Development Environment for GeantV (and others)

Pere Mato

GeantV Meeting, 18 September 2018

Introduction

- To develop GeantV requires to setup an environment with a large number of externals packages:
  - **ROOT**, Python, Boost, fftw, graphviz, GSL, mysql, xrootd, R, numpy, tbb, blas, pythia6
  - **Geant4**, XercesC, CLHEP, expat, hepmc3
  - **VecGeom**, umesimd, veccore,Vc, veccorelib
  - **benchmark, ninja, CMake**
- Building and installing these externals take a lot of time and effort
- These packages is a **sub-set of the LCG packages** that we do already build regularly
  - So, lets use the same tools and infrastructure

CVMFS Releases

- The easiest (if possible) is to access the releases from CVMFS
  - limited number of platforms (slc6, centos7, ubuntuX, mac10X,..)
- A dedicated stack for GeantV is re-build every night (dev)
  - used by the GeantV nightlies

```
$ ls -1 /cvmfs/sft.cern.ch/lcg/views/devgeantv/latest/

x86_64+avx2+fma-centos7-gcc7-opt
x86_64+avx2+fma-slc6-gcc62-opt
x86_64-centos7-gcc48-opt
x86_64-centos7-gcc49-opt
x86_64-centos7-gcc62-opt
x86_64-centos7-gcc7-opt
x86_64-slc6-gcc52-opt
x86_64-slc6-gcc62-opt
x86_64-slc6-gcc7-opt
x86_64+sse3-centos7-gcc48-opt
x86_64+sse3-centos7-gcc62-opt
x86_64+sse3-slc6-gcc52-opt
x86_64+sse3-slc6-gcc7-opt
x86_64+sse3-ubuntu1604-gcc54-opt
x86_64-ubuntu1604-gcc54-opt
```

CVMFS Releases (2)

- In addition another dedicated stack for GeantV is build less regularly
  - more stability, with only ‘released versions’

```
$ ls -1 /cvmfs/sft.cern.ch/lcg/views/geant-latest  # aka geantvext20180918

x86_64-centos7-gcc62-opt
x86_64-mac1012-clang90-opt
x86_64-mac1013-clang91-opt
x86_64-slc6-gcc62-opt
x86_64-slc6-gcc7-opt
x86_64-ubuntu1604-gcc54-opt
x86_64-ubuntu1804-gcc7-opt
```

- To set the environment need to source a setup file, E.g.

```
$ source /cvmfs/sft.cern.ch/lcg/views/geant-latest/x86_64-centos7-gcc62-opt/setup.sh

x86_64-centos7-gcc62-opt
x86_64-mac1013-clang90-opt
x86_64-slc6-gcc62-opt
x86_64-slc6-gcc7-opt
x86_64-ubuntu1404-gcc48-opt
x86_64-ubuntu1604-gcc54-opt
```

Full sequence of commands

- Git clone, cmake, make should just work!

```
$ source /cvmfs/sft.cern.ch/lcg/views/geant-latest/x86_64-centos7-gcc62-opt/setup.sh
$ git clone https://gitlab.cern.ch/GeantV/geant.git
$ mkdir build; cd build
$ cmake ../geant/
...
$ make -j10
...
```

- Some caveats
  - geant-latest (from January) does not have ‘veccorelib’
  - devgeantv/latest (nightly) should work

Non-CVMFS Releases

- In some cases you would like to have local installations of the externals, e.g.
  - No CMVFS installed, working in a network-less location, did not find the exact match of arch/os/compiler, etc.
- For this use case we have developed an utility command ('lcgcmake') to provide local installations
  - downloading existing binaries if matches arch/os/compiler, and exact version of package and depends (hash value)
  - building from sources
  - a combination of both

Installation of 'lcgcmake'

- Download package and set PATH

```
$ git clone https://gitlab.cern.ch/sft/lcgcmake.git
$ export PATH=$PWD/lcgcmake/bin:$PATH
```

- The command **lcgcmake** is now available

```
$ lcgcmake --help

This command drives the full process of building a LCG release using the lcgcmake tool.

positional arguments:
    {version,configure,config,conf,install,show,sh,run}
                        available sub-commands
    version             print the version of lcgcmake
    configure (config,conf)
                        configure the lcgcmake session by selecting the
                        software stack version, etc.
    install             Install or build a set of given targets with all their
                        dependencies
    show (sh)           get information from the lcgcmake session
    run                 run a command with the just installed software stack

optional arguments:
  -h, --help            show this help message and exit
  --verbose             Increase logging verbosity
  -q, --quiet           Decrease logging verbosity
  -a ARCH, --arch ARCH  processor architecture
  -o OSVERS, --os OSVERS
                        operating system keyword (e.g. slc6, mac1013, etc.)
```

Installing the GeantV externals

- Basic commands: configure, install, run

```
$ lcgcmake configure --prefix <installprefix>
$ lcgcmake install GeantV-externals
$ lcgcmake run
```

- Once the installation is finished, execute the standard commands to build GeantV

```
$ git clone https://gitlab.cern.ch/GeantV/geant.git
$ mkdir build; cd build
$ cmake ../geant/ -DCMAKE_INSTALL_PREFIX=<...>  -DCTEST=ON
...
$ make -j10
...
$ ctest
Test project /build/geantv/build
    Start 1: TestEm3_GV
1/2 Test #1: TestEm3_GV .......................   Passed    8.55 sec
    Start 2: TestEm5_GV
2/2 Test #2: TestEm5_GV .......................   Passed    7.61 sec
```

Conclusions

- Have been tested on:
  - MacOS 10.13 (‘lcgcmake run’ need to be replaced with ‘source …’)
  - Ubuntu 18.04 (require a number of prerequisites; see DockerFile)
  - SLC 6 (gcc 6.2)
  - …
- Give it a try. Your feedback and complains will be appreciated!!

Additional Slides

Dockerfile for ubuntu18

```
$ FROM ubuntu:18.04
MAINTAINER Pere Mato <pere.mato@cern.ch>

ENV LCGCMAKE_ROOT=/usr/local/lcgcmake    \
    ARCH=x86_64

RUN apt-get update && \
    apt-get upgrade -y

# Tools necessary for installing and configuring Ubuntu
RUN apt-get install -y \
    python git cmake nano \
    g++ gcc gfortran binutils uuid-dev libssl-dev \
    libx11-dev libxpm-dev libxft-dev libxext-dev libmotif-dev \
    mesa-common-dev libglu1-mesa libxi-dev libxmu-dev libglu1-mesa-dev \
    libncurses5-dev zlib1g-dev libcurl4-openssl-dev libreadline6-dev

# Setup LCGCMake
RUN git clone https://gitlab.cern.ch/sft/lcgcmake.git $LCGCMAKE_ROOT

RUN useradd -m hsf
ENV USER=hsf
ENV HOME=/home/hsf
WORKDIR /build


RUN ln -s $LCGCMAKE_ROOT/bin/lcgcmake /usr/local/bin/lcgcmake
RUN chown hsf:hsf -R $LCGCMAKE_ROOT
RUN chown hsf:hsf /usr/local/bin/lcgcmake
RUN chmod +x /usr/local/bin/lcgcmake
RUN chown hsf:hsf /opt
RUN chown hsf -R /build

USER hsf
```