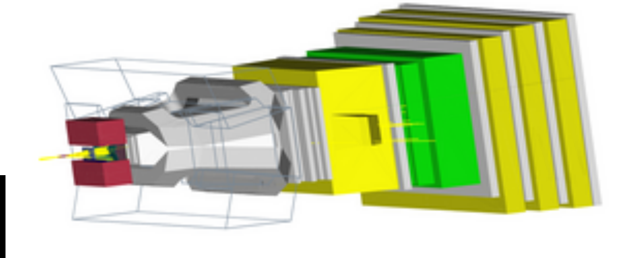
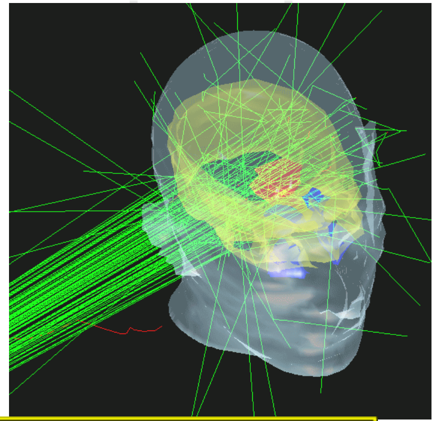
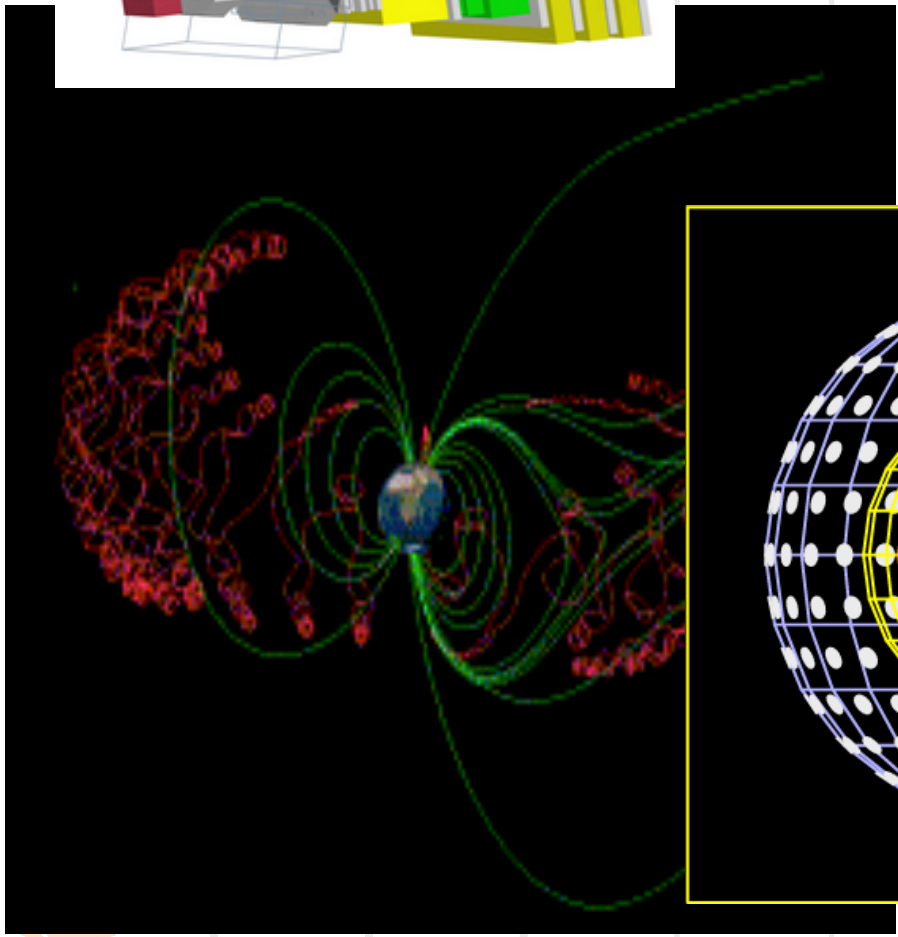
一、Geant4介绍

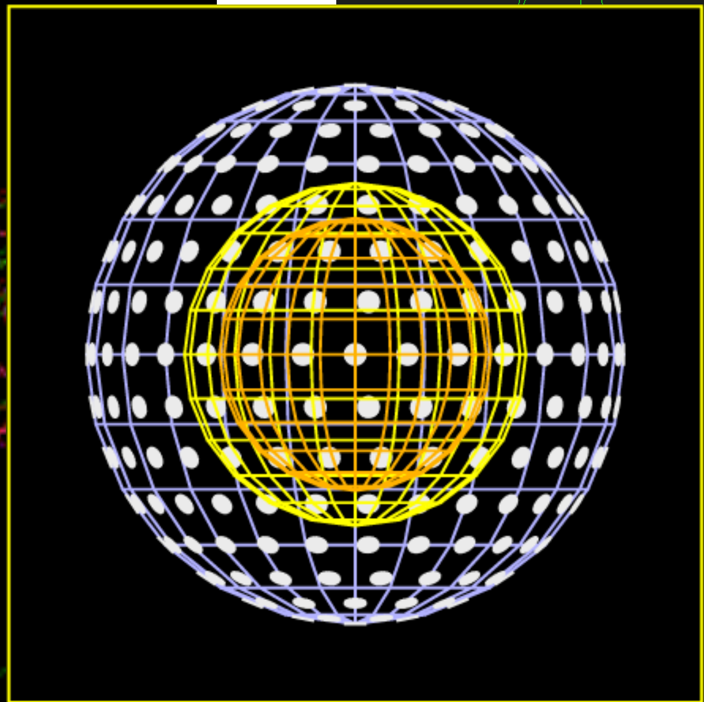
- 简介：Geant4是一款基于蒙特卡罗算法的处理粒子与物质相互作用的软件开发包。它是CERN(欧洲核子研究中心)RD44项目主导的多国合作的结晶，其最初想法是考虑如何将现代计算机技术应用并改进以FORTRAN为基础的Geant3（1993年）。之后1994年秋由CERN的探测器研究与开发委员会的RD44项目主导，决定开发一款全新的基于面向对象化技术的软件，能为下一代亚原子物理实验提供模拟。其后发现它能广泛地应用于核领域、加速器、航天、核医学物理等多个领域。其应用领域也在不断拓宽、功能不断加强。

☯应用领域广：涉及粒子与物质相互作用的各个领域









广泛地应用于高能物理、核物理、天体物理、加速器、核医学等多个粒子相关领域

- Geant4定义了大量的元素粒子和核，其默认定义的粒子超过100种
- 能模拟的粒子种类：一般的粒子，如电子，光子，质子等；短寿命共振粒子(resonant particles)如矢量介子(vector mesons)和δ重子(delta baryons)；原子核，如氘核，α粒子和重离子(包括超重核)；夸克，底夸克和胶子;
- 伴随这些粒子定义的属性包括名称、质量、电荷、自旋、同位旋、宇称、衰变模型等等。
- 分类：轻子，介子，重子，玻色自，短寿命粒子和离子

Geant4资源

官方主页： http://geant4.web.cern.ch/geant4/

google中查关键词 geant4 cern

国内蒙特卡罗学术论坛

[www.hejishult.cn（更全面）](http://www.hejishult.cn)

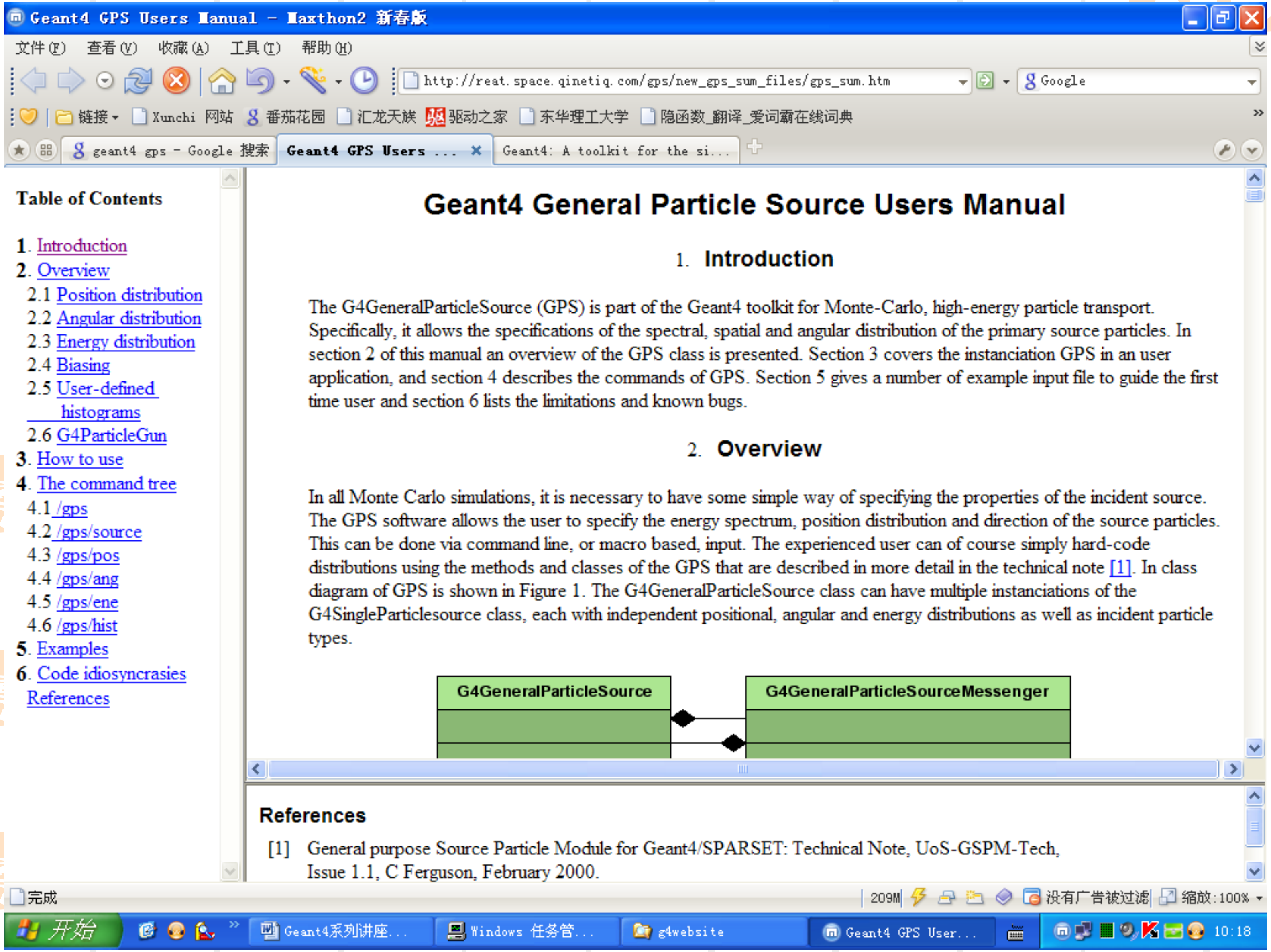
www.52mc.net/（更专业）

国内蒙特卡罗学术论坛，包括MCNP、EGS、GEANT、FlUKA等版块。

GeneralParticleSource 介绍网站

http://reat.space.qinetiq.com/gps/new_gps_sum_files/gps_sum.htm

google中查关键词 geant4 gps



Geant4 GPS Users Manual - Maxthon2 新春版

http://reat.space.qinetiq.com/gps/new_gps_sum_files/gps_sum.htm

Table of Contents

Geant4 General Particle Source Users Manual

1. Introduction

The G4GeneralParticleSource (GPS) is part of the Geant4 toolkit for Monte-Carlo, high-energy particle transport. Specifically, it allows the specifications of the spectral, spatial and angular distribution of the primary source particles. In section 2 of this manual an overview of the GPS class is presented. Section 3 covers the instanciation GPS in an user application, and section 4 describes the commands of GPS. Section 5 gives a number of example input file to guide the first time user and section 6 lists the limitations and known bugs.

2. Overview

In all Monte Carlo simulations, it is necessary to have some simple way of specifying the properties of the incident source. The GPS software allows the user to specify the energy spectrum, position distribution and direction of the source particles. This can be done via command line, or macro based, input. The experienced user can of course simply hard-code distributions using the methods and classes of the GPS that are described in more detail in the technical note [1]. In class diagram of GPS is shown in Figure 1. The G4GeneralParticleSource class can have multiple instanciations of the G4SingleParticlesource class, each with independent positional, angular and energy distributions as well as incident particle types.

G4GeneralParticleSource

G4GeneralParticleSourceMessenger

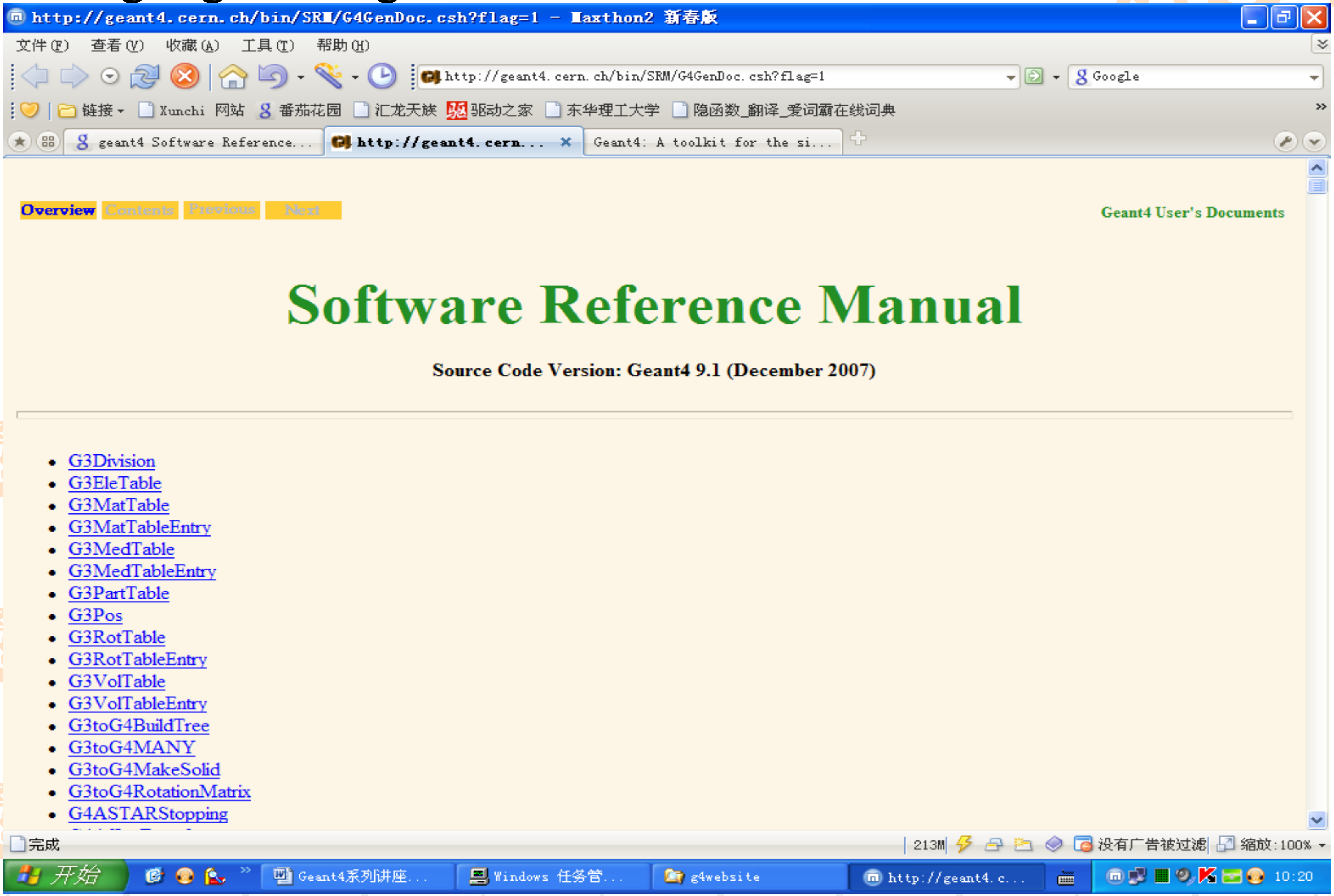
References

[1] General purpose Source Particle Module for Geant4/SPARSET: Technical Note, UoS-GSPM-Tech, Issue 1.1, C Ferguson, February 2000.

类定义参考

http://geant4.cern.ch/bin/SRM/G4GenDoc.csh?flag=1

google中查 geant4 Software Reference



二、Geant4安装及Linux系统介绍

支持Geant4的系统平台：

- UNIX systems
  - SUN-SunOS v.5.8, CC v.5.4 (WS6)
    - G4SYSTEM: SUN-CC
- Linux systems
  - CERN Scientific Linux SLC3， gcc 3.2.3
    - G4SYSTEM: Linux-g++
- Windows systems
  - Win/XP & Cygwin32, MSVC++ 7.1 .NET
    - G4SYSTEM: WIN32-VC7
- Other systems, not (yet) officially supported
  - MacOS 10.3 and g++ gcc 3.3
    - G4SYSTEM: Darwin-g++

学习Geant4需要掌握的软件知识

C＋＋基本知识：

C＋＋基础知识，Geant4程序以C＋＋程序语言编写，掌握C＋＋语法基本概念；

Linux/Unix知识：

熟悉Linux操作系统，掌握Linux/Unix基本命令，学会用C＋＋编译器编译程序；

面向对象化技术：

掌握基本概念，若涉及复杂模拟系统需要进一步了解；

Linux系统简介与Geant4安装

在Windows中安装Geant4需要的软件：

- Cygwin，虚拟机模拟Linux工作环境
- MSVC++ 7.1 .NET，用VisualC＋＋2005 版完全安装即可
- CLHEP
- Geant4安装文件

（运行相当慢，不建议采用）

Linux操作系统核心最早是由芬兰的Linus Torvalds 1991年8月在芬兰赫尔辛基大学上学时发布的，后来经过众多世界顶尖的软件工程师的不断修改和完善，Linux得以在全球普及开来，在服务器领域及个人桌面版得到越来越多的应用

Linux文件系统不同与Windows，它没有盘符的概念，其采用阶层式目录结构管理文件；

Shell：操作系统与外部最主要的接口就叫做shell。shell是操作系统最外面的一层。shell管理你与操作系统之间的交互；向操作系统解释你的输入，并且处理各种各样的操作系统的输出结果。Geant4程序的运行通过Shell输入命令完成。(有点类似Dos，单功能强大许多)

Linux系统安装

安装Scientific Linux 4.3， Geant4对gcc编译器的版本有要求，这个版本的Linux符合要求(也可以用别的Linux版本，然后装个合适的gcc就可以了)。

Scientific Linux 下载地址：

ftp://ftp.scientificlinux.org/linux/scientific/43/iso/i386

找到对应的系统下载，64位cpu的可以下载64的系统（连同校验码一起下载），然后下载个校验工具进行校验（MD5或其它校验工具）。若校验成功刻录成iso光盘就可以安装了。

若装双系统，把Linux安装到最后一个分区，在最后个分区空出10G空间。

安装步骤（双系统）：

- 腾出最后一个分区10G左右空间，用分区工具删除，使之成为空闲分区
- 插入安装盘，重启，进入安装界面，校验光盘；
- 选择鼠标类型、语言等基本信息；
- 选择安装分区（使用空闲分区），创建boot分区（100M）
- 创建swap分区（600M），剩下的给挂载点（/）

硬盘安装可参考：http://www.linuxdiyf.com/bbs/rf/1.htm

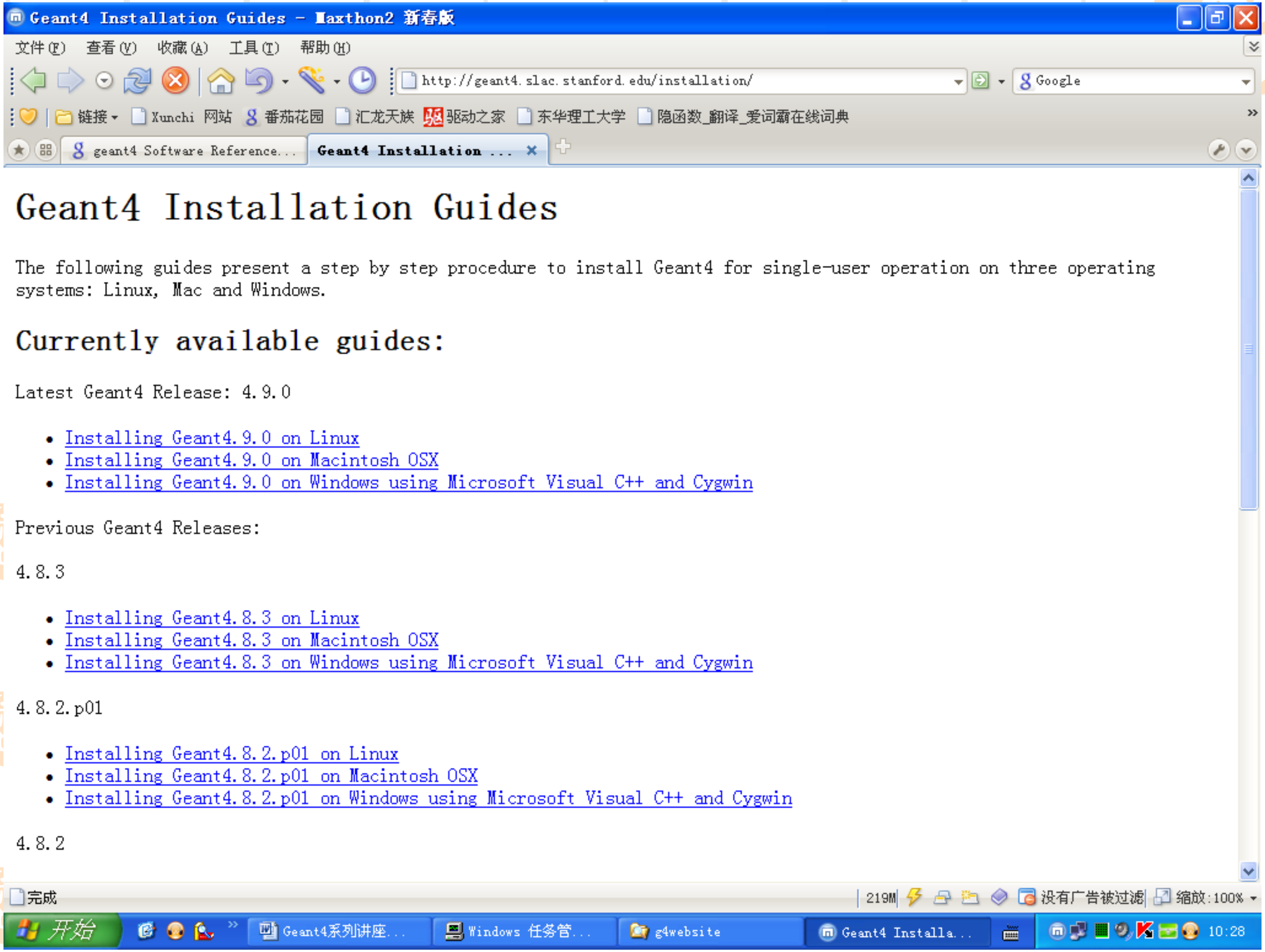
在Linux系统中安装Geant4

Linux系统下软件安装根据安装包的形式有多种安装方法。Geant4采用源码安装的方式，其安装包为“.tar.gz”或“.tgz”形式的源码包(clhep-2.0.3.1.tgz, G4EMLOW.4.3.tar.gz, geant4.9.0.tar.gz)

其安装过程分为以下几步：

- 解压缩 tar –zxvf clhep-2.0.3.1.tgz
- 进入解压后的目录，进行配置 ./configure
- 编译 make
- 安装 make install

具体安装按照官方网上步骤来:



Geant4 Installation Guides

The following guides present a step by step procedure to install Geant4 for single-user operation on three operating systems: Linux, Mac and Windows.

Currently available guides:

Latest Geant4 Release: 4.9.0

- Installing Geant4.9.0 on Linux
- Installing Geant4.9.0 on Macintosh OSX
- Installing Geant4.9.0 on Windows using Microsoft Visual C++ and Cygwin

Previous Geant4 Releases:

4.8.3

- Installing Geant4.8.3 on Linux
- Installing Geant4.8.3 on Macintosh OSX
- Installing Geant4.8.3 on Windows using Microsoft Visual C++ and Cygwin

4.8.2.p01

- Installing Geant4.8.2.p01 on Linux
- Installing Geant4.8.2.p01 on Macintosh OSX
- Installing Geant4.8.2.p01 on Windows using Microsoft Visual C++ and Cygwin

4.8.2

环境变量设置脚本

将以下内容保存到成名为script的文本文件，存放到Linux系统用户名路径下

```
!# /bin/sh
export G4WORKDIR=/g4work
export LD_LIBRARY_PATH=/geant4set/CLHEP/lib/:$LD_LIBRARY_PATH
source /geant4set/geant4/geant4.9.0/env.sh
Bash
```

上述路径可根据具体系统设置更改，从终端进入保持该文件的目录，输入

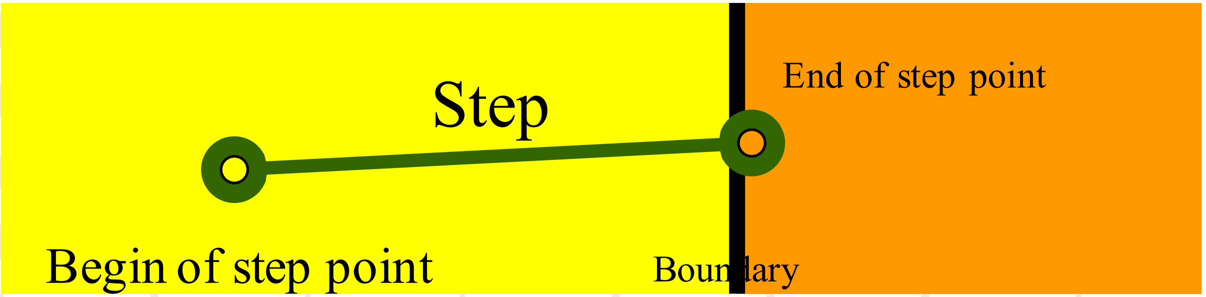
chmod 777 script

更改文件为可读写可执行文件

此后要运行geant4，只需要打开终端，输入 ./script 就完成了环境变量设置

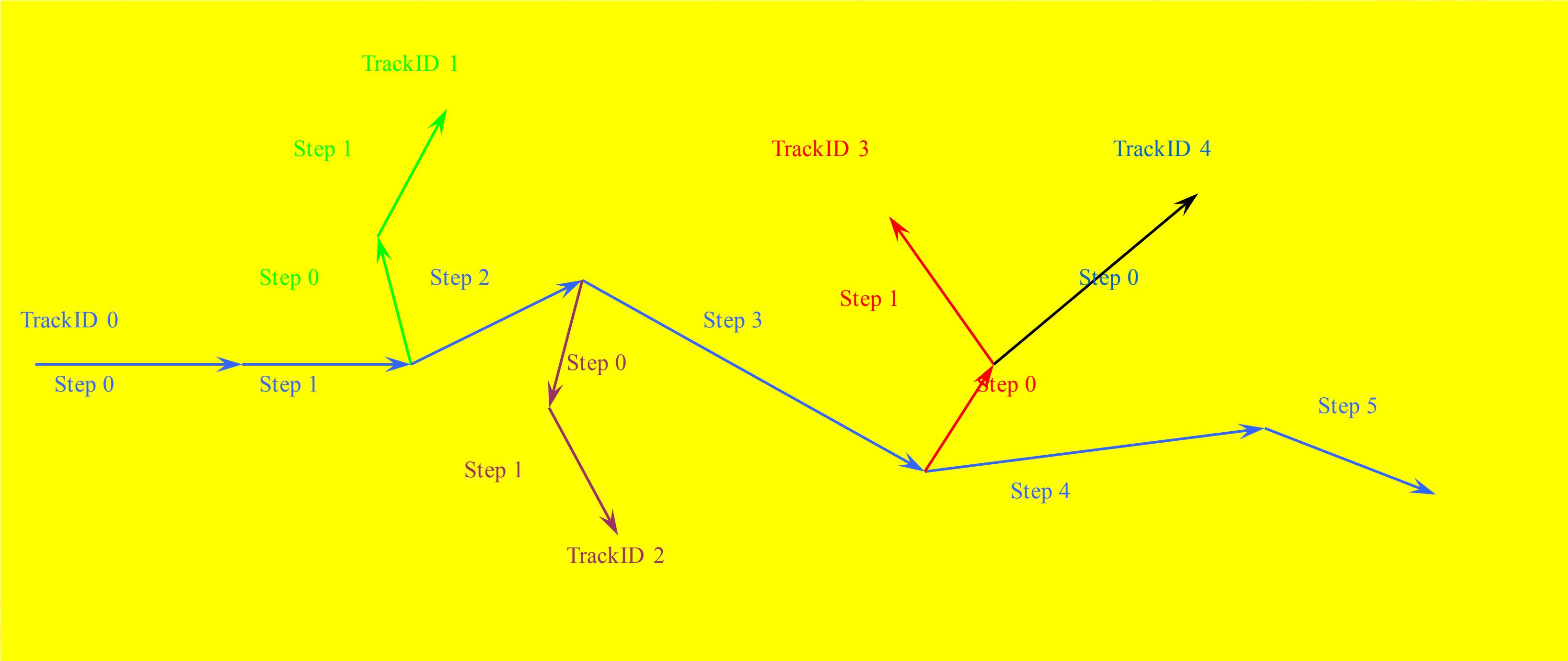
三、Geant4中的一些概念

Step：



Geant4以被输运的粒子为对象，每相邻两次碰撞点组成一个Step，通过Step可以提取碰撞点前后的粒子状态信息，以及粒子在两个碰撞点之间的变化信息，如对应Step的能量损失、动量变化、粒子飞行时间等；

Track 、Event 和Run



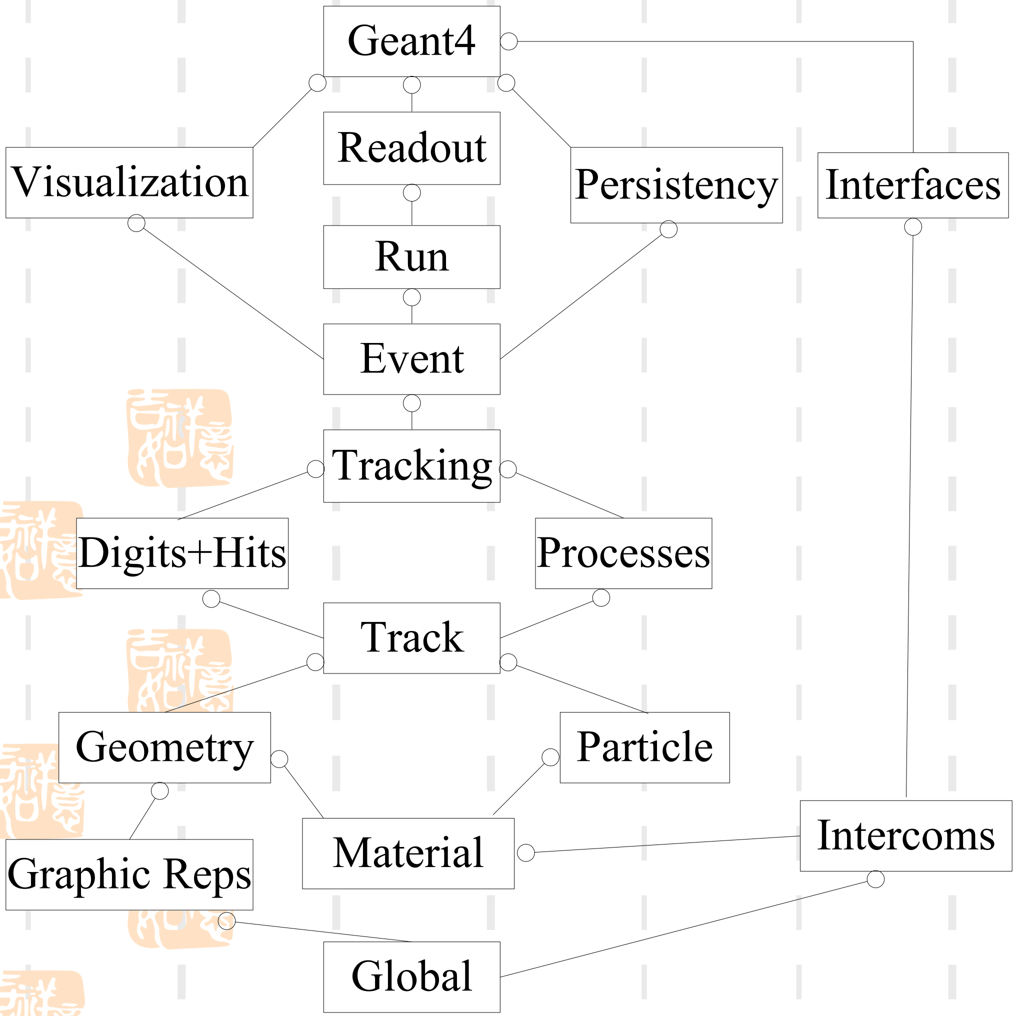
●多个连续的Step组成Track；每一Track有对应的TrackID号和ParentID号，通过Track也可以提取粒子信息；

●一次入射粒子事件为一个Event，它可以由多个Track组成，Event也有ID号；

●多次入射Event事件称为Run；

四、Geant4程序框架

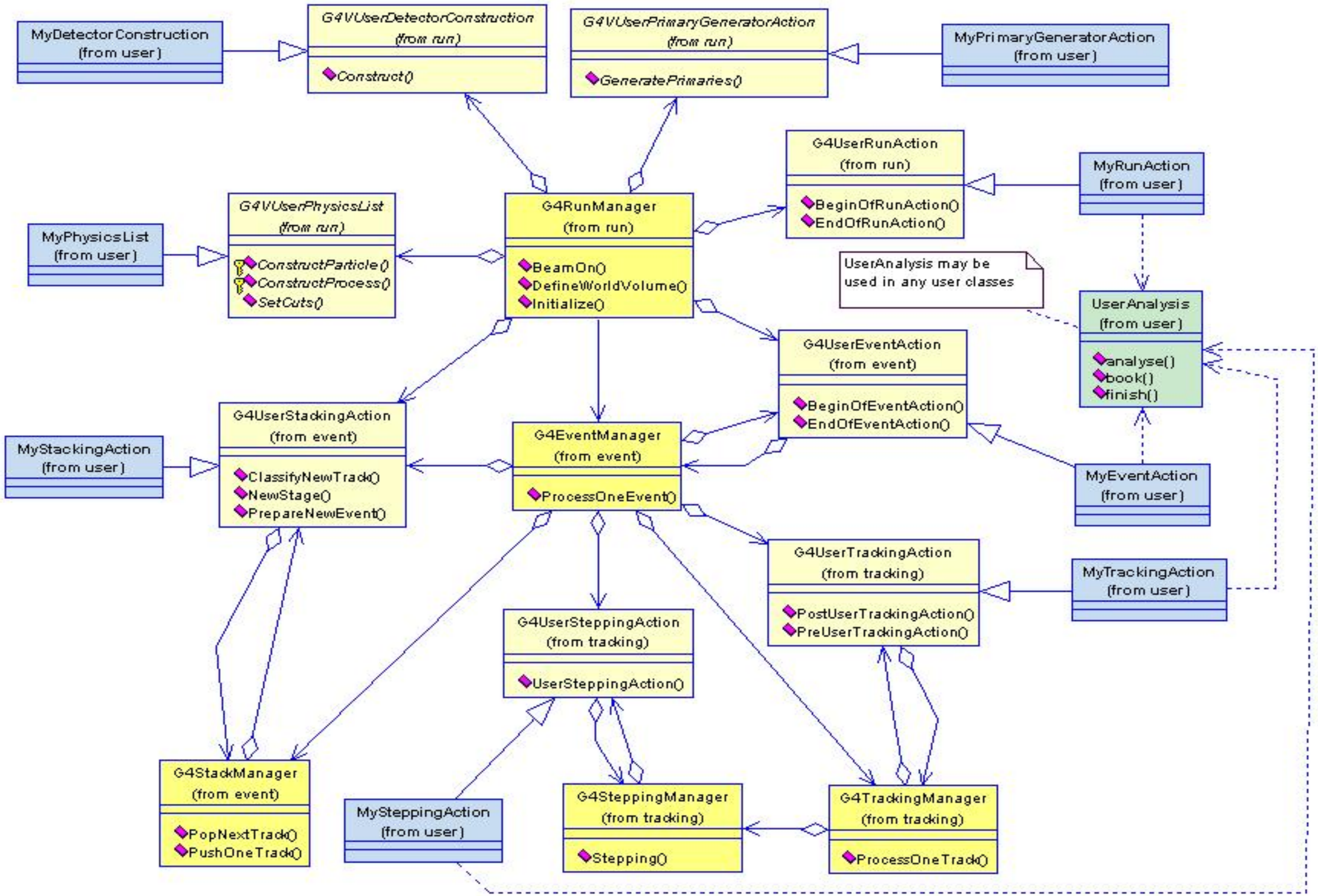
结构框架



Geant4基于面向对象技术采用模块化设计，编写语言为C++。它通过类机制将相互联系紧密的类封装一起以完成特定的功能。通过尽量减少不同的类模块之间的联系来完成模块化设计。

Overview of Geant4 advanced examples

MGP
26 October 2001



C语言：
File1.hh

```
        struct student{
                int unm;
                char name[30];
                }a;
        ............
```

File2.hh

```
        .............
```

```
include “file1.hh”
include “file2.hh”
Main（）
{ int i，j，k；
 .....
对a操作
  ...

}
```

对比C语言

```
#include "G4RunManager.hh"
#include "G4UImanager.hh"
.....
#include "ExN01PhysicsList.hh"
#include "ExN01PrimaryGeneratorAction.hh"
int main()
{
G4RunManager* runManager = new G4RunManager;

  // set mandatory initialization classes
G4VUserDetectorConstruction* detector = new ExN01DetectorConstruction;
  runManager->SetUserInitialization(detector);
......................
......................
  return 0;
}
```

exampleN01

ExampleN01包含文件：

- include文件夹：
  - ExN01DetectorConstruction.hh
  - ExN01PhysicsList.hh
  - ExN01PrimaryGeneratorAction.hh
- src文件夹：
  - ExN01DetectorConstruction.cc
  - ExN01PhysicsList.cc
  - ExN01PrimaryGeneratorAction.cc
- exampleN01.cc
- GNUmakefile

exampleN01.cc

```
// $Id: exampleN01.cc,v 1.6 2006/06/29 17:47:10 gunter Exp $
// GEANT4 tag $Name: geant4-09-00 $
// --------------------------------------------------------------
//      GEANT 4 - exampleN01
// --------------------------------------------------------------

#include "G4RunManager.hh"
#include "G4UImanager.hh"

#include "ExN01DetectorConstruction.hh"
#include "ExN01PhysicsList.hh"
#include "ExN01PrimaryGeneratorAction.hh"

int main()
{
  // Construct the default run manager
  //
  G4RunManager* runManager = new G4RunManager;

  // set mandatory initialization classes
  //
  G4VUserDetectorConstruction* detector = new ExN01DetectorConstruction;
  runManager->SetUserInitialization(detector);
  //
  G4VUserPhysicsList* physics = new ExN01PhysicsList;
  runManager->SetUserInitialization(physics);

  // set mandatory user action class
  //
  G4VUserPrimaryGeneratorAction* gen_action = new ExN01PrimaryGeneratorAction;
```

exampleN01.cc

```
runManager->SetUserAction(gen_action);

// Initialize G4 kernel
//
runManager->Initialize();

// Get the pointer to the UI manager and set verbosities
//
G4UImanager* UI = G4UImanager::GetUIpointer();
UI->ApplyCommand("/run/verbose 1");
UI->ApplyCommand("/event/verbose 1");
UI->ApplyCommand("/tracking/verbose 1");

// Start a run
//
G4int numberOfEvent = 3;
runManager->BeamOn(numberOfEvent);

// Job termination
//
// Free the store: user actions, physics_list and detector_description are
//             owned and deleted by the run manager, so they should not
//             be deleted in the main() program !
//
delete runManager;

return 0;
}
```

ExN01DetectorConstruction.hh

```
// $Id: ExN01DetectorConstruction.hh,v 1.6 2006/06/29 17:47:13 gunter Exp $
#ifndef ExN01DetectorConstruction_H
#define ExN01DetectorConstruction_H 1

class G4LogicalVolume;
class G4VPhysicalVolume;

#include "G4VUserDetectorConstruction.hh"

class ExN01DetectorConstruction : public G4VUserDetectorConstruction
{
  public:

    ExN01DetectorConstruction();
    ~ExN01DetectorConstruction();

    G4VPhysicalVolume* Construct();
private:
    // Logical volumes
    //
    G4LogicalVolume* experimentalHall_log;
    G4LogicalVolume* tracker_log;
    G4LogicalVolume* calorimeterBlock_log;
    G4LogicalVolume* calorimeterLayer_log;
    // Physical volumes
    //
    G4VPhysicalVolume* experimentalHall_phys;
    G4VPhysicalVolume* calorimeterLayer_phys;
    G4VPhysicalVolume* calorimeterBlock_phys;
    G4VPhysicalVolume* tracker_phys;
};
#endif
```

ExN01DetectorConstruction.cc

```
// $Id: ExN01DetectorConstruction.cc,v 1.9 2006/06/29 17:47:19 gunter Exp $
// GEANT4 tag $Name: geant4-09-00 $
//

#include "ExN01DetectorConstruction.hh"

#include "G4Material.hh"
#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4LogicalVolume.hh"
#include "G4ThreeVector.hh"
#include "G4PVPlacement.hh"
#include "globals.hh"

ExN01DetectorConstruction::ExN01DetectorConstruction()
 :  experimentalHall_log(0), tracker_log(0),
    calorimeterBlock_log(0), calorimeterLayer_log(0),
    experimentalHall_phys(0), calorimeterLayer_phys(0),
    calorimeterBlock_phys(0), tracker_phys(0)
{;}

ExN01DetectorConstruction::~ExN01DetectorConstruction()
{
}

G4VPhysicalVolume* ExN01DetectorConstruction::Construct()
{

  //------------------------------------------------------ materials
```

ExN01DetectorConstruction.cc

```
G4double a;  // atomic mass
G4double z;  // atomic number
G4double density;

G4Material* Ar =
new G4Material("ArgonGas", z= 18., a= 39.95*g/mole, density= 1.782*mg/cm3);

G4Material* Al =
new G4Material("Aluminum", z= 13., a= 26.98*g/mole, density= 2.7*g/cm3);

G4Material* Pb =
new G4Material("Lead", z= 82., a= 207.19*g/mole, density= 11.35*g/cm3);

//------------------------------------------------ volumes

//------------------------------ experimental hall (world volume)
//------------------------------ beam line along x axis

G4double expHall_x = 3.0*m;
G4double expHall_y = 1.0*m;
G4double expHall_z = 1.0*m;
G4Box* experimentalHall_box
  = new G4Box("expHall_box",expHall_x,expHall_y,expHall_z);
experimentalHall_log = new G4LogicalVolume(experimentalHall_box,
                                 Ar,"expHall_log",0,0,0);
experimentalHall_phys = new G4PVPlacement(0,G4ThreeVector(),
                          experimentalHall_log,"expHall",0,false,0);
```

ExN01DetectorConstruction.cc

```
//---------------------------- a tracker tube
G4double innerRadiusOfTheTube = 0.*cm;
G4double outerRadiusOfTheTube = 60.*cm;
G4double hightOfTheTube = 50.*cm;
G4double startAngleOfTheTube = 0.*deg;
G4double spanningAngleOfTheTube = 360.*deg;
G4Tubs* tracker_tube = new G4Tubs("tracker_tube",innerRadiusOfTheTube,
                         outerRadiusOfTheTube,hightOfTheTube,
                         startAngleOfTheTube,spanningAngleOfTheTube);
tracker_log = new G4LogicalVolume(tracker_tube,Al,"tracker_log",0,0,0);
G4double trackerPos_x = -1.0*m;
G4double trackerPos_y = 0.*m;
G4double trackerPos_z = 0.*m;
tracker_phys = new G4PVPlacement(0,
        G4ThreeVector(trackerPos_x,trackerPos_y,trackerPos_z),
        tracker_log,"tracker",experimentalHall_log,false,0);
//---------------------------- a calorimeter block
G4double block_x = 1.0*m;
G4double block_y = 50.0*cm;
G4double block_z = 50.0*cm;
G4Box* calorimeterBlock_box = new G4Box("calBlock_box",block_x,
                              block_y,block_z);
calorimeterBlock_log = new G4LogicalVolume(calorimeterBlock_box,
                              Pb,"caloBlock_log",0,0,0);
G4double blockPos_x = 1.0*m;
G4double blockPos_y = 0.0*m;
```

ExN01DetectorConstruction.cc

```
    G4double blockPos_z = 0.0*m;
    calorimeterBlock_phys = new G4PVPlacement(0,
                G4ThreeVector(blockPos_x,blockPos_y,blockPos_z),
                calorimeterBlock_log,"caloBlock",experimentalHall_log,false,0);
//---------------------------- calorimeter layers
G4double calo_x = 1.*cm;
    G4double calo_y = 40.*cm;
    G4double calo_z = 40.*cm;
    G4Box* calorimeterLayer_box = new G4Box("caloLayer_box",
                                    calo_x,calo_y,calo_z);
    calorimeterLayer_log = new G4LogicalVolume(calorimeterLayer_box,
                                    Al,"caloLayer_log",0,0,0);
    for(G4int i=0;i<19;i++) // loop for 19 layers
    {
      G4double caloPos_x = (i-9)*10.*cm;
      G4double caloPos_y = 0.0*m;
      G4double caloPos_z = 0.0*m;
      calorimeterLayer_phys = new G4PVPlacement(0,
                G4ThreeVector(caloPos_x,caloPos_y,caloPos_z),
                calorimeterLayer_log,"caloLayer",calorimeterBlock_log,false,i);
    }
//-----------------------------------------------------------
    return experimentalHall_phys;
  }
```

ExN01PhysicsList.hh

```
// $Id: ExN01PhysicsList.hh,v 1.6 2006/06/29 17:47:15 gunter Exp $
// GEANT4 tag $Name: geant4-09-00 $
// ExN01PhysicsList
//  Construct/define particles and physics processes
//
//  Particle defined in ExampleN01
//    geantino
//  Process defined in ExampleN01
//    transportation

#ifndef ExN01PhysicsList_h
#define ExN01PhysicsList_h 1

#include "G4VUserPhysicsList.hh"
#include "globals.hh"

class ExN01PhysicsList: public G4VUserPhysicsList
{
  public:
    ExN01PhysicsList();
    ~ExN01PhysicsList();
protected:
    // Construct particle and physics process
    void ConstructParticle();
    void ConstructProcess();
    void SetCuts();
};
#endif
```

ExN01PhysicsList.cc


```
// $Id: ExN01PhysicsList.cc,v 1.6 2006/06/29 17:47:21 gunter Exp $
// GEANT4 tag $Name: geant4-09-00 $

#include "ExN01PhysicsList.hh"
#include "G4ParticleTypes.hh"
ExN01PhysicsList::ExN01PhysicsList()
{;}
ExN01PhysicsList::~ExN01PhysicsList()
{;}
void ExN01PhysicsList::ConstructParticle()
{ G4Geantino::GeantinoDefinition();
}
void ExN01PhysicsList::ConstructProcess()
{  // Define transportation process
  AddTransportation();
}
void ExN01PhysicsList::SetCuts()
{  // uppress error messages even in case e/gamma/proton do not exist
  G4int temp = GetVerboseLevel();                                    SetVerboseLevel(0);
  //  " G4VUserPhysicsList::SetCutsWithDefault" method sets
  //   the default cut value for all particle types
  SetCutsWithDefault();
  // Retrieve verbose level
  SetVerboseLevel(temp);
}
```

ExN01PrimaryGeneratorAction.hh

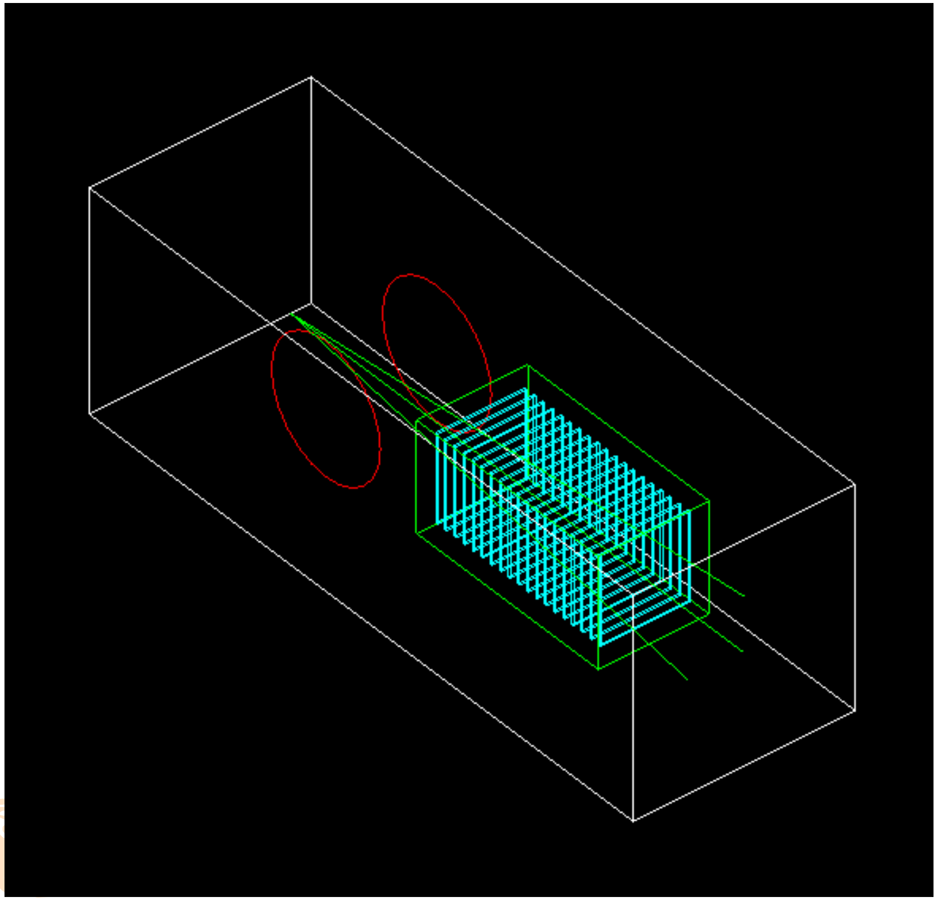
```
// $Id: ExN01PrimaryGeneratorAction.hh,v 1.5 2006/06/29 17:47:17 gunter Exp $
// GEANT4 tag $Name: geant4-09-00 $
//

#ifndef ExN01PrimaryGeneratorAction_h
#define ExN01PrimaryGeneratorAction_h 1

#include "G4VUserPrimaryGeneratorAction.hh"

class G4ParticleGun;
class G4Event;

class ExN01PrimaryGeneratorAction : public G4VUserPrimaryGeneratorAction
{
  public:
    ExN01PrimaryGeneratorAction();
    ~ExN01PrimaryGeneratorAction();

  public:
    void GeneratePrimaries(G4Event* anEvent);

  private:
    G4ParticleGun* particleGun;
};

#endif
```

ExN01PrimaryGeneratorAction.cc

```
// $Id: ExN01PrimaryGeneratorAction.cc,v 1.6 2006/06/29 17:47:23 gunter Exp $
// GEANT4 tag $Name: geant4-09-00 $

#include "ExN01PrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "G4ParticleGun.hh"
#include "G4ParticleTable.hh"
#include "G4ParticleDefinition.hh"
#include "globals.hh"

ExN01PrimaryGeneratorAction::ExN01PrimaryGeneratorAction()
{
  G4int n_particle = 1;
  particleGun = new G4ParticleGun(n_particle);

  G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
  G4String particleName;
  particleGun->SetParticleDefinition(particleTable->FindParticle(particleName="geantino"));
  particleGun->SetParticleEnergy(1.0*GeV);
  particleGun->SetParticlePosition(G4ThreeVector(-2.0*m, 0.0, 0.0));
}
```

ExN01PrimaryGeneratorAction.cc

```
ExN01PrimaryGeneratorAction::~ExN01PrimaryGeneratorAction()
{
  delete particleGun;
}
void ExN01PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
  G4int i = anEvent->GetEventID() % 3;
  G4ThreeVector v(1.0,0.0,0.0);
  switch(i)
  {
    case 0:
      break;
    case 1:
      v.setY(0.1);
      break;
    case 2:
      v.setZ(0.1);
      break;
  }
  particleGun->SetParticleMomentumDirection(v);
  particleGun->GeneratePrimaryVertex(anEvent);
}
```



Geant4的运行模式:

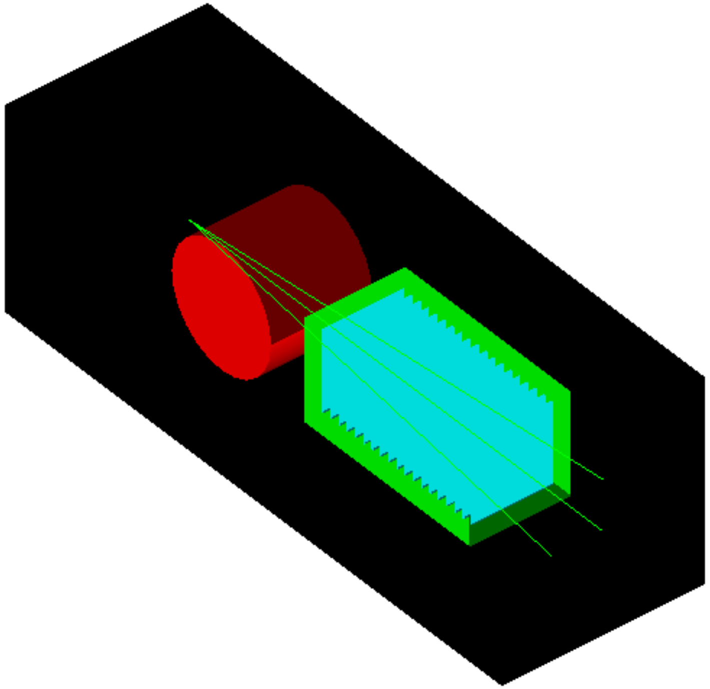
1）Hard－coded模式

2）interactive模式，命令行

3）batch模式，macro文件读入命令

2）混合模式

每种模式的实现必须在Main()中通过程序实现



ExampleN01的几何结构图，以体和线的形式分别显示；