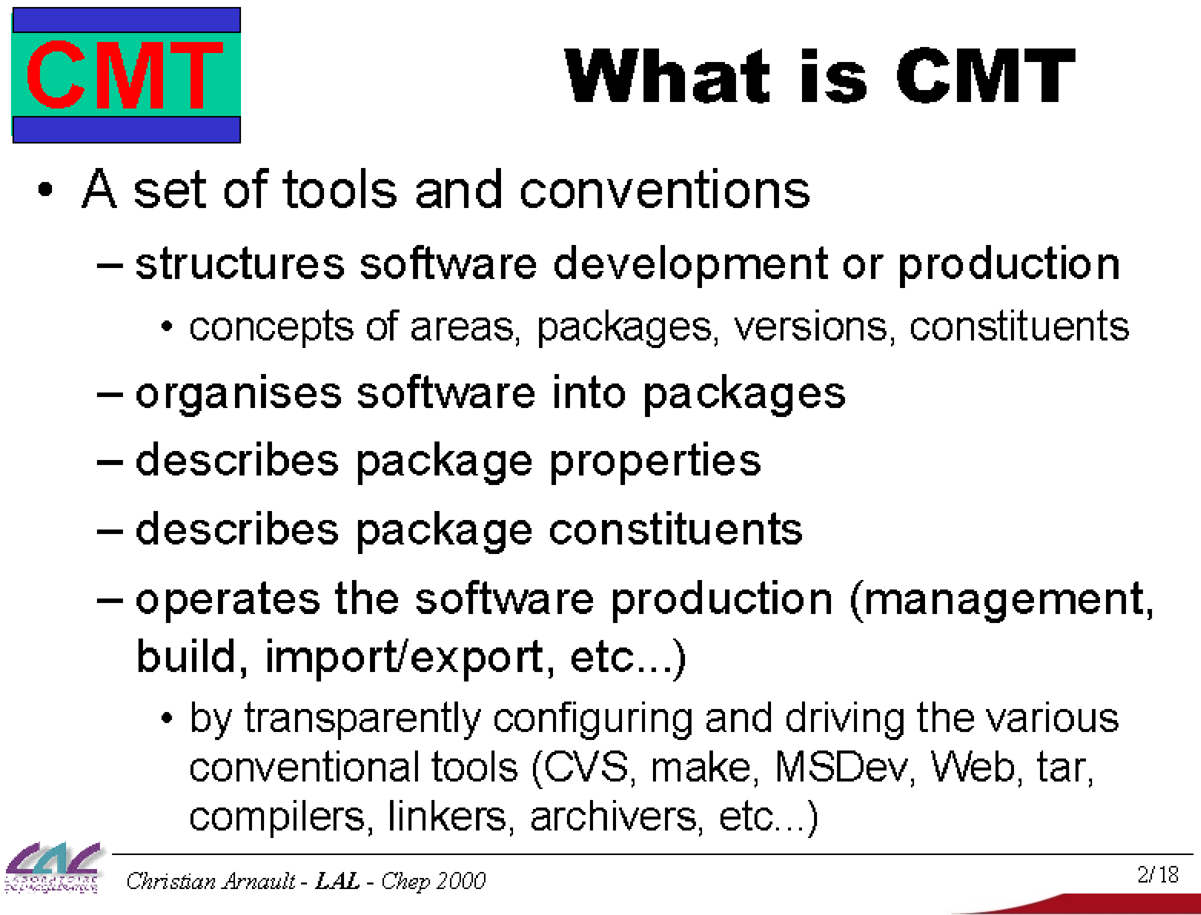
- *Set up in 2006*
- *~70 packages*
- *Split into 6 master sets for convenience*



- *Controlled by CMT*
- *Version control with CVS*
- *An "nd280 release" is a defined set of versions of each package.*

CMT

- *"Makefile generator"*
- *Written by particle physicist*
- *Only used in HEP.*



What is CMT

- A set of tools and conventions
  - structures software development or production
    - concepts of areas, packages, versions, constituents
  - organises software into packages
  - describes package properties
  - describes package constituents
  - operates the software production (management, build, import/export, etc...)
    - by transparently configuring and driving the various conventional tools (CVS, make, MSDev, Web, tar, compilers, linkers, archivers, etc...)

Christian Arnault - LAL - Chep 2000 2/18

CMT …

- *Single file "**requirements**" defines all the information CMT needs:*
  - Packages this one depends on, using versioning (v<major id>r<minor id>p<patch id>)
  - Executables that need building
  - Non standard things that need doing
  - Non standard compile/link commands
- *Knows about CVS*
- *Creates makefiles*
- *No longer actively maintained*

CMAKE

- *Industry standard "makefile generator" for building software.*
- *Highly configurable*
- *Lots of documentation*
- *Actively developed*

GIT

- *Industry standard version control system*
- *Need I say more?*

If it Ain’t Broke Don’t Fix It.

- *Why go to the bother of changing from CMT to CMAKE/GIT?*
- *ND280 is being upgraded.*
- *Software needs to change to match. GIT is better for branching etc. which will be a big advantage.*
- *Expect to run nd280 software for many more years.*
- *New students/post docs are often already familiar with git and cmake.*
- *Active development/support for CMAKE/GFIT . Lots of documentation - just google it, or even buy a book!*

Converting from CMT/CVS to CMAKE/GIT

- *Tried to keep close the to the CMT structure/philosophy.*
- *CMAKE does not know about GIT natively*
- *Need to provide the "glue" between GIT and CMAKE in a separate "pilot" package.*
- *Took advantage of relatively simple structure of most nd280 packages:*
  - Package **X** builds a library called lib**X**.so from a bunch of C++ source files in the *src* directory. It may also build some executables from a main routine in the *app* directory.
  - It is documented with doxygen.
- *Take advantage of the fact that all "external" packages can now be built with CMAKE.*

A typical ND280 package in CMAKE

- ***CMakeLists.txt*** *is equivalent of CMT* ***requirements*** *file:*

Standard cmake

include nd280 specific functions

define the version number of this package

initialise the project

define the dependencies

create a standard shared object library

create an executable

finish off

```
# CMakeLists.txt for <package> package. It creates a library ...

cmake_minimum_required(VERSION 3.9 FATAL_ERROR)

find_package(nd280SoftwarePolicy 3.1)

if( NOT nd280SoftwarePolicy_FOUND)
  message(FATAL_ERROR " nd280SoftwarePolicy not found - abort ")
endif()

include(<package>PackageVersion.cmake)

ND280_PROJECT(<package> ${PACKAGE_VERSION})

include(<package>ND280_USE.cmake)

ND280_STANDARD_LIBRARY()

ND280_EXECUTABLE(ExecutableName MainRoutine.cxx)

ND280_END_PROJECT()
```

Supporting files…

- *<package>PackageVersion.cmake:*

  Defines the package version number:

```
set(PACKAGE_VERSION "major.minor.patch" )

…
```

- <package>ND280_USE.cmake:

  Defines the packages this one depends on. Try to define the minimum set necessary.

  - List of "use statements"

```
ND280_USE(oaRawEvent )
```

  - Only "master packages" contain version numbers,e.g.

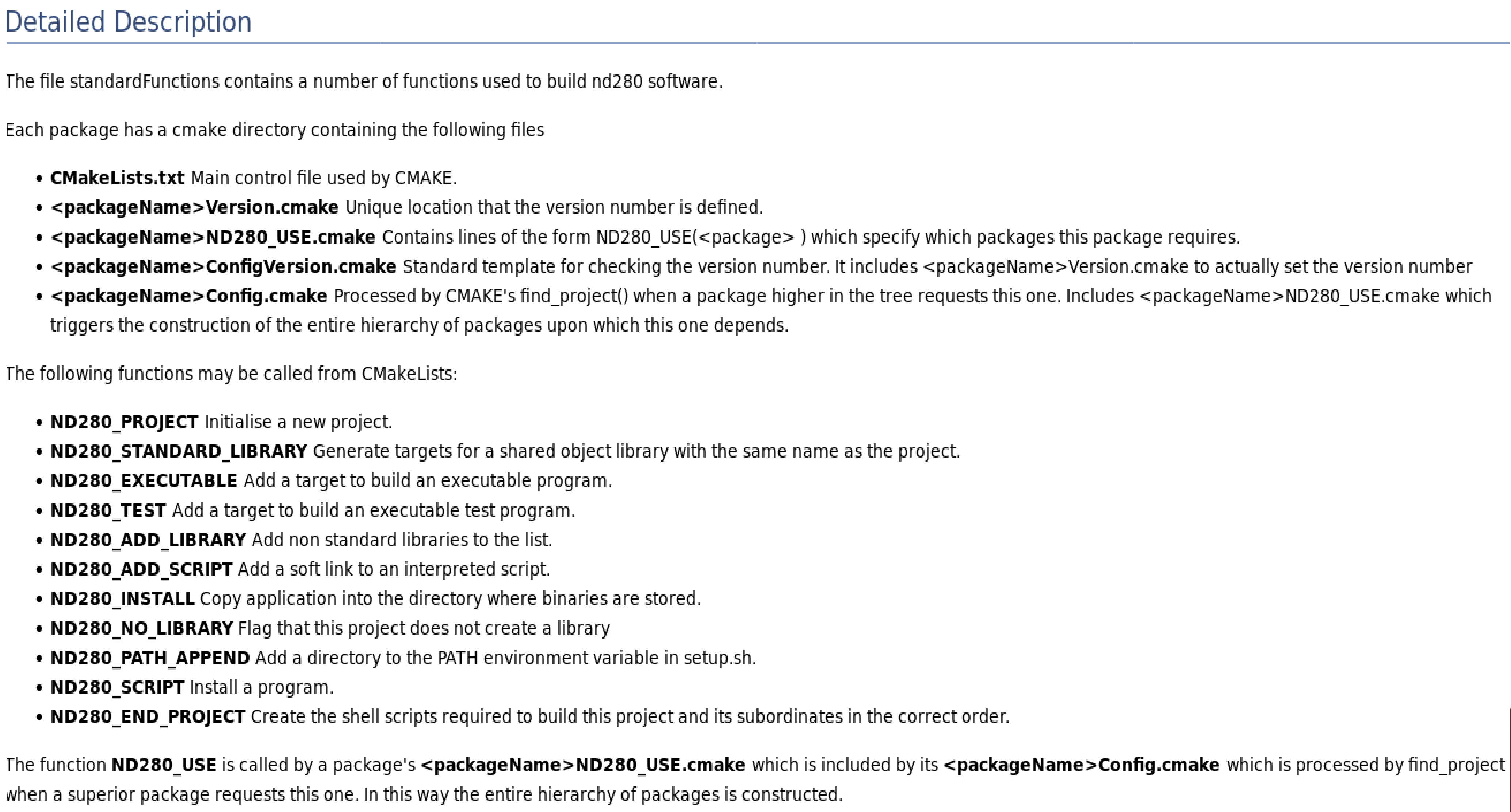
```
reconMasterND280_USE.cmake

ND280_USE(reconUtils 1.35.1 )
ND280_USE(RECPACK 4.17.1 )
ND280_USE(recPackRecon 8.53.1 )
ND280_USE(sbcatRecon 5.5.1 )
ND280_USE(p0dRecon 9.9.1 )
ND280_USE(tpcRecon 6.33.1 )
ND280_USE(trexRecon 2.35.1 )
ND280_USE(fgdRecon 6.9.1 )
…
```

Where the work gets done…

- *nd280SoftwarePolicy includes a single file "standardFunctions.cmake" which defines a set of ND280 specific functions which do the heavy lifting.*

Doxygen style documentation is available …



Detailed Description

The file standardFunctions contains a number of functions used to build nd280 software.

Each package has a cmake directory containing the following files

- **CMakeLists.txt** Main control file used by CMAKE.
- **<packageName>Version.cmake** Unique location that the version number is defined.
- **<packageName>ND280_USE.cmake** Contains lines of the form ND280_USE(<package>) which specify which packages this package requires.
- **<packageName>ConfigVersion.cmake** Standard template for checking the version number. It includes <packageName>Version.cmake to actually set the version number
- **<packageName>Config.cmake** Processed by CMAKE's find_project() when a package higher in the tree requests this one. Includes <packageName>ND280_USE.cmake which triggers the construction of the entire hierarchy of packages upon which this one depends.

The following functions may be called from CMakeLists:

- **ND280_PROJECT** Initialise a new project.
- **ND280_STANDARD_LIBRARY** Generate targets for a shared object library with the same name as the project.
- **ND280_EXECUTABLE** Add a target to build an executable program.
- **ND280_TEST** Add a target to build an executable test program.
- **ND280_ADD_LIBRARY** Add non standard libraries to the list.
- **ND280_ADD_SCRIPT** Add a soft link to an interpreted script.
- **ND280_INSTALL** Copy application into the directory where binaries are stored.
- **ND280_NO_LIBRARY** Flag that this project does not create a library
- **ND280_PATH_APPEND** Add a directory to the PATH environment variable in setup.sh.
- **ND280_SCRIPT** Install a program.
- **ND280_END_PROJECT** Create the shell scripts required to build this project and its subordinates in the correct order.

The function **ND280_USE** is called by a package's **<packageName>ND280_USE.cmake** which is included by its **<packageName>Config.cmake** which is processed by find_project when a superior package requests this one. In this way the entire hierarchy of packages is constructed.

ND280_USE

- *ND280_USE function builds the hierarchy of packages needed by this one.*
- *Wrapper for cmake function **find_package***
  - searches for package by name and version
  - by convention, in directory called <package>_<version>
  - If found, runs configuration file, which normally just includes <package>ND280_USE.cmake which calls ND280_USE…

Main ND280_ functions

- ***ND280_STANDARD_LIBRARY***
  Wrapper for add_library
- ***ND280_APPLICATION:***
  wrapper for add_executable
- ***ND280_END_PROJECT***
  Creates scripts to build the project and its dependencies in the right order.
  Also generates setup scripts including any package specific ones.

Building the s/w

- *cmake ../cmake*
- *../bin/setup.sh*
- *../bin/makeAll.sh*

ND280 S/W on GITLab

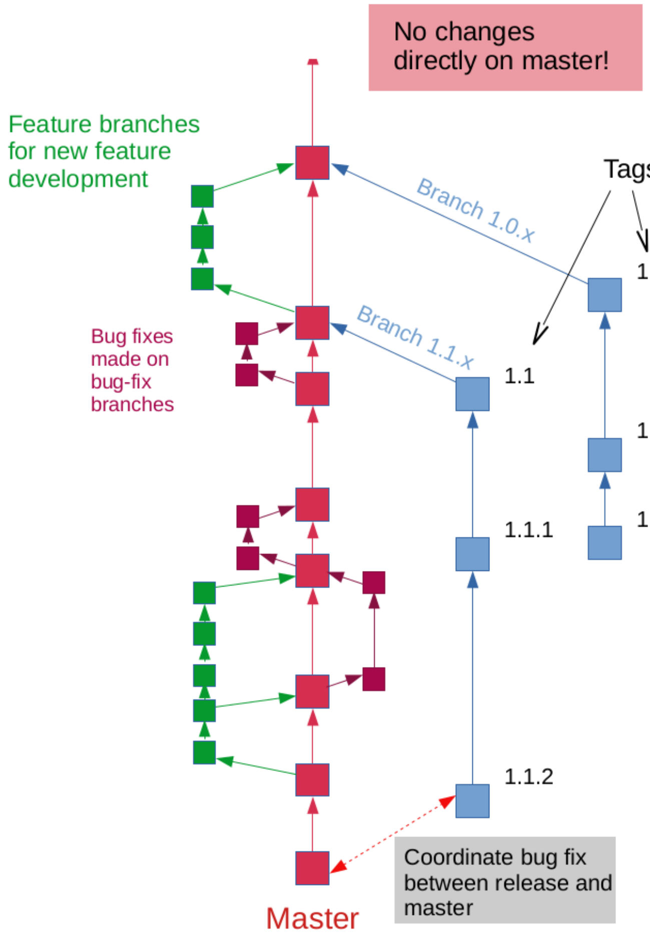
- *Package version ->branch + tag*
- *patches -> tag*

nd280SoftwarePilot :

clones packages

"git checkout" correct version

rename directory to keep CMAKE happy


No changes directly on master!
Feature branches for new feature development
Tags
Branch 1.0.x
1.0
Branch 1.1.x
Bug fixes made on bug-fix branches
1.1
1.0.1
1.0.2
1.1.1
1.1.2
Coordinate bug fix between release and master
Master

Issues with Externals

- *Use **ExternalProject_Add** to hand craft each "External" package*
- *Some external packages, e.g. ROOT, Geant4*
    - Create their own Config files when they are built
    - But we need them before this (during cmake stage) to create the hierarchy of packages
    - Need "placeholder" config files which load the generated config files if they exist but always satisfy find_package

Conclusions and Future Directions

- *ND280's existing software arrangements needed updating to support the hardware upgrade.*
- *Used industry standard CMAKE + GIT*
- *Emulated CMT's approach where possible*
- *Succesfully converted after ~1 year's work.*
- *SInce becoming official rapid development of software for upgraded detector*
- *Further development work to take advantage of CI possibilities with GITLab*