



---

# 1er Taller sobre Herramientas Computacionales para la Investigación en Protección de la Radiación.

Libro de Trabajo.

Por

Fernando Andrés Quiñonez Granados  
Doctor en Física de Partículas.

Junio, 2017.  
BUCARAMANGA - COLOMBIA.

# Índice general

<b>I</b>	<b>Herramientas Computacionales.</b>	<b>1</b>
<b>1</b>	<b>Introducción</b>	<b>2</b>
1.1	Instalación . . . . .	3
1.1.1	CLHEP 2.3.4.3 . . . . .	4
1.1.2	BOOST . . . . .	5
1.1.3	Vc 1.3.3 . . . . .	6
1.1.4	VecGeom 00.05.01 . . . . .	6
1.1.5	ROOT 5.34.36 . . . . .	7
1.1.6	GEANT4 10.03.p03 . . . . .	8
1.1.7	ECAT . . . . .	12
1.1.8	LMF . . . . .	13
1.1.9	FFTW . . . . .	13
1.1.10	ITK 4.11.0 . . . . .	14
1.1.11	RTK 1.2.0 . . . . .	20
1.1.12	GATE 8.0 . . . . .	20
1.1.13	FLUKA. . . . .	22
1.1.14	CORSIKA. . . . .	22
1.1.15	FORM. . . . .	24

# Índice de cuadros

# Índice de figuras

1.1	Configuration of ITK 1. . . . .	15
1.2	Configuration of ITK 2. Ojo con <i>python support</i> no funciona. . . . .	16
1.3	Configuration of ITK 3. . . . .	17
1.4	Configuration of ITK 4. . . . .	18
1.5	Configuration of ITK 5. . . . .	19
1.6	Configuration of RTK. . . . .	21

# Parte I

## Herramientas Computacionales.

# Capítulo 1

## Introducción

El propósito principal de este libro es dar a la comunidad científico-médica y a entusiastas de los temas en tecnología de Física Nuclear o de la Física de Partículas aplicados al estudio de la Medicina, una herramienta computacional llamada GATE [?], [?]. GATE fue creado por científicos de distintas nacionalidades, basado en otra herramienta computacional llamada GEANT4 [?] [?], desarrollada por científicos de la Organización Europea para la Investigación Nuclear CERN. GEANT4 ha servido como herramienta computacional para trabajar en temas relacionados con la simulación del paso de las partículas a través de la materia, habiendo obtenido una gran aceptación en la comunidad científica, de tal manera que ha dado lugar a una transferencia de tecnología, desde la Física de Partículas hacia los campos de las aplicaciones médicas y aplicaciones espaciales. Es en el campo de las aplicaciones médicas en donde GATE se ha especializado, siendo el software más exitoso. GATE ha ganado dos veces el premio del artículo más citado de la revista *Physics in Medicine and Biology* en 2009 [?] y 2015 [?]. Debido a que GATE es basado en GEANT4, todos los aspectos referentes a la geometría de los sistemas detectores, los procesos físicos involucrados en las interacciones de las partículas, y la generación de los eventos primarios, pueden ser leídos con mayor detalle desde el manual de GEANT4 presente en [?]. GEANT4 está escrito en C++ y los usuarios deben desarrollar sus proyectos mediante la programación en GEANT4/C++ para luego analizar los resultados de las simulaciones con ROOT [?]. Por su parte GATE fue escrito en GEANT4/C++ explotando las características del módulo *intercoms* de GEANT4 para que sus usuarios programaran en un nuevo tipo de lenguaje de programación de macros, los resultados de las simulaciones también se pueden analizar en ROOT o con algún otro software de análisis de datos que el usuario prefiera. Es así, que el objetivo principal de este libro es servir como una guía de usuario para programar los macros de GATE.

El único pre-requisito para empezar a leer este libro, es haber tenido familiaridad con algún sistema operativo tipo Linux, en particular

como instalar programas, como editar archivos, crear carpetas, borrar carpetas, nociones básicas de macros de bash, todo mediante el uso de la Terminal. En este capítulo veremos como instalar GATE, la forma de uso básico y las partes más importantes presentes en un código típico de GATE mediante el análisis del código Demo.

Copiar texto desde una Terminal: `Ctrl` + `Shift` + `C`

Pegar texto a una Terminal: `Ctrl` + `Shift` + `V`

## 1.1. Instalación

Las siguientes instrucciones han sido probadas en un sistema GNU/LINUX distribución **UBUNTU 16.04** de 64 bits. Se recomienda fuertemente no actualizar nunca después de instalado el sistema GNU/LINUX. Primero debemos instalar los pre-requisitos. En sistemas con base en DEBIAN el comando para instalar un libreria es

```
sudo apt-get install libreria
```

Para otras distribuciones de linux, el usuario debe cambiar el nombre de las librerías instaladas por sus respectivos equivalentes, por ejemplo, en las distribuciones con base en RED HAT usando

```
sudo yum install libreria
```

Se debe tener presente que cada vez que el usuario edite el archivo de configuración oculto `$HOME/.bashrc`, una de dos, o se cierran todas las terminales abiertas, o se ejecuta el comando `source`

```
source $HOME/.bashrc
```

sobre dicho archivo de configuración oculto en cada una de las terminales abiertas y las que se vayan a abrir. Esto se hace con el propósito de que BASH reconozca lo que se ha adicionado a este archivo. Primero debemos instalar los compiladores, los cuales son los pre-requisitos más importantes, gcc, g++ (y ¿por qué no? gfortran), todos en la versión 4.8.


```
sudo add-apt-repository ppa:ubuntu-toolchain-r/test
sudo apt-get update
```

```
sudo apt-get install gcc-4.8 g++-4.8 gfortran-4.8
sudo update-alternatives --remove-all gcc
sudo update-alternatives --remove-all g++
sudo update-alternatives --remove-all gfortran
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.8 20
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-4.8 20
sudo update-alternatives --install /usr/bin/gfortran gfortran /usr/bin/gfortran-4.8 20
sudo update-alternatives --config gcc
sudo update-alternatives --config g++
sudo update-alternatives --config gfortran
```

Luego si podemos instalar otros paquetes que serán necesarios para instalar otros programas más grandes o plataforma computacional que también son necesarias para finalmente poder tener el sistema GATE bien instalado.

```
sudo apt-get install \
unp \
git \
dpkg-dev \
make \
binutils \
libx11-dev \
libxpm-dev \
libxft-dev \
libxext-dev \
cmake \
cmake-curses-gui
```

### 1.1.1. CLHEP 2.3.4.3

Copiar y pegar en una terminal las siguientes ordenes, una línea a la vez, dando  al final de cada línea.

```
cd $HOME
```



```
mkdir CLHEP
cd CLHEP
wget http://proj-clhep.web.cern.ch/proj-clhep/DISTRIBUTION/tarFiles/clhep-2.3.4.3.tgz
unp clhep-2.3.4.3.tgz
mkdir 2.3.4.3-build 2.3.4.3-install
cd 2.3.4.3-build
cmake -DCMAKE_INSTALL_PREFIX=$HOME/CLHEP/2.3.4.3-install $HOME/CLHEP/2.3.4.3/CLHEP
cmake --build . --config RelWithDebInfo
ctest
cmake --build . --target install
```

Finalmente adicionar al archivo `$HOME/.bashrc` las siguientes líneas

```
export PATH=$PATH:$HOME/2.3.4.3-install/bin:$HOME/2.3.4.3-install/include
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/CLHEP/2.3.4.3-install/lib
```

### 1.1.2. BOOST

<http://www.boost.org/> El último release 1.65.1.

```
cd
mkdir BOOST
cd BOOST
tar xvfz boost_1_65_1.tar.gz
mv boost_1_65_1 1.65.1
mkdir 1.65.1-install
./bootstrap --prefix=/home/quinonez/BOOST/1.65.1-install
./b2
make -j8
```

luego imprime en pantalla

The Boost C++ Libraries were successfully built!

The following directory should be added to compiler include paths:

```
/home/quinonez/BOOST/1.65.1
```

The following directory should be added to linker library paths:

```
/home/quinonez/BOOST/1.65.1/stage/lib
```

### **1.1.3. Vc 1.3.3**

Descargar desde <https://github.com/VcDevel/Vc/releases>

```
unp Vc-1.3.3.tar.gz
cmake -DCMAKE_INSTALL_PREFIX=$HOME/VC/1.3.3-install $HOME/VC/1.3.3
make -j4
make install
```

Finalmente adicionar al archivo `$HOME/.bashrc` las siguientes líneas

```
export Vc_DIR=$HOME/VC/1.3.3-install
```

### **1.1.4. VecGeom 00.05.01**

Descargar la versión que se encuentra en <https://gitlab.cern.ch/VecGeom/VecGeom/tree/v00.05.01> y descomprimirla en el directorio vacío `$HOME/VECGEOM`

```
cd VECGEOM
unp VecGeom-v00.05.01-25b970bf1d72bda0b1b7946f336b38380f7f688c.tar.gz
mv VecGeom-v00.05.01-25b970bf1d72bda0b1b7946f336b38380f7f688 00.05.01
```

```
mkdir 00.05.01-buildinstall
cd 00.05.01-buildinstall
cmake -DBACKEND=Scalar -DGEANT4=OFF -DUSOLIDS=ON -DUSOLIDS_VECGEOM=ON \
-DVC_DIR=$HOME/VC/1.3.3-install \
-DCMAKE_INSTALL_PREFIX=`pwd` $HOME/VECGEOM/00.05.01
make install
```

poner en el `.bashrc` el siguiente export

```
export USolids_DIR=$HOME/VECGEOM/00.05.01-buildinstall/lib/cmake/USolids
```

### 1.1.5. ROOT 5.34.36

**Por amor a Dios.**

## 5.34.36

**no tener activado CONDA como variable de ambiente.**

Copiar y pegar en una terminal las siguientes instrucciones, una línea a la vez, dando  al final de cada línea.

```
cd $HOME
mkdir ROOT
cd ROOT
wget https://root.cern.ch/download/root_v5.34.36.source.tar.gz
tar xvvfz root_v5.34.36.source.tar.gz
mv root 5.34.36
cd 5.34.36
```

instalar también las librerías adicionales

```
sudo apt-get install \
gfortran \
libssl-dev \
```

```
libpcre3-dev \  
libglu1-mesa-dev \  
libglew-dev \  
libftgl-dev \  
libmysqlclient-dev \  
libfftw3-dev \  
graphviz-dev \  
libavahi-compat-libdnssd-dev \  
libldap2-dev \  
python-dev \  
libxml2-dev \  
libkrb5-dev \  
libgsl-dev \  
libqt4-dev
```

Copiar y pegar en una terminal las siguientes instrucciones, una línea a la vez, dando `Enter` al final de cada línea.

```
./configure linuxx8664gcc --all  
make -j4
```

Finalmente adicionar al archivo `$HOME/.bashrc` las siguientes líneas

```
source $HOME/ROOT/5.34.36/bin/thisroot.sh
```

### 1.1.6. GEANT4 10.03.p03

Copiar y pegar en una terminal las siguientes instrucciones, una línea a la vez, dando `Enter` al final de cada línea.

```
cd $HOME  
mkdir GEANT4  
cd GEANT4
```

Descargar el comprimido del proyecto fuente `geant4.10.03.p03.tar.gz` en la siguiente dirección

`https://github.com/Geant4/geant4/releases/tag/v10.3.3`

y guardarlo en `$HOME/GEANT4`.

Copiar y pegar en una terminal las siguientes instrucciones, una línea a la vez, dando `Enter` al final de cada línea.

```
tar xvfz geant4.10.03.p03.tar.gz
mv geant4.10.03.p03 10.03.p03
mkdir 10.03.p03-build 10.03.p03-install
```

instalar también las librerías adicionales

```
sudo apt-get install \
libxerces-c-dev \
libxerces-c-doc \
libxerces-c-samples \
libxerces-c3.1 \
gccxml \
libxmu-dev \
libmotif-dev
```

correr el comando

```
cd 10.03.p03-build
```

luego copiar y pegar en una terminal la siguiente orden dando `Enter` al final

```
cmake \
-DMAKE_INSTALL_PREFIX=$HOME/GEANT4/10.03.p03-install \
-DCLHEP_ROOT_DIR=$HOME/CLHEP/2.3.4.3-install \
-DGEANT4_USE_SYSTEM_CLHEP=ON \
-DGEANT4_USE_SYSTEM_EXPAT=ON \
-DGEANT4_USE_GDML=ON \
-DGEANT4_USE_QT=ON \
```

```
-DGEANT4_USE_OPENGL_X11=ON \
-DGEANT4_USE_RAYTRACER_X11=ON \
-DGEANT4_USE_SOLIDS="ALL" \
$HOME/GEANT4/10.03.p03
```

ahora se construye el proyecto mediante

```
make -j4
make install
```

Se crea el folder `$HOME/GEANT4/10.03.p03-install/share/Geant4-10.3.3/data` y vamos a él

```
mkdir -p $HOME/GEANT4/10.03.p02-install/share/Geant4-10.3.2/data
cd $HOME/GEANT4/10.03.p02-install/share/Geant4-10.3.2/data
```

Se deben descargar los datos comprimidos:

- `G4PhotonEvaporation4.3.2.tar.gz`  
 Datos en el formato Evaluated Nuclear Structure Data File (ENSDF) el cual es mantenido por [?].
- `G4RadioactiveDecay.5.1.1.tar.gz`  
 Datos en el formato Evaluated Nuclear Structure Data File (ENSDF) el cual es mantenido por [?].
- `G4NDL.4.5.tar.gz`  
 Datos que vienen de la librería *ENDF/B-VI*, los cuales son desarrollados y mantenidos por [?]. Los archivos originales puede ser descargados en [?].
- `G4TENDL.1.3.tar.gz`  
 Base de datos traducida de la librería de datos de protones incidentes TENDL-2014, esta librería posee los mismos isotopos que G4NDL, alrededor de 2600 [?], [?], [?], (escrita en formato ENDF-6). La librería completa TENDL-2014 puede ser descargada de la página de la IAEA [?], [?], [?]. El rango de energías de la partícula incidente en TENDL-2014 es desde 1 a 200 MeV.
- `G4SAIDDATA.1.1.tar.gz`  
 Datos evaluados desde la base de datos para protones, neutrones, piones inelásticos y elásticos, e intercambio de carga y secciones eficaces de nucleones con energía bajo 3 GeV [?].

- `G4NEUTRONXS.1.4.tar.gz`

Los datos evaluados son producidos usando el data set G4NDL mediante el procedimiento para neutrones con energías  $E < 10\text{MeV}$ . Para energías  $E > 20\text{MeV}$  la sección eficaz son calculadas por las clases de GEANT4 G4BGGNucleonInelasticXS y G4BGGNucleonElasticXS. Para el intervalo  $E \in [10, 20]\text{MeV}$  se usa una interpolación lineal.

- `G4EMLOW.6.50.tar.gz`

Los modelos de datos de Livermore para procesos electromagnéticos de baja energía. Todos han sido derivados de los modelos previos:

- EPDL97 - Evaluated Photon Data Library.
- EPDL97 - Evaluated Photon Data Library.
- EADL - Evaluated Atom Data Library.

Los modelos correspondientes a la extensión GEANT4-DNA se han obtenido desde [?], [?].

- `G4PII.1.3.tar.gz`

Gran trabajo realizado por [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?], [?].

- `RealSurface.1.0.tar.gz`

Datos provenientes de [?].

- `G4ENSDFSTATE.2.1.tar.gz`

Datos disponibles en [?].

- `G4ABLA.3.0.tar.gz`

desde la siguiente dirección

[http://geant4.web.cern.ch/geant4/support/datafiles\\_origin.shtml](http://geant4.web.cern.ch/geant4/support/datafiles_origin.shtml)


y guardarlos en el folder `$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data`, una vez hecho esto se procede a descomprimir estos archivos mediante la siguiente orden

```
for i in $(ls); do unp $i; done
```

Finalmente adicionar al archivo `$HOME/.bashrc` las siguientes líneas

```
source $HOME/GEANT4/10.03.p01-install/bin/geant4.sh
export G4NDL=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4NDL4.5
export G4EMLOW=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4EMLOW6.50
export PhotonEvaporation=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/PhotonEvaporation4.3
export RadioactiveDecay=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/RadioactiveDecay5.1.1
export G4SAIDDATA=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4SAIDDATA1.1
export G4NEUTRONXS=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4NEUTRONXS1.4
export G4ABLA=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4ABLA3.0
export G4PII=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4PII1.3
export RealSurface=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/RealSurface1.0
export G4ENSDFSTATE=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4ENSDFSTATE2.1
export G4TENDL=$HOME/GEANT4/10.03.p01-install/share/Geant4-10.3.1/data/G4TENDL1.3
```

### 1.1.7. ECAT

<http://opengatecollaboration.org/ECAT> Copiar y pegar en una terminal las siguientes ordenes, una línea a la vez, dando  al final de cada línea.

```
cd $HOME
mkdir ECAT
cd ECAT
```

Descargar el archivo comprimido `ecat.tar.gz` desde la dirección <http://opengatecollaboration.org/ECAT> y guardarlo en `$HOME/ECAT`, luego se siguen ejecutando las siguientes ordenes

```
tar xvfz ecat.tar.gz
cp Makefile.unix Makefile
cp utils/Makefile.unix utils/Makefile
```

Antes de proseguir se deben corregir un error en `Makefile` y en `utils/Makefile`, además de un error en el código fuente de ECAT, esto es, en ambos `Makefile` cambiar la línea 3 cambiar por `CC = gcc`, y en el código fuente cambiar la línea 186 del archivo



utils/show\_header.c por `if (mh->dosage>1000.0f)`. Además de esto se debe instalar la librería `happycoders-libsocket` y hacer una copia cambiando su nombre, para que pueda ser encontrada por ECAT.

```
make
cd utils
cp Makefile.unix Makefile
sudo apt-get install happycoders-libsocket
sudo cp /usr/lib/libsocket.so.0 /usr/lib/libsocket.so
make
cd ..
mkdir include lib
cp *.h include/
cp libecat.a lib/
```

### 1.1.8. LMF

```
sudo apt-get install libsfml-dev
cd
mkdir LMF
cd LMF
./configure
```

copias y pegar la salida de `configure` en `includes/lmf_format.h`, luego

```
make
cd examples
make
```

### 1.1.9. FFTW

Obtener `fftw-3.3.6-pl2.tar.gz` de la página <http://www.fftw.org/download.html>

```
cd
mkdir FFTW
cd FTW
unp fftw-3.3.6-pl2.tar.gz
mv fftw-3.3.6-pl2 3.3.6-pl2
make -j4
sudo make install
```

### 1.1.10. ITK 4.11.0

<https://itk.org/ITK/resources/software.html>

```
cd
mkdir ITK
cd ITK
sudo apt-get install bison valgrind
unp InsightToolkit-4.11.0.tar.gz
mv InsightToolkit-4.11.0 4.11.0
mkdir 4.11.0-build 4.11.0-install
cd 4.11.0-build
ccmake ../4.11.0
```

En la nueva terminal emulada, oprimir la tecla `c` para realizar la primera configuración. Luego de hacer una primera configuración, el archivo lucirá así Activar la opción `Module_ITKReview` ON para poder usar RTK. También activar las opciones

```
ITK_USE_FFTWD ON
ITK_USE_FFTWF ON
```

```
make -j4
make install
```

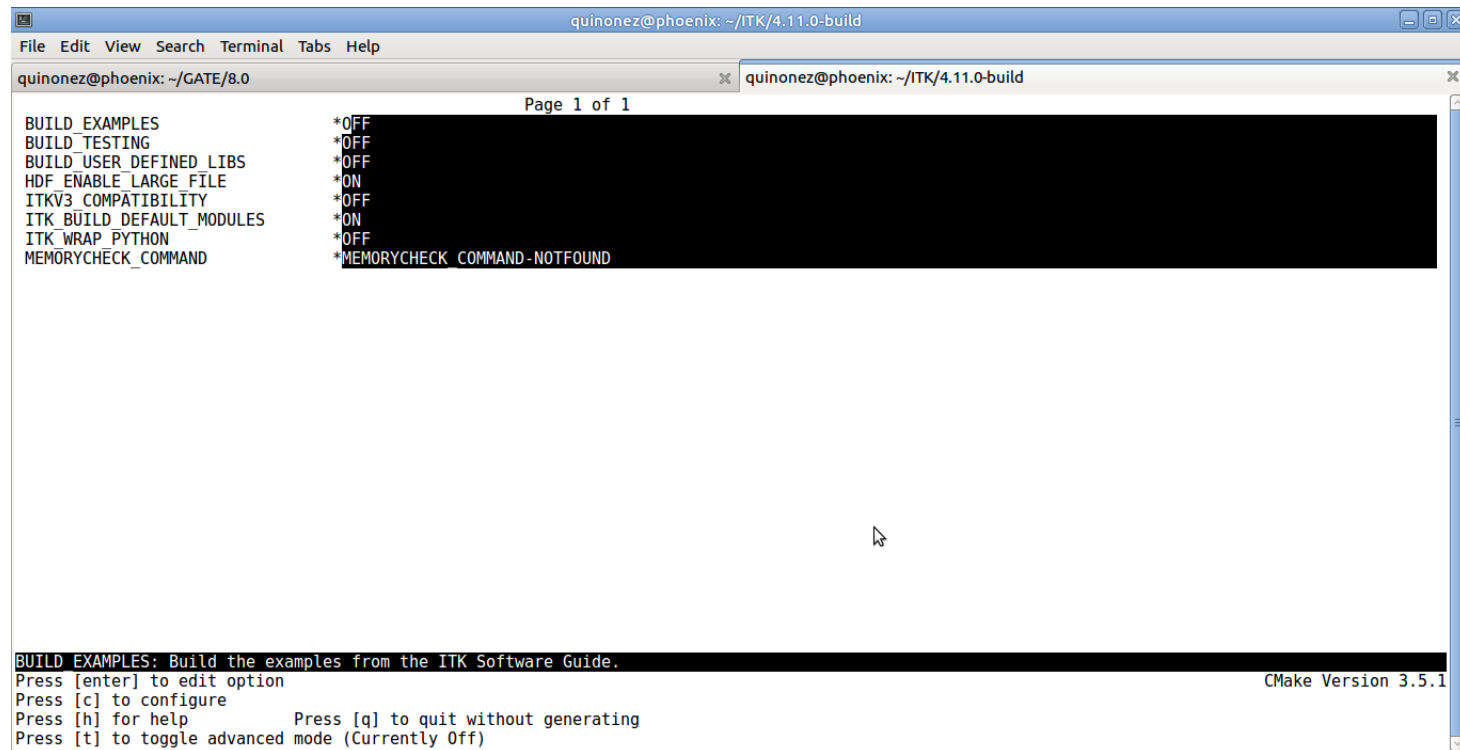
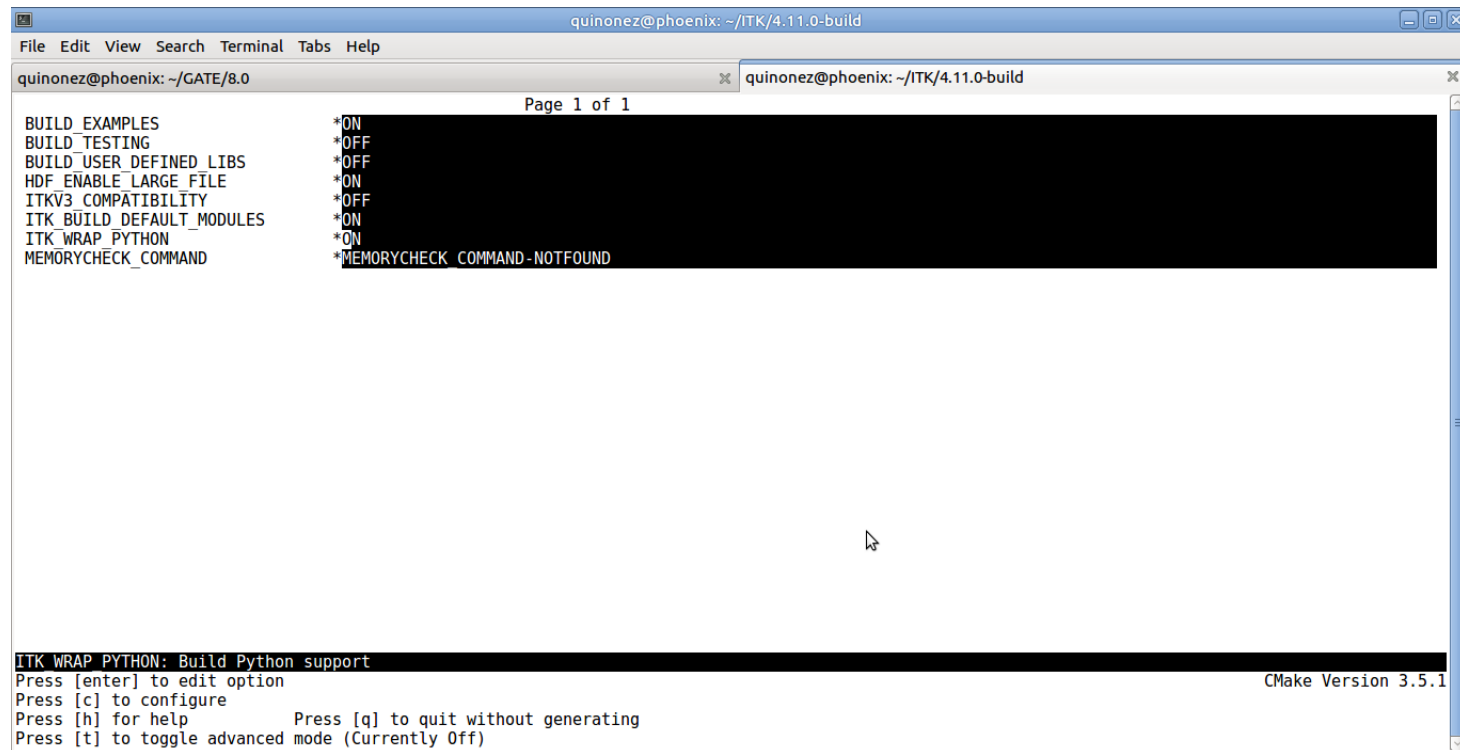


Figura 1.1: Configuration of ITK 1.

Figura 1.2: Configuration of ITK 2. Ojo con *python support* no funciona.

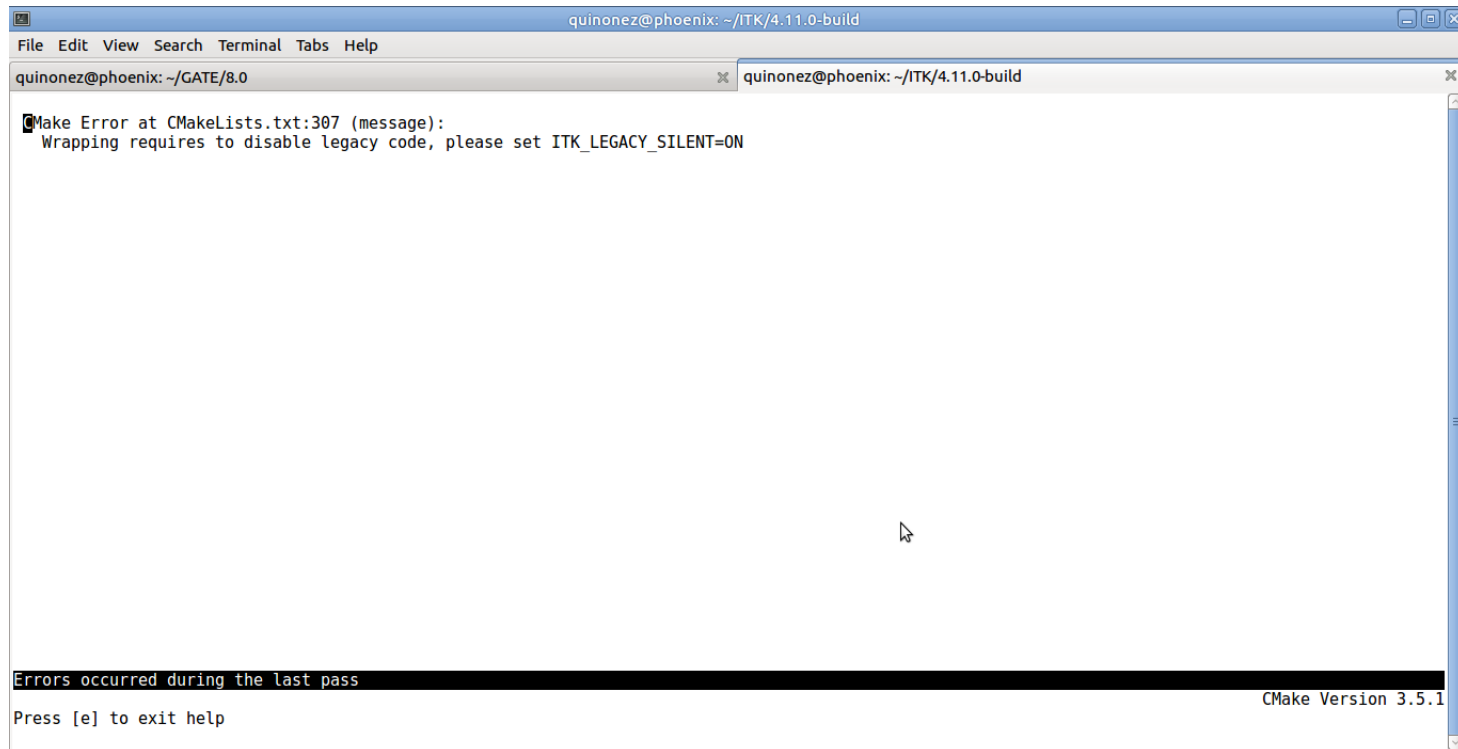


Figura 1.3: Configuration of ITK 3.

```

quinonez@phoenix: ~/ITK/4.11.0-build
File Edit View Search Terminal Tabs Help
quinonez@phoenix: ~/GATE/8.0  quinonez@phoenix: ~/ITK/4.11.0-build  quinonez@phoenix: ~/ITK/4.11.0-build
Page 2 of 10
CMAKE_C_FLAGS_MINSIZEREL -O5 -DNDEBUG
CMAKE_C_FLAGS_RELEASE -O3 -DNDEBUG
CMAKE_C_FLAGS_RELWITHDEBINFO -O2 -g -DNDEBUG
CMAKE_EXE_LINKER_FLAGS
CMAKE_EXE_LINKER_FLAGS_DEBUG
CMAKE_EXE_LINKER_FLAGS_MINSIZE
CMAKE_EXE_LINKER_FLAGS_RELEASE
CMAKE_EXE_LINKER_FLAGS_RELWITH
CMAKE_EXPORT_COMPILE_COMMANDS OFF
CMAKE_HP_PTHREADS OFF
CMAKE_INSTALL_PREFIX /home/quinonez/ITK/4.11.0-install
CMAKE_LINKER /usr/bin/ld
CMAKE_MAKE_PROGRAM /usr/bin/make
CMAKE_MODULE_LINKER_FLAGS
CMAKE_MODULE_LINKER_FLAGS_DEBU
CMAKE_MODULE_LINKER_FLAGS_MINS
CMAKE_MODULE_LINKER_FLAGS_RELE
CMAKE_MODULE_LINKER_FLAGS_RELW
CMAKE_NM /usr/bin/nm
CMAKE_OBJCOPY /usr/bin/objcopy
CMAKE_OBJDUMP /usr/bin/objdump
CMAKE_RANLIB /usr/bin/ranlib
CMAKE_SHARED_LINKER_FLAGS
CMAKE_SHARED_LINKER_FLAGS_DEBU
CMAKE_SHARED_LINKER_FLAGS_MINS
CMAKE_SHARED_LINKER_FLAGS_RELE
CMAKE_SHARED_LINKER_FLAGS_RELW
CMAKE_INSTALL_PREFIX: Install path prefix, prepended onto install directories.
Press [enter] to edit option
Press [c] to configure
Press [h] for help
Press [t] to toggle advanced mode (Currently On)
CMake Version 3.5.1

```

Figura 1.4: Configuration of ITK 4.

```

quinonez@phoenix: ~/ITK/4.11.0-build
File Edit View Search Terminal Tabs Help
quinonez@phoenix: ~/GATE/8.0  quinonez@phoenix: ~/ITK/4.11.0-build  quinonez@phoenix: ~/ITK/4.11.0-build
Page 5 of 13
ITK_BUILD_DEFAULT_MODULES      ON
ITK_COMPUTER_MEMORY_SIZE      1
ITK_CPPCHECK_TEST              OFF
ITK_DOXYGEN_CHM                OFF
ITK_DOXYGEN_DOCSET             OFF
ITK_DOXYGEN_ECLIPSEHELP        OFF
ITK_DOXYGEN_HTML               ON
ITK_DOXYGEN_LATEX              OFF
ITK_DOXYGEN_QHP                OFF
ITK_DOXYGEN_RTF                OFF
ITK_DOXYGEN_XML                OFF
ITK_DYNAMIC_LOADING            ON
ITK_FORBID_DOWNLOADS           OFF
ITK_LEGACY_REMOVE              OFF
ITK_LEGACY_SILENT              ON
ITK_TEMPLATE_VISIBILITY_DEFAULT OFF
ITK_TRANSFORM_FACTORY_MAX_DIM  4
ITK_USE_64BITS_IDS             OFF
ITK_USE_BRAINWEB_DATA          OFF
ITK_USE_CCACHE                 OFF
ITK_USE_CONCEPT_CHECKING      ON
ITK_USE_FFTWD                  OFF
ITK_USE_FFTWF                  OFF
ITK_USE_FLOAT_SPACE_PRECISION  OFF
ITK_USE_GIT_PROTOCOL           OFF
ITK_USE_GOLD_LINKER            ON
ITK_USE_GPU                    OFF
ITK_LEGACY_SILENT: Silence all legacy code messages.
Press [enter] to edit option
Press [c] to configure
Press [h] for help
Press [t] to toggle advanced mode (Currently 0n)
CMake Version 3.5.1

```

Figura 1.5: Configuration of ITK 5.

### 1.1.11. RTK 1.2.0

`http://www.openrtk.org/`


```
cd
mkdir RTK
cd RTK
```

Descargar la versión 1.2.0, colocarla en el folder RTK.

```
mv RTK-1.2.0 1.2.0
mkdir 1.2.0-build 1.2.0-install
cd 1.2.0-build
ccmake ../1.2.0
```

```
make -j4
make install
```

### 1.1.12. GATE 8.0

Copiar y pegar en una terminal las siguientes ordenes, una línea a la vez, dando  al final de cada línea.

```
cd $HOME
mkdir -p $HOME/GATE
cd $HOME/GATE
```

Obtener el archivo `gate_v8.0.tar.gz` desde la dirección `http://opengatecollaboration.org/GATE80` y guardarlo en `$HOME/GATE`, luego

```
cd $HOME/GATE
tar xvfz gate_v8.0.tar.gz
mv gate_v8.0 8.0
mkdir 8.0-build 8.0-install
cd 8.0-build
ccmake ../8.0
```



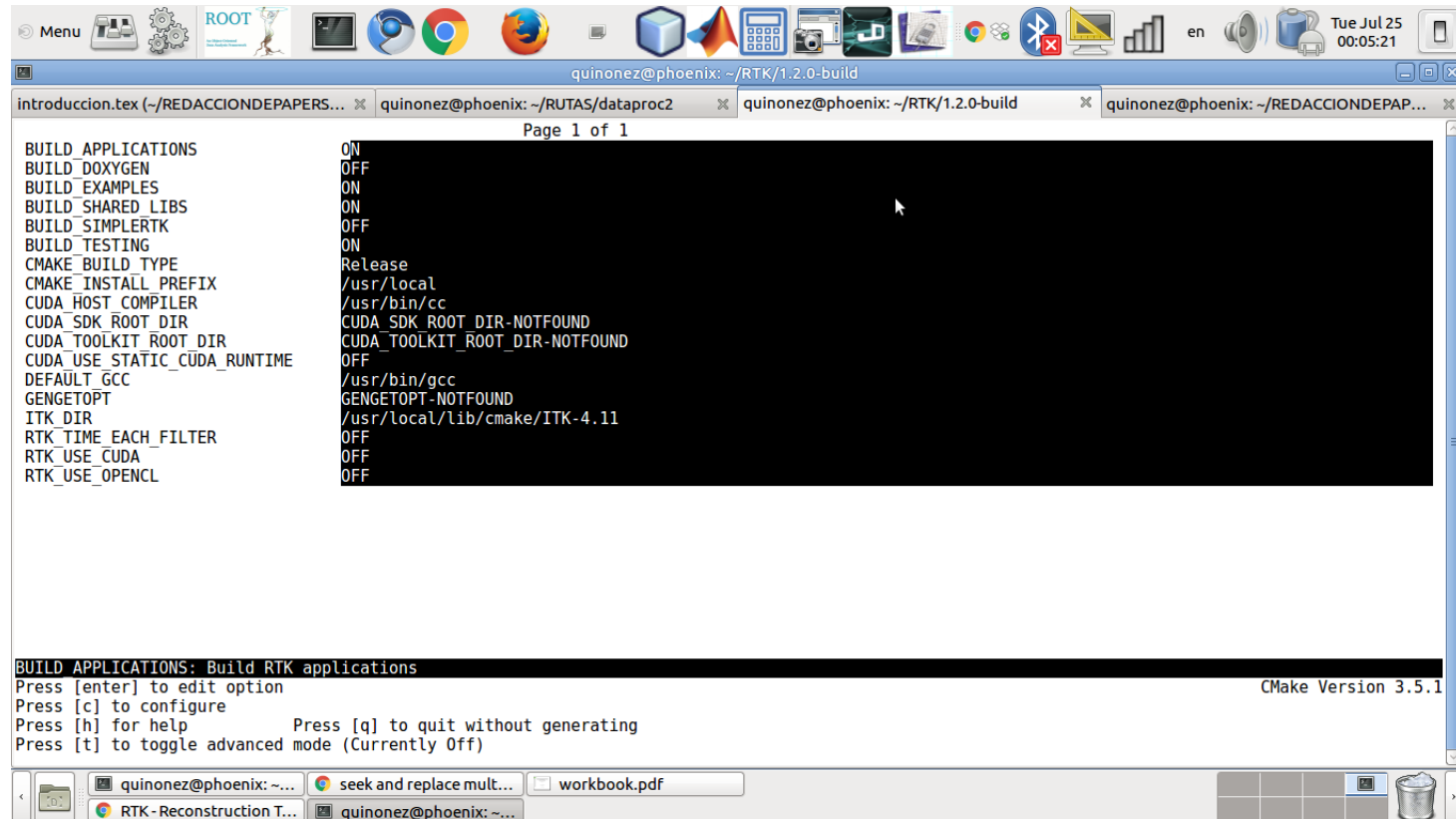


Figura 1.6: Configuration of RTK.

luego seguir las ordenes que se especifican en la figura ?? . Después construir el sistema mediante

```
make -j4
make install
```

### 1.1.13. FLUKA.

```
cd
mkdir FLUKA
unp
```

Editar el .bashrc agregando

```
export FLUPRO=${HOME}/FLUKA
export FLUFOR=gfortran
export PATH="${HOME}/FLUKA:$PATH"
export PATH="${HOME}/FLUKA/flutil:$PATH"
```

luego

```
cd $FLUPRO
make -j4
flutil/ldpmqmd
```

### 1.1.14. CORSIKA.

```
cd
mkdir CORSIKA
unp corsika-76400.tar.gz
mv corsika-76400 76400
cd 76400
./coconut
```

2

5

1

2

1a

1

1

1b

1

4a

5

6

6a

7

7a

7b

7c

9

9a

9b

d2

z

yes

k

make -j4

el binario de salida debe ser run/corsika76400Linux\_QGSII\_gheisha\_root

**1.1.15. FORM.**

Software para cálculo simbólico de uso en Física Teórica de Altas Energías. <https://www.nikhef.nl/~form/> <https://github.com/vermaseren/form>

```
cd
mkdir FORM
git clone https://github.com/vermaseren/form 4.2
cd 4.2
git checkout tags/v4.2.0
```

Luego se puede hacer

```
sudo apt-get install autoconf libgmp-dev zlib1g-dev
autoreconf -i
make -j4
sudo make install
```