Current CMT use notes

+

CMake evaluation

F.Carbognani

Currently we are making use of the CMT plug-in for SVN (CMT support CVS and SVN) to proceed with the installation of a packages baseline this way:

- Create a virtual package in SVN that just contain a requirements file defining the baseline (such as DASet, BasicSet, etc.), commit and tag it (ex: v1r2)

- Go into the area where the installation should happen and then:
  - ◆ *cd /<install_area>*
  - ◆ *cmt checkout -R -r v1r2 DASet* **(with this command all the packages needed for the dependencies defined by the "use" relationship are recursively exported from the SVN archive)**

- Then the cmt broadcast is used to compile the whole set of packages in the correct order as defined by the use relationship:
  - ◆ *cd /<install_area>/DASet/v1r2/cmt*
  - ◆ *cmt broadcast cmt config*
  - ◆ *cmt broadcast make*

- That's it!

The availability of such CMT plug-in was one of the reason to choose SVN as CVS successor

- The evaluation of the CMake/CPack/CTest/CDash toolsuite has proceeded. This toolchain constitute a complete software management solution and allow to:
  - **build the software**
  - **package distributions (tar, rpm and deb files supported**)
  - **support automated test and report via dedicated web page**
- Available CMT to CMake translation scripts (made for the GAUDI Project) have been studied and an early Virgo version drafted (available on the package CMT2CMake).
- Need to create an equivalent of VirgoPolicy: VirgoPolicy.cmake
- The current version of the cmt2CMake.py script is able to manage the simplest requirements files. The CMakeLists.txt for a simplified Fr requirements look like this:

```
#############################################################################
# CMakeLists.txt file for building v8r15
#############################################################################
cmake_minimum_required (VERSION 2.8)
set(CMAKE_MODULE_PATH /users/fcarbogn/CMakeTests/CMT2CMake/v1r0/mgr/)
INCLUDE(VirgoPolicy)
VIRGO_USE_PACKAGE(root)
VIRGO_USE_PACKAGE(VirgoPolicy)
include_directories(${CMAKE_CURRENT_SOURCE_DIR}/src/zlib)
#---Libraries--------------------------------------------------------------
SET( Frame_srcs FrameL.c FrIO.c FrFilter.c zlib/*.c )
VIRGO_LINKER_LIBRARY(Frame ${Frame_srcs} LIBRARIES Frame)
SET( FrameROOT_srcs FrameL.c FrIO.c FrFilter.c zlib/*.c
../Linux-i686-SL4//FR_G_ROOT.cc )
VIRGO_LINKER_LIBRARY(FrameROOT ${FrameROOT_srcs} LIBRARIES Frame)


#---Executables------------------------------------------------------------
VIRGO_EXECUTABLE(FrCopy FrCopy.c LIBRARIES Frame)
VIRGO_EXECUTABLE(FrDump FrDump.c LIBRARIES Frame)
VIRGO_EXECUTABLE(FrCheck FrCheck.c LIBRARIES Frame)
```

- Now there is the need to be able to manage the most complex versions of requirements files. It may become an exponential effort (size of the equivalent scripts developed for GAUDI is a bit scaring)

- The CTest/CDash potentialities have been checked on existing Virgo packages and a test web pages have been setup. Figure below report a CDash screenshot. Available also online for example at:

  - **http://sprserver.ego-gw.it/CDash/index.php?project=Virgo&date=20120822**