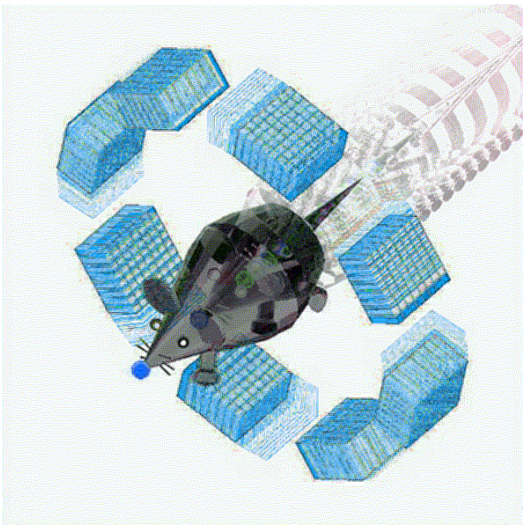
Geant4
Application for
Tomographic
Emission



Geant4.4.1 installation help guide

UNIVERSITE DE LAUSANNE

Apr. 29, 2003
Daniel Strul
UNIL – IPHE
Lausanne
Daniel.Strul@iphe.unil.ch

Contents

1 Introduction

I have summarized in this document some advices regarding the installation of *Geant4.4.1* for members of the OpenGate Collaboration. While I have tried to be as complete as possible in the description of this procedure, I have removed any information regarding the installing of non-essential packages (*Anaphe*, *GAG*…).

Note that *Geant4.4.1* is no longer the current release of Geant4 (which is *Geant4.5*). However, the compatibility of *Gate* with *Geant4.5* has not been fully insured yet. In the meantime, *Geant4.4* remains the reference environment for running GATE.

This installation was tested on workstations running Linux (*Red Hat 6.2+egcs 1.1* and *Red Hat 8.0+gcc3.2)*: while this guide is partly relevant for any Unix-based system, I don't know how much help it is for other platforms. Note that the installation of Geant4 itself does not require you to be the local system-administrator, but it demands a good knowledge of your system. Furthermore, you may at times need super-user (sys-admin) privileges for the installation of helper packages.

Thanks to all who helped me in the writing of this guide, and good luck!

2 Choosing the installation platform

The release notes for Geant4.4.1 provide the following guidelines:

```
Supported and Tested Platforms

  o SUN Solaris 5.7, C++ 5.2 Patch 109508-03.
  o Linux, gcc 2.95.2, egcs-2.91.66 (egcs 1.1.2).
    This configuration was tested with the RedHat 6.1 and 7.2 distributions,
    versions of Geant4 have also been compiled successfully on other Linux
    distributions, like Debian or Suse.
    The default RedHat compiler gcc-2.96 distributed in RedHat 7.X is NOT
    supported. It has been verified that it produces incorrect binaries,
    therefore is not reliable. See also note below.
  o Windows-2000 and CygWin Tools with: Visual C++ 6.0 Service Pack 5

Platforms configured but not tested and not supported:
  o DEC V4.0, cxx C++ V6.1-027 with/without ObjectSpace STL
  o HP 10.20, aCC C++ B3910B A.01.23 with/without ObjectSpace STL
  o SGI V6.5.5, CC 7.2.1 with ObjectSpace STL
  o AIX 4.3.2, xlC compiler with ObjectSpace STL
  o MacOS 10.1, gcc 2.92.2
```

I have experience with only two kinds of installation (*Red Hat 6.2+egcs 1.1* and *Red Hat 8.0+gcc3.2*), so it is difficult to make any meaningful comparison and to tell which one is best. However, here are some additional notes:

- Linux platforms are generally the easiest solution, as you have access to a lot of free source codes and precompiled packages.
- If you use *Red Hat 7.X*, you **MUST** change the compiler to *egcs* or *gcc 2.95*, as the default compiler (*gcc 2.96*) is buggy (it's a development version, which is not supported by GNU).
- At the time when I'm writing (Apr. 2003), the collaboration still is struggling with the installation of *GATE* under *gcc 3.X* (*Red Hat 8.0*…), and it is not clear whether this compiler can be safely used. To be continued…

3 Geant4.4.1 installation requirements

The basic elements needed for the installation and running of Geant4.4.1 are :

- A C++ compiler
- GNU Make
- The *CLHEP* library: a public library of classes designed for high-energy physics.

In addition to those elements, you also want to have some visualization capabilities in Geant4. To do so you must make sure that you have one additional package installed on your system: *OpenGL*. Most of these elements are probably already present on your system, so the only things you really have to worry about is *CLHEP* and *OpenGL*. Help for the installing of both is providing in the two next chapters.

4 Installing CLHEP

4.1 Downloading

The *CLHEP* library provides Geant4 with all the mathematical tools it needs, such as vectors, matrices, random functions, and so on. Documentation can be found at:

http://wwwinfo.cern.ch/asd/lhc++/clhep/index.html

There have been several releases of these library, the current one being *CLHEP-1.8.0.0*. Geant4.4.1 is meant to be used with an older release of this library: *CLHEP-1.7.5.0*. You can download it from:

http://wwwinfo.cern.ch/asd/lhc++/clhep/export/share/CLHEP/1.7.5.0/clhep-1.7.5.0.tgz

4.2 Installing

There are two ways of installing *CLHEP*:

- By default, *CLHEP* will try to install itself into */usr/local*. However, only the sys-admin can write into this directory: thus, if you're not sys-admin, the installation will fail.
- If you're not sys-admin, you must install *CLHEP* somewhere else, in a directory where you have write permissions. For instance, you could install it into your home directory. A cleaner solution is to install it into a directory dedicated to 'optional packages' (typically */opt* or */soft*) or into a directory dedicated to your work with Geant4 (e.g. */home/G4Project*).

In Lausanne, all G4-related packages are installed (as much as possible) in */opt*, so I will describe this option. Please have a look at the **note 1** at the end of this chapter before proceeding with this installation:

1. Copy the *CLHEP* archive *clhep-1.7.5.0.tgz* into your installation directory:
```
cp clhep-1.7.5.0.tgz /opt
```
2. Step into the target directory:
```
cd /opt
```
3. Uncompress the archive with *gtar* (on some systems, this command will be named *tar*):
```
tar xvzf clhep-1.7.5.0.tgz
```
4. You now have a subdirectory *CLHEP* within */opt*, step into it:
```
cd CLHEP
```
5. Launch the configuration command and specify the target directory for the installation of CLHEP (e.g. */opt/CLHEP*):
```
./configure –prefix=/opt/CLHEP
```
6. Compile and install with *gmake* (or *make*):
```
gmake install
```

Note 1:

We've seen above a regular installation of *CLHEP*. Its main default is that *CLHEP* is always installed in the same place (e.g. */opt/CLHEP*): thus, if you want to install a newer version of *CLHEP* (for installing a newer version of Geant4), you will have to overwrite your previous installation. I thus prefer the following variant:

1. Copy the *CLHEP* archive *clhep-1.7.5.0.tgz* into your installation directory:
```
cp clhep-1.7.5.0.tgz /opt
```
2. Step into the target directory:
```
cd /opt
```
3. Uncompress the archive with *gtar* (on some systems, this command will be named *tar*):
```
tar xvzf clhep-1.7.5.0.tgz
```
4. You now have a CLHEP directory, rename it:
```
mv CLHEP CLHEP-1.7.5.0
```
5. Create an alias:
```
ln –s CLHEP-1.7.5.0 CLHEP
```
6. Step into the *CLHEP* directory:
```
cd CLHEP
```
7. Launch the configuration:
```
./configure –prefix=/opt/CLHEP-1.7.5.0
```
8. Compile and install:
```
gmake install
```
9. (Optional) Remove the link:
```
cd ..
rm CLHEP
```

With this variant, you can have several versions of CLHEP (and thus several versions of Geant4) at the same time.

Note 2:

You now have a compiled *CLHEP* library in the subdirectory *lib* of the current directory (e.g. in */opt/CLHEP/lib*). Step into that subdirectory and check the name of the library: if it's something like *libCLHEP-g++.1.7.5.0.a*, it will make it difficult for other programs (such as Geant4) to find the library, because they are expecting a much simpler name (*libCLHEP.a*). In that case, create an alias with a command as below:

```
ln -s <current library name> libCLHEP.a.
```

Note 3:

After you've done the installation, you still have full write permissions for the *CLHEP* directory: I tend to remove all write permissions afterwards, to make sure than I don't erase *CLHEP* afterwards by mistake:

```
cd /opt
chmod –R –w CLHEP-1.7.5.0
```

5 Installing OpenGL

5.1 Downloading

Geant4 includes visualization capabilities through a range of graphic drivers. However, while the drivers are indeed included in the Geant4 package, the visualization programs (servers) are not. Consequently, you will be able to use the Geant4 visualization capabilities only if you have at least one visualization server installed on your system. While most of these drivers may be installed later on, it is worth having at least *OpenGL* as a default viewer. In the simplest case, you probably already have *OpenGL* installed on your system, and you can skip the following discussion.

On Unix/Linux systems, if you have no visualization program available, the easiest solution probably is to install *Mesa*, a shareware package that implements an OpenGL-compatible server and works fine with Geant4. Documentation and source codes can be found at:

http://www.mesa3d.org/

If you have a Linux workstation, you can check whether the packages needed for *Mesa* are already installed on your system using queries in the *rpm* database (*rpm* packages are ready-to-install groups of files for Linux systems):

For *Linux 6.2/7.X*:

```
rpm –q Mesa
rpm –q Mesa-devel
```

For *Linux 8.0*:

```
rpm –q XFree86-Mesa-libGL
rpm –q XFree86-Mesa-libGLU
rpm –q XFree86-devel
```

If that's not the case, you may install *Mesa* (but you need to be sys-admin to do so) by downloading these packages from:

http://rpmfind.net/

Once this is done:

1. Open a session as the sys-admin: `su –`
2. Install all packages: *Linux 6/7*:
```
rpm –Uhv mesa-devel
rpm –Uhv mesa
```
Linux 8:
```
rpm –Uhv XFree86-Mesa-libGL
rpm –Uhv XFree86-Mesa-libGLU
rpm –Uhv XFree86-devel
```
3. Exit from the sys-admin session: `exit`

Note 1:

Once you have installed *Mesa*, you should make a note of the directory where the libraries are: you might need later (for the compilation of your programs) to set an environment variable *OGLHOME* to point to this directory.

Note 2:

The Geant4 source files typically include the *OpenGL*/*Mesa* header files using the following instructions:

```
#include <GL/gl.h>
#include <GL/glx.h>
#include <GL/glu.h>
```

This means that the compiler will be looking for these header files in subdirectories of the directories listed in your include search-path, i.e. typically */usr/include/GL*, */usr/local/include/GL*, etc… If the Mesa header file directory has been installed somewhere else (*/usr/X11R6/include* for instance) and the compiler fails to find the Mesa header files, the compilation of the Geant4 OpenGL driver may fail.

You should thus make sure that the Mesa include directory is named *GL*, and that it grafted on one of the directories of your include search-path. If this condition is not fulfilled, a quick-fix (which requires super-user privileges) is to create a link from */usr/include* to your Mesa include directory using:

```
ln –s <Mesa include dir> /usr/include/GL.
```

6 Downloading Geant4.4.1

6.1 Geant4 sources

The components of Geant4.4.1, along with the installation guide and release notes, can be found at:

http://wwwinfo.cern.ch/asd/geant4/source/source_archive.html

It is possible to download pre-compiled libraries for some systems, but I've never done so, so I will essentially describe the other alternative (i.e. downloading the sources and compiling them on-site). In that case, the archive to download is: *geant4.4.1.p01.gtar.gz*.

Once you've downloaded the Geant4 archive, you should decompress it into your target directory:

1. Create a directory for all Geant4 installations and data:
```
mkdir /opt/geant4
```
2. Copy the Geant4 archive *geant4.4.1.p01.gtar.gz* into the installation directory:
```
cp geant4.4.1.p01.gtar.gz /opt/geant4
```
3. Step into the target directory:
```
cd /opt/geant4
```
4. Uncompress the archive with *gtar* (on some systems, this command will be named *tar*):
```
tar xvzf geant4.4.1.p01.gtar.gz
```
5. You now have a subdirectory *geant4.4.1.p01* within */opt/geant4*.

Note:

You should uncompress the geant4 source archive into a directory where you have enough free space. It is hard to say how much disk space you need exactly, as it depends on many factors: what you install, which compilation options you choose, which data packages you add on, where you place your temporary directory and so on. On one of our installations, Geant4 takes about 320 Mb, including 60 Mb of libraries, 190 Mb of data files, and 70 Mb of temporary files. We also have a much bigger installation (the Geant4 libraries have been compiled with the debug option), with 450 Mb of libraries and 450 Mb of temporary files, leading to a total (when one includes the data files) of about 1.1 Gb.

Once you have downloaded and uncompressed the Geant4 source archive, the Geant4 directory should include the following elements:

```
Configure
ReleaseNotes/
config/
environments/
examples/
source/
```

6.2 Geant4 data files

Four additional data files may be downloaded: *G4NDL3.5*, *G4PhotonEvap1.0*, *G4RadiativeDecay2.0*, *G4EMLow1.1*. I am not sure whether these packages are optional or whether all of them are needed. In practice, I now download 3 packages out of 4, i.e. all the packages except *G4PhotonEvap1.0* (the photon evaporation data are provided by *G4RadiativeDecay2.0*).

You can download these data packages anywhere on your hard-disk. Yet, I find it easier to create a subdirectory *data* in the Geant4 directory (*/opt/geant4*), and to transfer all the data packages there. If you do so, the content of the subdirectory *data* will look as below:

```
G4EMLOW1.1.tar.gz
G4NDL3.5.tar.gz
G4RadiativeDecay.2.0.tar.gz
```

Then, you must decompress these 3 archives:

```
tar xvzf G4EMLOW1.1.tar.gz
tar xvzf G4NDL3.5.tar.gz
tar xvzf G4RadiativeDecay.2.0.tar.gz
```

You will thus get three data subdirectories (you may then destroy the compressed archives):

```
G4EMLOW1.1/
G4NDL3.5/
G4RadiativeDecay.2.0/
```

7 Installing Geant4.4.1

7.1 Configuration

Once all packages have been downloaded and uncompressed, you are ready to install Geant4. The best solution is to use the automated installation procedure, as described in section 2.2 of the installation guide (beware, that's actually the guide for Geant4.5.0):

http://wwwinfo.cern.ch/asd/geant4/G4UsersDocuments/UsersGuides/InstallationGuide/html/UnixMachines/unixMachines.html

To use this procedure, step into the base Geant4 directory. Then (assuming that you are using a C-shell such as *csh* or *tcsh)*, launch:

```
./Configure –install
```

From then on, you will be guided throughout the installation by the installer. However, I provide below some more details on a few tricky points: for each point, I first write a copy of the installer's question, and then some comments on your options.

OS and compiler

```
Definition of G4SYSTEM variable is Linux-egcs.
That stands for:

1) OS         : Linux

2) Compiler   : egcs

To modify default settings, select number above (e.g. 2)
[Press [Enter] for default settings]
```

The installation program automatically detects the optimal parameters for your platform. You can modify these parameters if needed, but don't do it if you can avoid it.

Geant4 installation place

```
Where is Geant4 installed?  [/users/dstrul/geant4]
```

The text between brackets is the installation location for G4 proposed by the installation script. As you may see on the example above, the installation script always proposes to install itself into your home directory. As I want G4 to be installed for all users and not only me, I generally modify this option to point to the real Geant4 directory (e.g */opt/geant4/geant4.4.1.p01*).

Header copying

```
Do you want to copy all Geant4 headers
in one directory? [n]
```

The default answer is "n", but I find it much more convenient to change this option to "y": all Geant4 header files will be copied into one directory, so I'll be able to find the class declarations easily.

Target directories

```
Where will be directory to copy all Geant4 headers?
[/opt/geant4/geant4.4.1.p01/include]


(OPTIONAL) You can now customise installation directories:
  G4TMP - tmp/ containing temporary dependency (.d) and object (.o) files;
  G4LIB - lib/ containing final static (.a) or shared (.so) libraries;

1) G4TMP:         /opt/geant4/geant4.4.1.p01/tmp

2) G4LIB:         /opt/geant4/geant4.4.1.p01/lib

To modify default settings, select number above (e.g. 2)
[Press [Enter] for default settings]
```

You may change these options if you have a problem of disk space on the G4 disk. If not, just keep the automatic settings.

Data directories

```
Please, specify directories where the Geant4 data is installed:

1) G4LEVELGAMMADATA:          /opt/geant4/geant4.4.1.p01/data/PhotonEvaporation

2) G4RADIOACTIVEDATA:         /opt/geant4/geant4.4.1.p01/data/RadiativeDecay

Data which is not part of the default Geant4 distribution:

3) G4LEDATA:                  /opt/geant4/geant4.4.1.p01/../G4EMLOW1.1

4) NeutronHPCrossSections:    /opt/geant4/geant4.4.1.p01/../G4NDL3.5

To modify default settings, select number above (e.g. 2)
[Press [Enter] for default settings]
```

The default options for these data directories are often incorrect, as is the case in the example above. You must change each of these locations, one after the other, to make sure that they point to the proper directory for each set of data files. If you have installed all data sets in a subdirectory *data* of the Geant4 directory, you must modify these settings as in the example below:

```
Please, specify directories where the Geant4 data is installed:

1) G4LEVELGAMMADATA:
      /opt/geant4/data/G4RadiativeDecay.2.0/data/PhotonEvaporation

2) G4RADIOACTIVEDATA:
      /opt/geant4/data/G4RadiativeDecay.2.0/data/RadiativeDecay

Data which is not part of the default Geant4 distribution:
```

```
3) G4LEDATA:                 /opt/geant4/data/G4EMLOW1.1

4) NeutronHPCrossSections:   /opt/geant4/data/G4NDL3.5
```

The 4 data directory paths (especially those for the low energy and neutron data) probably need to be updated to agree with the exact paths of your data packages.

CLHEP base directory

```
Please, specify where CLHEP is installed:

CLHEP_BASE_DIR:              /usr/local

According to it will be set:

CLHEP_INCLUDE_DIR
CLHEP_LIB_DIR
CLHEP_LIB

You will be asked about customizing these next.
[/usr/local]
```

The installation script always assumes that CLHEP is installed in */usr/local/CLHEP*. If that's not the case, you must change this setting to tell where CLHEP is installed (e.g. in */opt/CLHEP-1.7.5.0*).

CLHEP components

```
You can customize paths of you CLHEP installation:

1) CLHEP_INCLUDE_DIR:        /opt/CLHEP-1.7.5.0/include
2) CLHEP_LIB_DIR:            /opt/CLHEP-1.7.5.0/lib
3) CLHEP_LIB:                CLHEP

To modify default settings, select number above (e.g. 2)
[Press [Enter] for default settings]
```

Provided that you made sure that the *CLHEP* installation directory is right, the *CLHEP* include directory (setting 1) and library directory (setting 2) should be right as well. However, you must make sure that the name of the CLHEP library file (setting 3) is right too. To do so:

1. Open a new terminal
2. Step into the directory pointed at by *CLHEP_LIB_DIR*, e.g.:
   ```
   cd /opt/CLHEP-1.7.5.0/lib
   ```
3. Check whether there is a CLHEP libray:
   ```
   ls libCLHEP.*
   ```
4. If that's the case, you're fine, you can proceed with the installation

5. If that's not the case, it's likely that the library exists under another name. You must find this library:
```
ls libCLHEP*
```
6. Once you have found the real name of the *CLHEP* library, you must write it down without its extension and without the prefix "*lib*": that's the name you need for *CLHEP_LIB*. For instance, if you find a library which name is *libCLHEP-1.7.5.0-g++.a*, then the name you need for *CLHEP_LIB* is: *CLHEP-1.7.5.0-g++*

DEBUG mode

```
Do you want to compile libraries in DEBUG mode (-g)? [n]
```

By default, Geant4 is compiled in optimized mode. This means that your simulations will run as fast as possible, but you won't be able to follow with the debugger what's going on inside the Geant4 libraries. I tend to do two installations: first without the debugging option, then with it (changing the target *tmp* and *lib* directories from one installation to the other). The debug-enabled libraries are much larger and slower than the non-debug ones: for a start, just keep the default option.

Library type

```
By default 'static' (.a) libraries are built.

Do you want to build 'shared' (.so) libraries?
[n]
```

There are two opinions about this one:

- Some people prefer to build only static libraries, so the inclusion of the G4 libraries into *GATE* is done at link time. The *GATE* program becomes bigger, but once it's compiled, it's independent from the Geant4 libraries. Also, I found it easier to use the debugger with this option.
- Other people prefer to build the shared libraries: if they're built, they'll be used rather than the static ones, and thus they will be included in *GATE* only at run-time. Thus, the *GATE* program becomes much thinner, and is updated whenever you change the Geant4 libraries.

Feel free to pick your own choice, as both work fine!

User interface and visualization drivers

After you've selected the type of library you want, you are presented with a lot of options as to which user interface and visualization drivers you want to build. That's an issue where you should be rather cautious. Geant4 comes with a range of options for various user interfaces (XAW, Motif, Win32…) and visualization tools (Open GL, Motif, Dawn…). It is nice to enable as many as possible of these options, but most of them require external libraries, and your Geant4 installation may fail if these libraries are not already on your system. The Geant4 installer clearly states what is required for each option: check it out before enabling any 'exotic' option. I normally enable *OpenGLX* as OpenGL/Mesa is installed on my system. I also enable *DAWN* and *VRML*: I think these drivers don't need any external libraries and can be installed anywhere. Note that these drivers can be individually compiled later, so it is worth playing it safe for now.

Analysis option

```
  G4ANALYSIS_USE

Activates the configuration setup for allowing plugins to
analysis tools based on AIDA (Astract Interfaces for Data
Analysis). In order to use AIDA features and compliant analysis
tools, the proper environment for these tools will have to be
set (see documentation for the specific analysis tools).
[n]
```

While the default answer is “n”, you need this option set to have access to some GATE output and analysis features: set the option to “y”.

7.2 Compilation

Once the configuration is completed, the compilation starts. It normally takes a good while, especially if you did choose the optimized mode. Be patient!

If you have any problem at this stage, you can try having a look at the Geant4 User forum

http://geant4-hn.slac.stanford.edu:5090/Geant4-HyperNews/index

or at the Geant4 problem database:

http://wwwinfo.cern.ch/asdcgi/geant4/problemreport/query.cgi

If everything works fine, all Geant4 libraries have been created in the subdirectory *lib* of the Geant4 directory.

7.3 Creation of the Geant4 configurations scripts

Each time you want to compile or run a program based on Geant4 (GATE, examples…), you need to set a number of environment variables. To make sure that these variables are properly set each and every time, the best way is to ask Geant4 to create configuration scripts. To do so, from the Geant4 directory, launch (assuming that you are using a C-shell such as *csh* or *tcsh*):

```
./Configure
```

This command will create the configuration scripts *env.csh* and *env.sh* appropriate for your installation.

7.4 Write-protection

After you’ve done the installation, you still have full write permissions for the *Geant4* directory: I tend to remove all write permissions afterwards, to make sure than I don’t erase *Geant4* afterwards by mistake:

```
cd /opt
chmod –R –w geant4.4.1.p01
```

8 Compiling and running the novice example N01

8.1 Preparation

Once Geant4 is compiled and your configuration scripts have been generated, you are nearly ready to work with Geant4. However, there are a few things that may be worth doing first:

1. I would advise choosing or creating a work directory different from the Geant4 directory itself. The objective is to keep the Geant4 package directory safely separated from your applications and data. The default option of the Geant4 configuration scripts is to use your home directory as the working directory: this is by far the safest option, since it will prevent running into read/write permissions problems.
2. You should make sure that you source the configuration scripts *env.csh* or *env.sh* before each Geant4 work session (i.e. type the command *source /opt/geant4.4.1.p1/env.csh* for instance).
3. Note that the default option of the Geant4 configuration scripts is to use your home directory as your working directory. It is often more convenient to change the setting for the G4WORKDIR environment variable with:

```
setenv G4WORKDIR "."
```

8.2 Compiling and running the example novice/N01

You are now ready to compile your first example. To do so:

1. Step into the working directory you have chosen

```
cd <working_directory>
```

2. Copy the novice example directory from Geant4 into your working directory:

```
cp –r /opt/geant4/geant4.4.1.p01/examples/novice/N01
.
```

3. Step into the local copy of the novice example directory you've just created:

```
cd N01
```

4. Set your environment variables by sourcing the script *env.csh* or *Geant4*:

```
source /opt/geant4.4.1.p01/env.csh
```

   If you had changed G4WORKDIR before, you should see now the change confirmed in the last line:

```
In your environment you have the G4WORKDIR=.
```

5. Launch the compilation:

```
gmake
```

6. It should have created an executable program *exampleN01* in the subdirectory *bin/<system>* of your working directory. Launch it with:

```
bin/<system>/exampleN01
```

9 Compiling and running the novice example N02

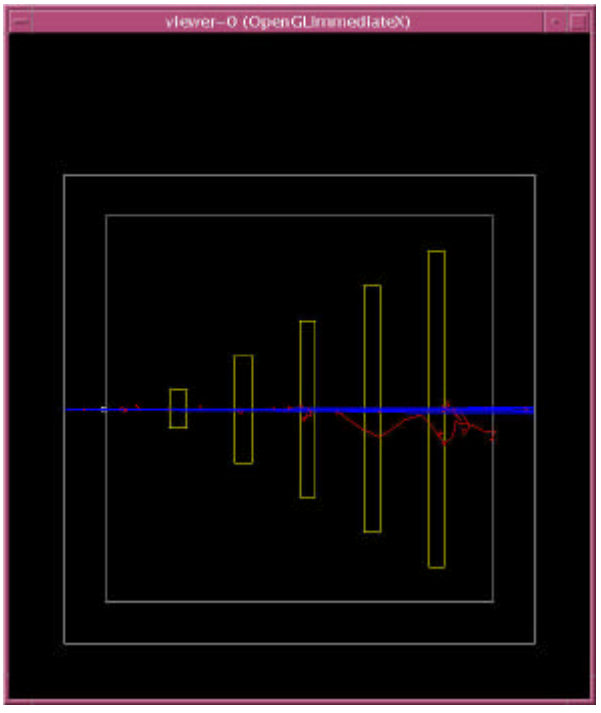
The main difference between this example and the example *N01* is that it uses a graphic output. Thus, this is the time when you find out whether your visualization drivers and servers have been properly installed…

Hopefully, it should work fine, but there is a rather wide range of problems that may appear at this stage, because you are now dependant on external packages such as *OpenGL*.

- The compilation or run may fail if the linker can not find the OpenGL/Mesa libraries (*libGL*, *libGLU*). You can try to include the missing path to your libraries using the command `setenv OGLHOME <path>` (before compiling).
- The program may launch fine, but fail to open a visual display. In that case, you should check that *libG4OpenGL* has been properly built when Geant4 was installed, and that the variable *G4VIS_USE_OPENGLX* has been properly set to 1 (one) before compiling the example.

More details on this issue, and on the proper settings of the environment variables for graphical output can be found in *README* in *$G4INSTALL/source/visualization*.

Note that this example requires several macro-command files (with the extension *.mac*). Thus, once you have successfully compiled and linked the program, you should copy these files from the example source directory (*$G4INSTALL/examples/novice/N02*) into the example bin directory (*$G4WORKDIR/bin/$G4SYSTEM*). You should copy the example *README* file as well, since this file tells you how to use the program *exampleN02*. If this example works properly, you should have a graphic output similar to the figure below.



Once you have managed to compile this example N02, most of the other novice examples should pose no major problem. Note that this example is perfect to test your other graphical drivers if you have any: simply edit the line */vis/open OGLIX* of the macro-file *vis.mac* to try another driver (*DAWN*, *DAWNFILE*, *VRML2FILE*, …).

10 Installing additional visualization drivers

10.1 The Geant4 visualization drivers

Geant4 includes visualization capabilities through a range of graphic drivers. I do not know the exact characteristics of each driver. Currently, I have only installed the three drivers below:

- *OpenGL*: It is the default viewer. Widely available, simple to use, but its possibilities are rather limited.
- *Dawn*: It is a 3D plotter/renderer. In interactive mode, it can be used to draw the same "scene" under different angles, light conditions, etc… It can generate very nice full-3D postscript outputs. It is interactive, but unfortunately not in real-time (i.e. you set the parameters, then you launch a drawing, then you modify the parameters and launch a new drawing, etc…)
- *VRML2FILE*: It is the most interactive solution for display: the simulation "scene" is stored into a VRML file, which can be automatically displayed (and observed along any angle) using a VRML viewer.

A potential problem with the Geant4 visualization modules is that the relevant informations are scattered throughout various documents. I have tried to summarize below a list of resources that may be of interest.

A good starting point obviously is the user's guide chapter on visualization:
http://geant4.web.cern.ch/geant4/G4UsersDocuments/UsersGuides/ForApplicationDeveloper/html/Visualization/index.html

and more specifically the section on drivers:
http://geant4.web.cern.ch/geant4/G4UsersDocuments/UsersGuides/ForApplicationDeveloper/html/Visualization/visdrivers.html

Practical installation guidance may also be found in the numerous *README* files within the subdirectory *source/visualization* of the Geant4 package.

In addition, the best resource for all visualization-related problems probably is the GEANT4 Fukui University Group Home Page at:
http://geant4.kek.jp/~tanaka/

10.2 Installing Dawn

The installation of *Dawn* proved easy enough. I simply downloaded the compressed file from the Dawn homepage and followed the instructions:

http://geant4.kek.jp/~tanaka/DAWN/About_DAWN.html

Once you have installed *Dawn*, the simplest way to test it is to use it for the visualization of the example novice/N02. To do so:

1) In the macro-file *vis.mac*, modify the line

```
/vis/open OGLIX
```

into

```
/vis/open DAWN
```

2) Set the environment variable *G4DAWN_HOST_NAME* as the name of the station where you're working. For instance, if you're working on the station *geant4.unil.ch*:

```
setenv G4DAWN_HOST_NAME geant4.unil.ch
```

3) Check that you have a PostScript viewer installed on your system. By default, Dawn assumes that you are using the program *gv* (ghostview). If you have a different viewer installed on your system, you should set its name using:

```
setenv DAWN_PS_PREVIEWER <name>
```

4) Launch the Dawn inet daemon *dawninetd* (this program normally has been compiled and installed when you installed Dawn).

5) Launch the simulation program *exampleN02*. If everything went OK, you should have at this stage the default output of the Dawn viewer, i.e. a PostScript drawing seen with a PostScript viewer such as GhostView.

There are several environment variables that may be used to customize the way the Dawn server works (they are described in the file *DAWN_ENV.html* in the subdirectory *DOC/HTML* of Dawn). The best combination I have found is as follows:

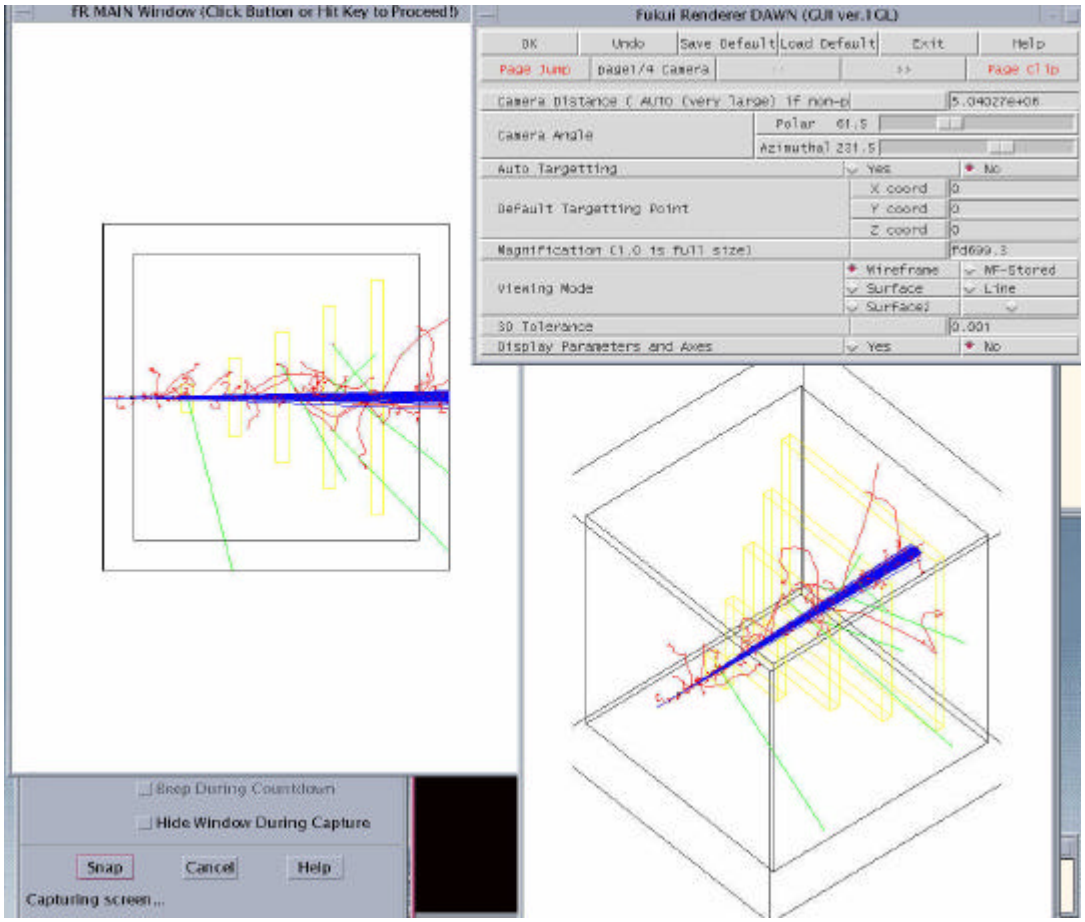
- Use Dawn in its local rather than its distant mode, so that you do not need to launch the daemon *dawninted* every time:

```
setenv G4DAWN_NAMED_PIPE       "1"
```

- Force the use of the Dawn interactive graphical user interface (GUI):

```
setenv G4DAWN_MULTI_WINDOW          "1"
```

With these settings, Dawn now provides a GUI, which allows you to fine-tune your display parameters. A typical output in this mode would look as the figure below:



10.3 Installing a VRMLfile viewer

For this visualization driver, I followed the link proposed in the Fukui University Group Home Page:

http://www.sim.no/downloads.html

This site proposed a small pre-compiled viewer called VRMLView pro.

To use and test this program on the example *N02*, you only have to do the following:

1) Install the executable *vrmlview* into one of your bin directories (and launch *rehash* if needed).

2) In the macro-file *vis.mac* of *exampleN02*, modify the line

```
/vis/open OGLIX
```

into

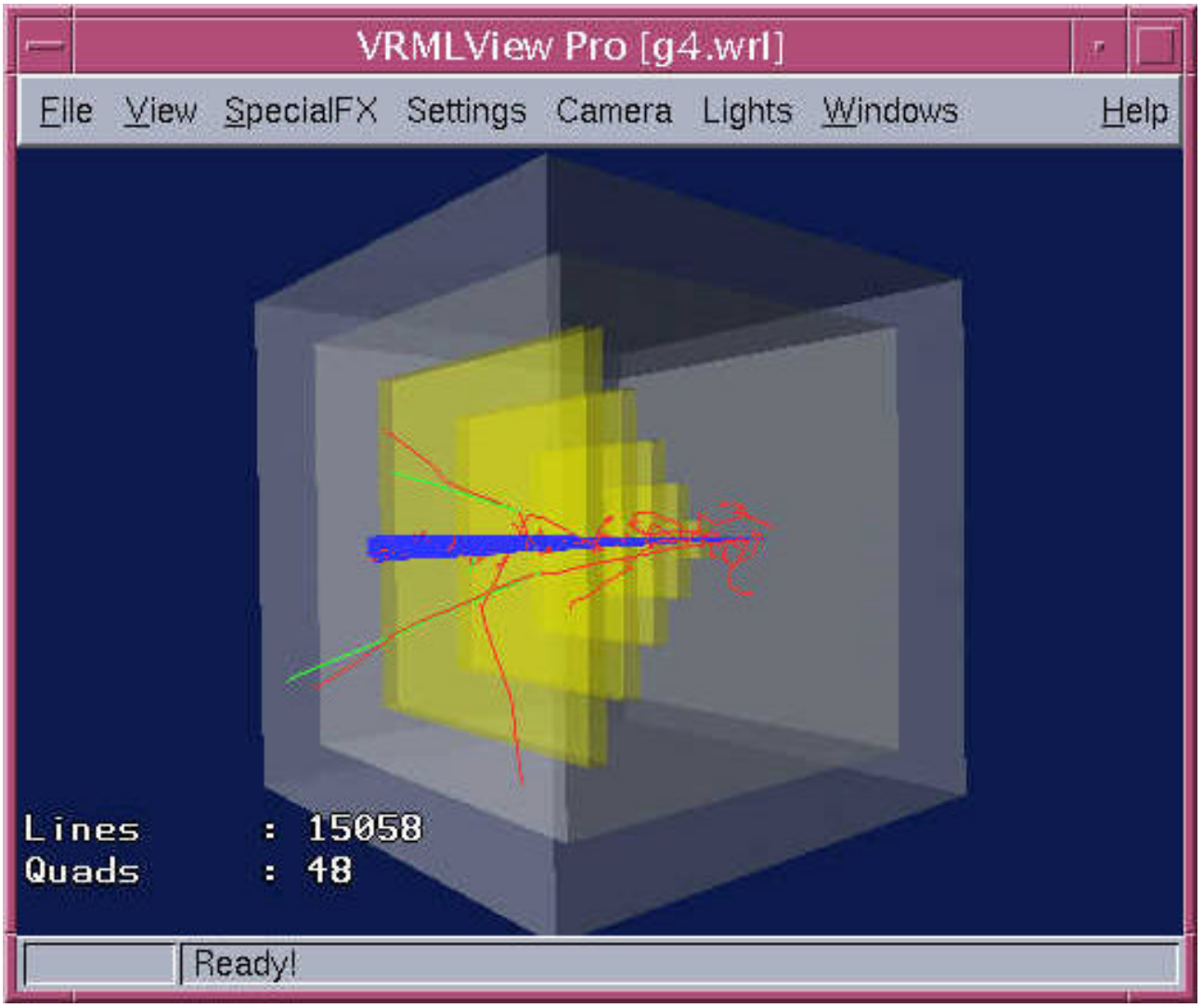
```
/vis/open VRML2FILE
```

2) Set the environment variable G4VRMLFILE_VIEWER:

```
setenv G4VRMLFILE_VIEWER vrmlview
```

4) Launch the simulation program *exampleN02*.

If everything went OK, you should obtain a VRML view of your scene, similar to the example below:



Note:

A member of our team has run into a problem when trying to use the VRMLFILE output. I report this problem below, since it's a good example of how things can sometimes go unexpectedly go wrong.

At first, the problem was that Geant4 couldn't find *vrmlview*, and kept complaining with:

```
sh: vrmlview: command not found
```

That was no big problem, as it only meant that the program *vrmlview* was not installed on that system. All we had to do was download *vrmlview* and install it somewhere in the path… or so we thought!

In fact, *vrmlview* did not launch, because it kept complaining that *libGL.so.1* was missing. That was extremely confusing: *libGL.so* and *libGLU.so* are two libraries of *OpenGL*; the *OpenGL* graphical output was working fine; so these libraries had to be somewhere!

It took us a while, but we eventually found out where the problem laid. In fact, the *OpenGL* features were provided by a *Mesa* library that had been compiled on site rather than installed from an *rpm* package. This installation showed the following problems:

- The libraries were not located in their usual directory;
- They were called *libMesaGL* and *libMesaGLU* instead of *libGL* and *libGLU*;
- And the worse problem of all: only the static libraries (*libMesaGL.a* and *libMesaGLU.a*) had been compiled, while the dynamic libraries (*libMesaGL. so* and *libMesaGLU. so*) had not.

At this stage, the only solution was to complete the installation of *Mesa* so as to provide the missing dynamic libraries. Once that was done, *vrmlview* and *OpenGL* lived happily together ever after…

11 Installing ROOT

11.1 Introduction

ROOT is a free data analysis and histogramming package developed at CERN. While you don't need it to install Geant4, you will need it to have access to the *GATE*'s real time plotter and off-line ROOT-based analysis. You can find more information about ROOT on their home-page:

http://root.cern.ch/

11.2 Downloading ROOT

You can download ROOT from:

http://root.cern.ch/root/Availability.html

You should also download the installation guidelines from:

http://root.cern.ch/root/Install.html

As you will see, several versions are available. If you are using *Linux 6.2*, you may download any of them, the recommended versions being either the version *Pro* or the version *New*. With Linux 8.0, you must download the version *New* (i.e. a version equal to or larger than 3.05/03), because older versions of *ROOT* are not compatible with *RH8.0*.

Once you've downloaded the Geant4 archive, you should decompress it into your target directory:

1. Create a directory for all *ROOT* installations:
```
mkdir /opt/root
```
2. Copy the *ROOT* archive into the installation directory:
```
cp root_v3.05.03.source.tar.gz /opt/root
```
3. Step into the target directory:
```
cd /opt/root
```
4. Uncompress the archive with *gtar* (on some systems, this command will be named *tar*):
```
tar xvzf root_v3.05.03.source.tar.gz
```
5. You now have a subdirectory *root* within */opt/root*. Rename it so that you can install newer versions of *ROOT* afterwards
```
mv root root3.05.03
```

11.3 Compiling ROOT

As you will see in the *ROOT* installation guidelines, there are only a few steps needed to compile ROOT:

1. Step into the ROOT directory:
```
cd root3.05.03
```
2. Define the installation directory (in the case below, the *ROOT* directory itself):
```
setenv ROOTSYS `pwd`
```
3. Check the list of available configurations:
```
./configure --help
```

4. Launch the configuration procedure for your architectures (*linux* in my case):
```
./configure linux
```
5. Launch the compilation:
```
gmake
```
6. Launch the installation:
```
gmake install
```
7. Protect *ROOT* against erasing:
```
chmod –R –w “.”
```

11.4 Preparing future work sessions with ROOT

ROOT is now installed on your system:. However, for future work sessions, you will need some variables to be set:

- *ROOTSYS* must indicate where *ROOT* is installed
- The *root* program must be in your path
- The *ROOT* libraries must be in your library path

To insure that this is always the case, both for you and for other users of your station, the best solution is to create configuration scripts for *ROOT* that will be executed each time you log on (note that you must have super-user privileges to do so):

1. Change into the sys-admin:
```
su -
```
2. Step into the directory where te log-in scripts are stored:
```
cd /etc/profile.d
```
3. Use your favorite text-editor two create a script *root.csh* with the following content:
```
setenv ROOTSYS /opt/root/root3.05.03
set path = ( $path $ROOTSYS/bin )
if ( $?LD_LIBRARY_PATH ) then
      setenv LD_LIBRARY_PATH “$LD_LIBRARY_PATH”:“$ROOTSYS/lib”
else
      setenv LD_LIBRARY_PATH “$ROOTSYS/lib”
endif
```
4. Similarly, use a favorite text-editor two create a script *root.sh* with the following content:
```
export ROOTSYS=/opt/root/root3.05.03
PATH=$(PATH):$ROOTSYS/bin
if [ X$LD_LIBRARY_PATH != X ] ; then
      LD_LIBRARY_PATH=$(LD_LIBRARY_PATH):$(ROOTSYS)/lib
else
      LD_LIBRARY_PATH=$(ROOTSYS)/lib
fi
export LD_LIBRARY_PATH
```
5. Change the execution rights of both scripts:
```
chmod +x root.*
```

6. Exit for the sys-admin mode:

```
exit
```

Now that these scripts have been introduced into the login script directory, the ROOT settings will be set for your next work session (if you want them for the current work session, you need to source the script manually).

12 Recompiling G4

There are some cases where you may want to recompile Geant4:

- You want to modify some installation options
- You want to add some drivers or user interfaces
- You have downloaded and installed a Geant4 patch

In these occasions you may want to try and recompile Geant4 directly with *make*, but there are often problems, as it is very difficult to set properly all the environment variables. In fact, the safest way to recompile Geant4 is to use the same procedure as when you compiled it first:

1. Step into the Geant4 installation directory

   ```
   cd /opt/geant4/geant4.4.1.p01
   ```

2. If you had write-protected the Geant4 installation, unprotect it

   ```
   chmod –R u+w “.”
   ```

3. Launch the configuration procedure and answer to all the script questions, then wait for the compilation to complete:

   ```
   ./configure -install
   ```

4. Update the Geant4 environment scripts:

   ```
   ./configure
   ```

5. Protect the Geant4 installation again

   ```
   chmod –R u-w “.”
   ```