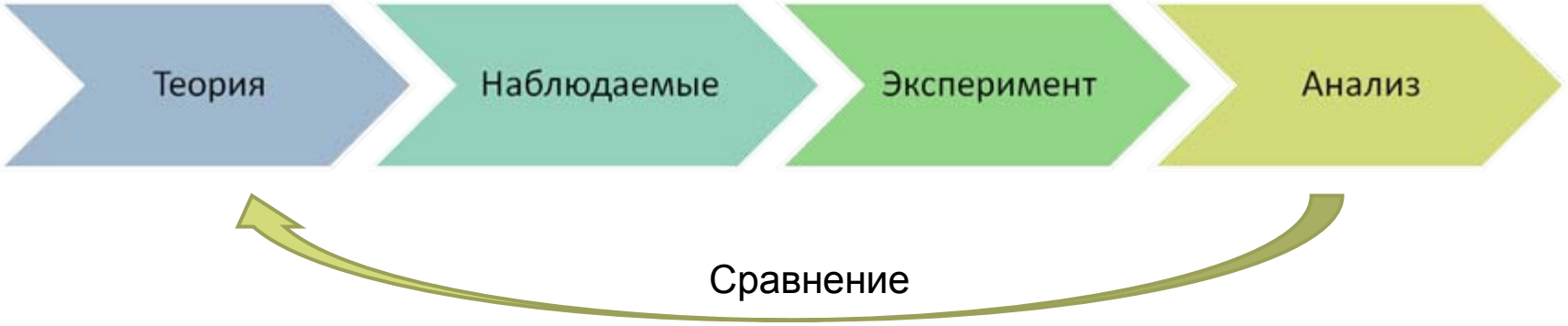
Компьютерные технологии в физике элементарных частиц.

Моделирование

Идеальный эксперимент

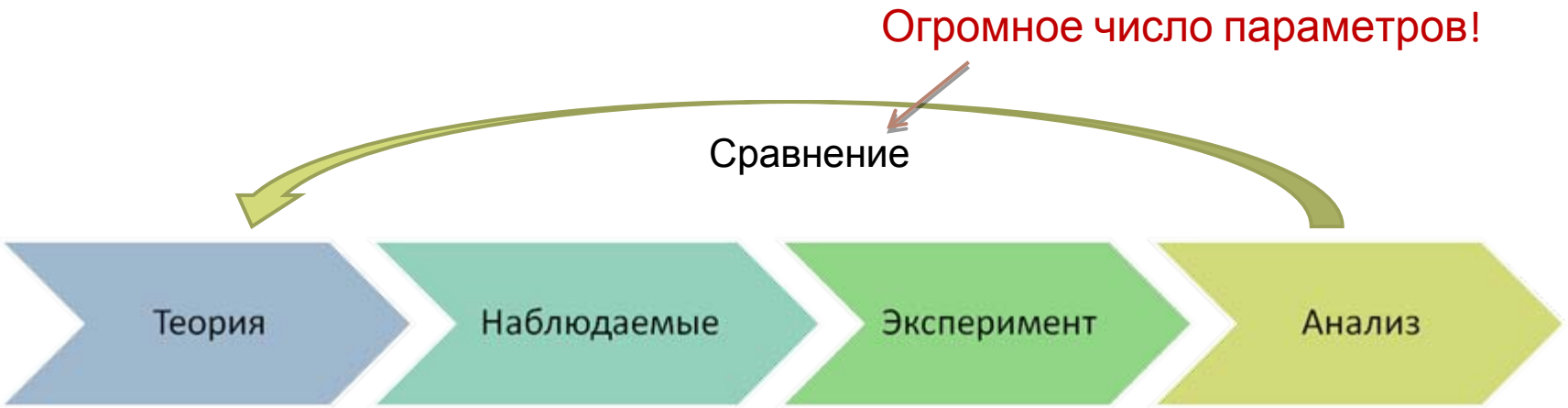


«Идеальная» постановка эксперимента:

- Теория предсказывает некоторые наблюдаемые,
- которые мы измеряем в эксперименте.
- Сравнение предсказанных и измеренных значений наблюдаемых позволяет сделать вывод о правомерности теории.

В реальной жизни все гораздо сложнее...

Реальный эксперимент



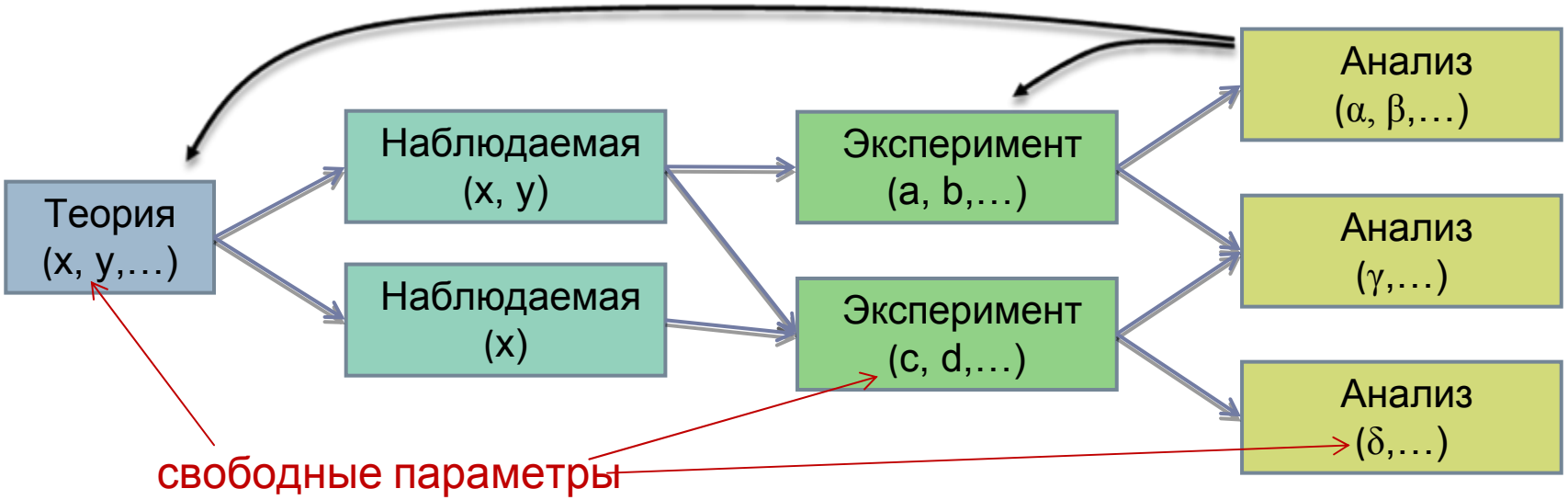
В теории есть неизвестные свободные параметры, например, масса Хиггса

Теория предсказывает множество наблюдаемых, которые по разному зависят от свободных параметров теории

Наблюдаемые измеряются в разных (нескольких) экспериментах, результат которых зависит от напряжений, давлений, эффективности, акцептанса,...

Одни и те же данные эксперимента обрабатываются с помощью разных алгоритмов, каждый из которых зависит от множества параметров (критерии отбора,...)

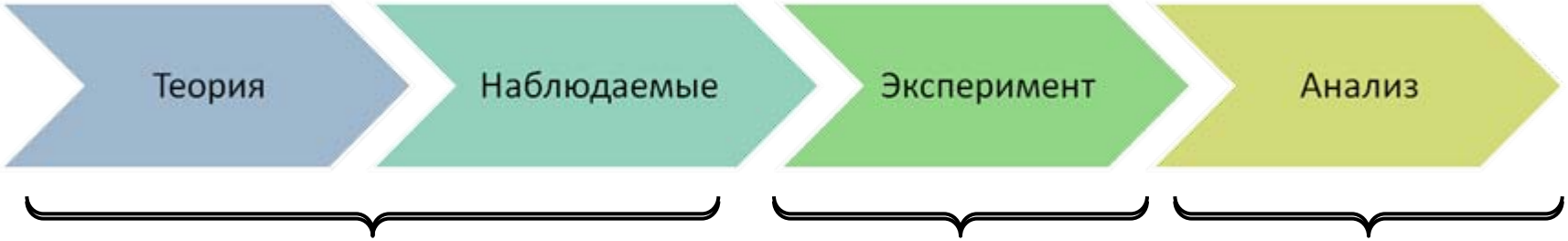
Зачем нужно моделировать эксперимент?



Из-за наличия огромного количества параметров, невозможно спроектировать, провести и проинтерпретировать результаты современного эксперимента в физике высоких энергий без помощи специального программного обеспечения, которое позволяет смоделировать всю цепочку от теории до результата для множества разных значений свободных параметров.

Задачи моделирования

GEANT4



Первичный генератор
•моделирование всей сложности теоретической модели

Моделирование детектора
•взаимодействие частиц с веществом детектора
•отклик электроники
•учет шумов и неработающих каналов
•определение акцептанса

Моделирование анализа
•оценка чувствительности (статистической точности)
•анализ систематических ошибок, смещений,...

Эффективность и акцептанс

Очень часто моделирование используется для определения акцептанса детектора. Конструкция современных детекторов настолько сложна, что, как правило, другим способом определить акцептанс невозможно.

Эффективность – вероятность зарегистрировать событие, если оно произошло. Определяется свойствами детектора.

Переменные, описывающие событие (например, импульсы и координаты частиц,…)

$$\text{Акцептанс } a = \int \varepsilon(\bar{x}) f(\bar{x}) d\bar{x}$$

Средняя эффективность по полному ансамблю исследуемого класса событий. Зависит как от свойств детектора, так и от параметров теории.

Многомерная плотность вероятности для исследуемых событий. Предсказывается теорией.

Программные пакеты для моделирования экспериментов в ФВЭ

На конференции MC200 Conference, Lisbon, October 2000, было представлено большое количество различных пакетов для моделирования:

EGS4, EGS5, EGSnrc
Geant3, Geant4
MARS
MCNP, MCNPX, A3MCNP, MCNP-DSP, MCNP4B
MVP, MVP-BURN
Penelope
Peregrine
Tripoli-3, Tripoli-3 A, Tripoli-4

Многие из них до сих пор существуют в своих нишах, но наиболее распространенным стал пакет Geant4

DPM
EA-MC
FLUKA
GEM
HERMES
LAHET
MCBEND
MCU
MF3D
NMTC
MONK
MORSE
RTS&T-2000
SCALE
TRAX
VMC++

GEANT4

GEANT4 = GEometry ANd Tracking, version 4

Это набор программных инструментов, позволяющих моделировать прохождение частиц через вещество. Функциональность Geant4 включает описание геометрии и состава вещества, модели взаимодействия частиц с веществом, проведение частиц, описание отклика чувствительных элементов детектора.

Официальная веб-страница: http://www.cern.ch/geant4

История GEANT-ов началась в 70-х годах в ЦЕРН.

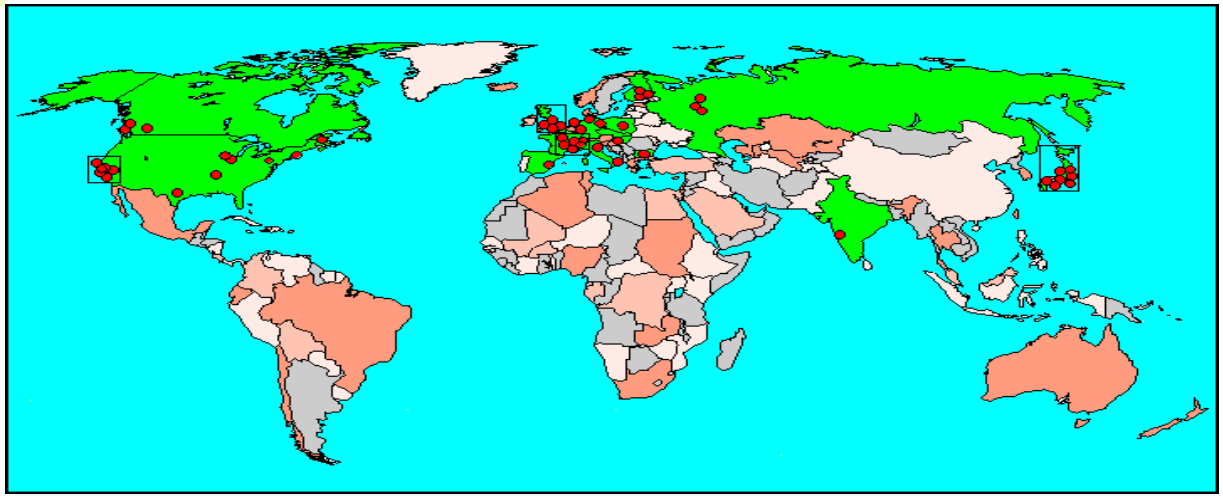
В 80-х - 90-х годах основным пакетом для моделирования был GEANT3, написанный на FORTRAN.

GEANT4 – наследник GEANT3, кардинально переработанный, написанный на C++, со значительно более широкими возможностями

Создание Geant4 началось в 1994, первый релиз состоялся в 1998.

Geant4 используется не только в ФВЭ, но и в медицинских приложениях (ядерная медицина), в космических исследованиях.

Коллаборация GEANT4



TRIUMF

Lebedev

J.W.Goethe Universität

Collaborators also from non-member institutions, including
Budker Inst. of Physics
IHEP Protvino
MEPHI Moscow
Pittsburg University

Применение GEANT4

- Физика высоких энергий

  Изначально Geant4 был предназначен именно для моделирования экспериментов в области физики высоких энергий. Широкое применение началось в начале 2000-х. На сегодняшний день, Geant4 – фактически стандарт в ФВЭ и используется в сотнях экспериментах.

- Астрофизика

  Моделирование детекторных космических экспериментов
  Моделирование космических лучей
  Моделирование действия радиации на космические аппараты

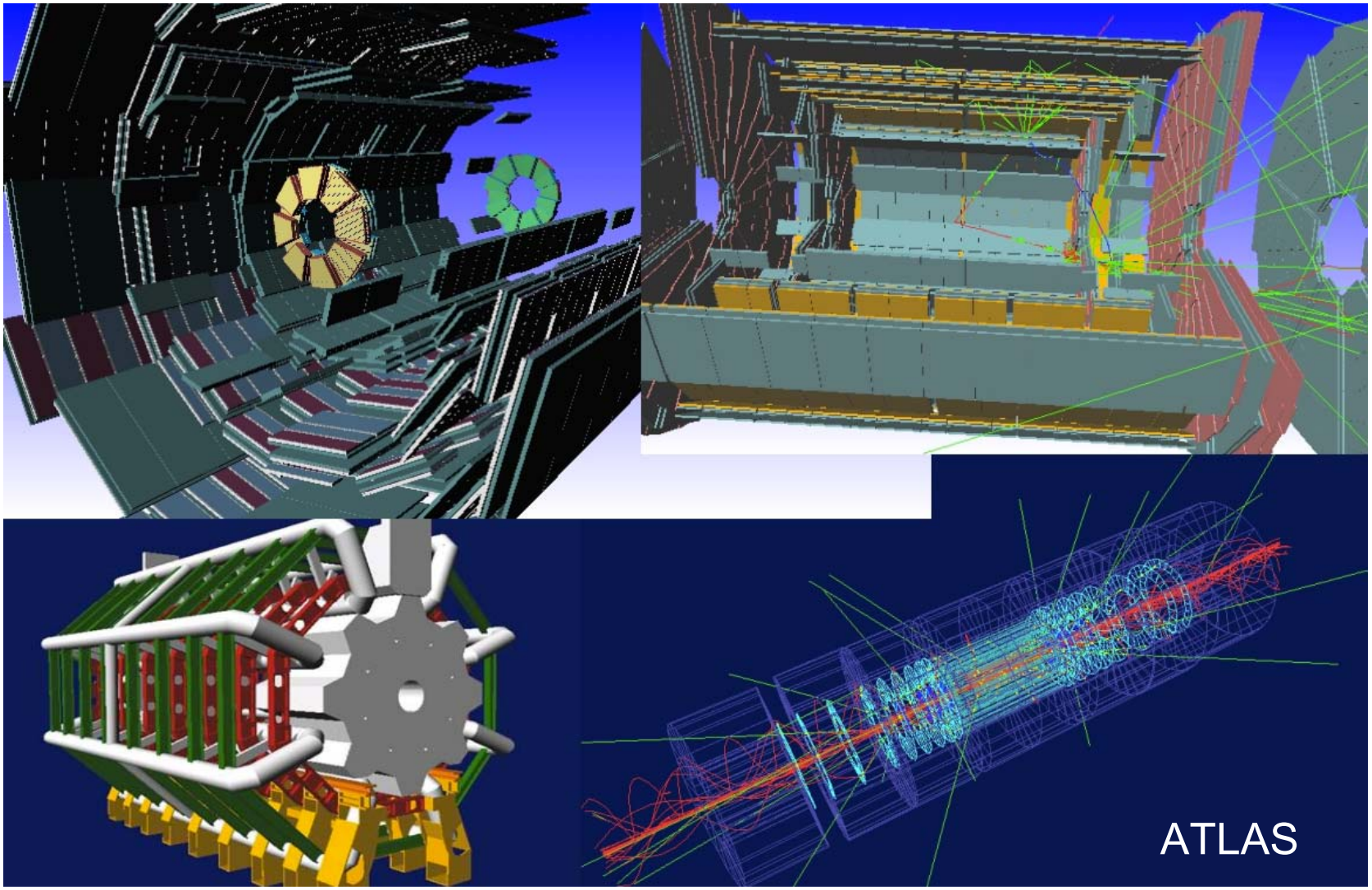
- Медицина и биология

  Ядерная медицина: моделирование поглощенной дозы и эффекта от облучения
  Позитронная томография

- Ускорители

  Моделирование поведения пучка

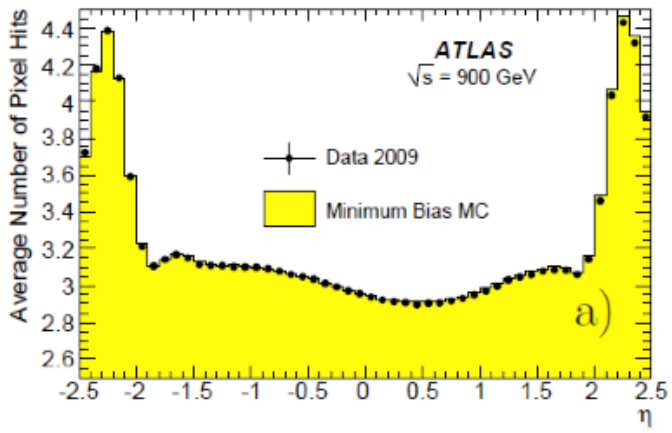
Применение GEANT4: ФВЭ

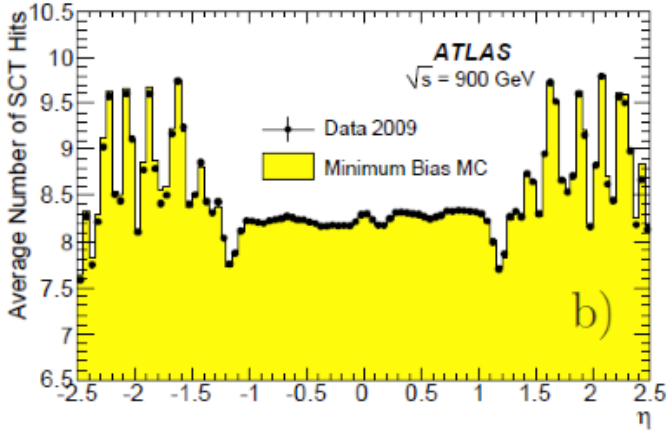


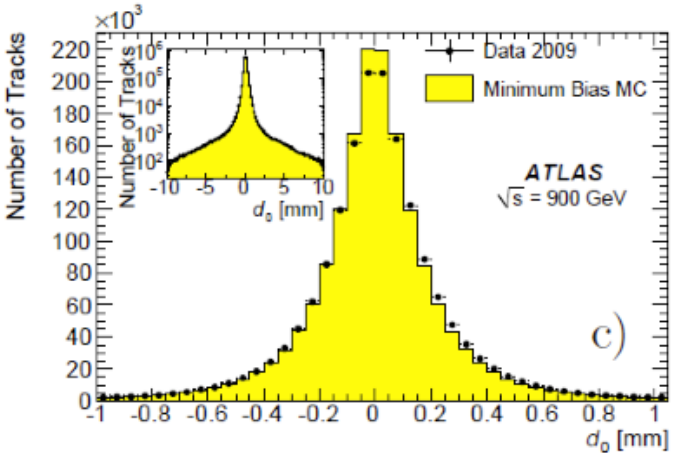
Применение GEANT4: сравнение с измерениями

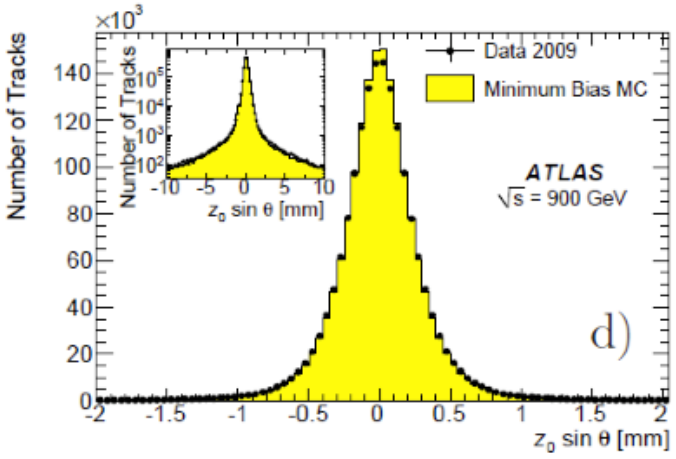
Очень важно добиться того, чтобы результаты моделирования согласовывались с измеряемыми параметрами (распределениями,...).

Благодаря совместной работе разработчиков и пользователей GEANT4, удается добиться отличного согласия.



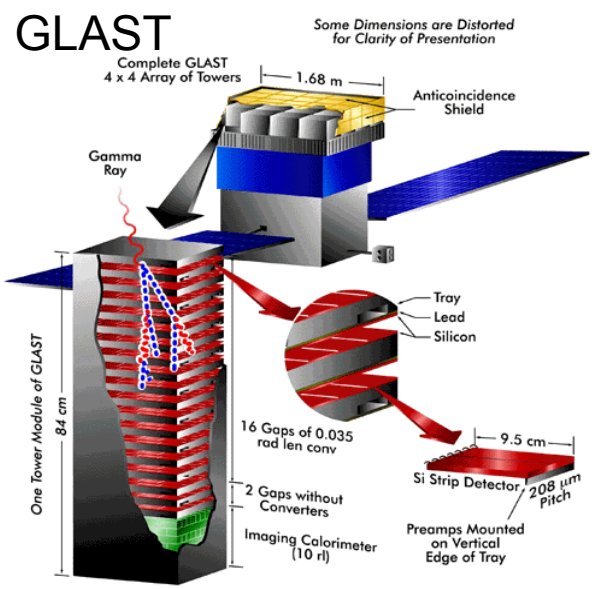


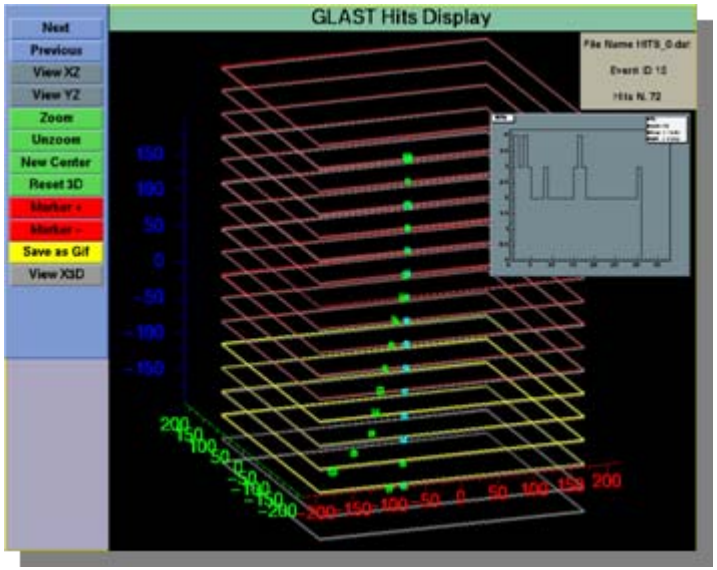


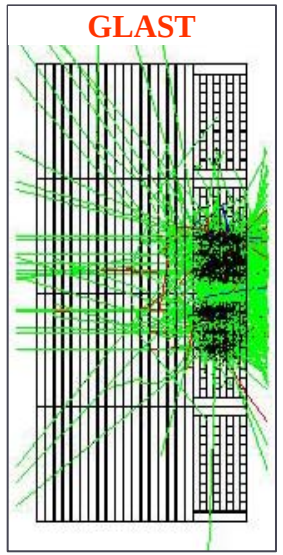


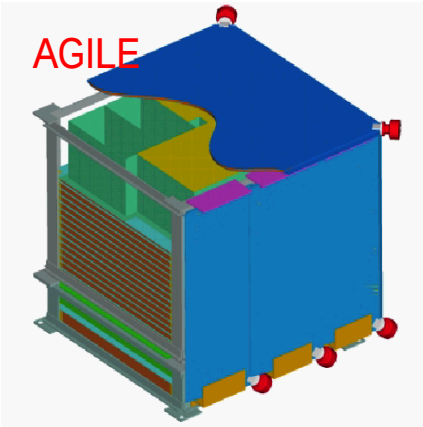
T. LeCompte (ANL)

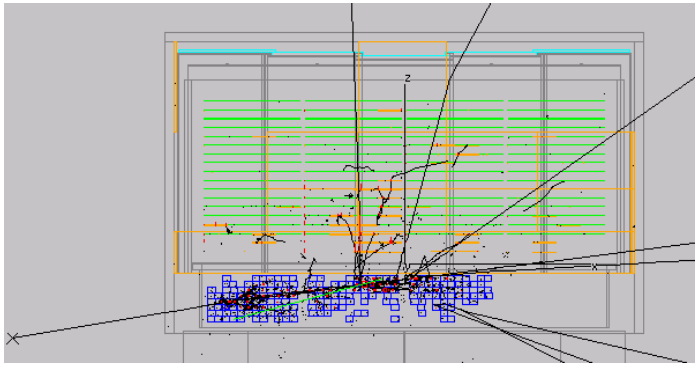
Применение GEANT4: γ-астрофизика









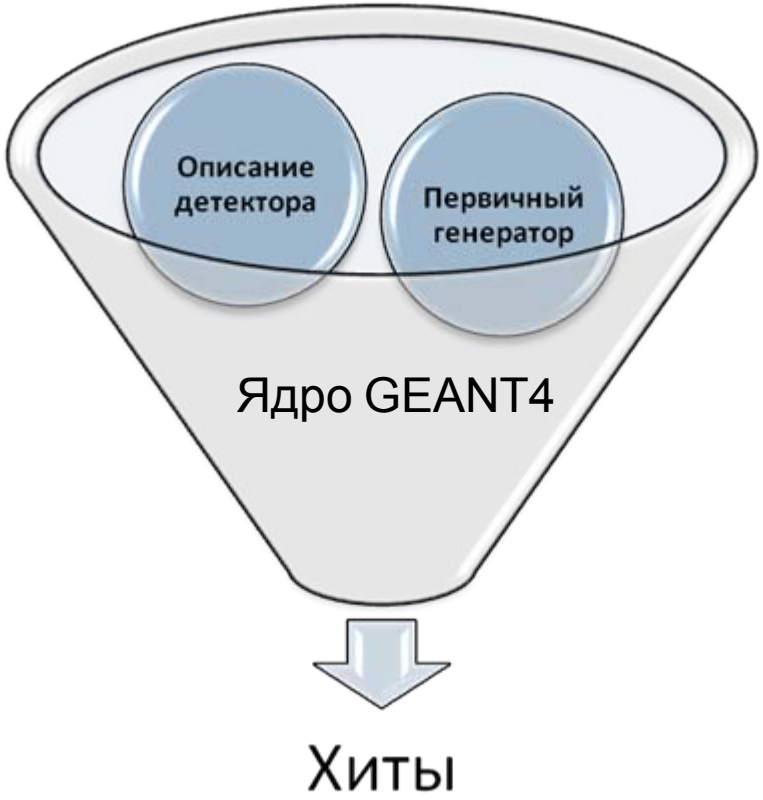


Toolkit

Geant4 – это не программа, а набор программных инструментов (toolkit), с помощью которых пользователь может создать программу моделирования.

Пользователь должен описать конструкцию детектора и сгенерировать первичные частицы. Ядро Geant4 моделирует взаимодействие этих частиц с детектором (включая моделирование вторичных частиц) и формирует хиты, описывающие единичные акты взаимодействия. Пользователь использует хиты при моделировании отклика электроники детектора.

На основе Geant4 разработано множество более специализированных программных каркасов.



Метод Монте-Карло

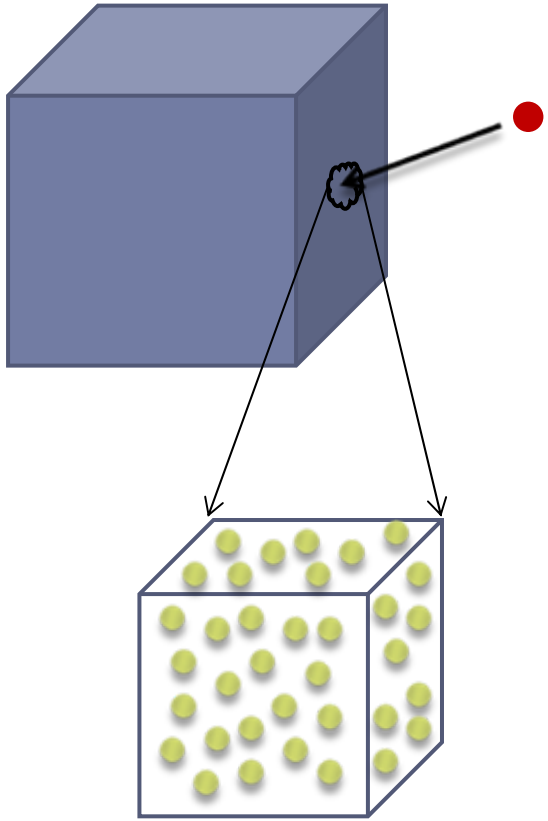
Частица взаимодействует не с веществом, а с отдельными атомами вещества.

В Geant4 предполагается, что центры взаимодействия распределены в пределах вещества равномерно и случайно, согласно известным макроскопическим параметрам вещества: химический и изотопный состав, плотность, фазовое состояние,...

Моделирование когерентных процессов (например, распространение холодных нейтронов) в Geant4 возможно, но требует дополнительный усилий.

Для расчетов используется метод Монте-Карло.

Известно, что этот метод является наиболее эффективным при расчете многомерных интегралов. А моделирование и является расчетом очень многомерного интеграла: $a = \int \varepsilon(\bar{x}) f(\bar{x}) d\bar{x}$.



Длина свободного пробега

Какова вероятность того, что частица пройдет расстояние L в веществе без взаимодействия?

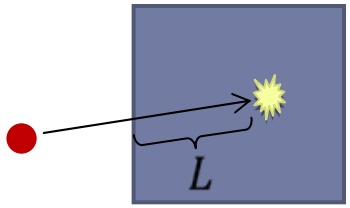
$$\mathrm{P}(L) = \exp(-n \cdot L \cdot \sigma) = \exp(-L/\lambda)$$, где

n – концентрация центров взаимодействия (молекул), σ – сечение взаимодействия («размер молекулы»), $\lambda = 1/n\sigma$ – длина свободного пробега.

Попробуйте вывести эту формулу, разбив вещество на бесконечное количество бесконечно тонких слоев.

Соответственно, вероятность того, что частица провзаимодействует, пройдя расстояние L:

$$p(L) = 1 - \mathrm{P(L)} = 1 - \exp(-L/\lambda)$$



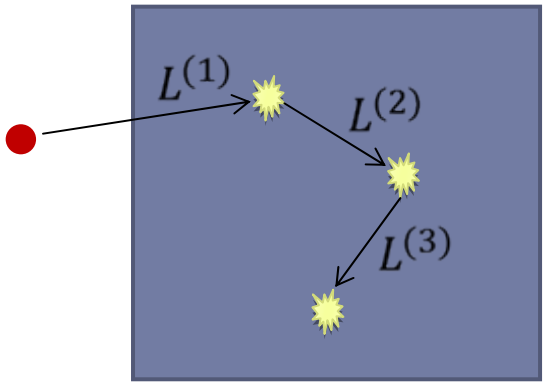
Замечание:

- n зависит только от макроскопических свойств вещества
- σ зависит от свойств самой частицы (тип, импульс,…) и микроскопических свойств вещества

Базовый алгоритм моделирования взаимодействия частицы с веществом

Пусть частица взаимодействует с веществом только одним образом.

Кроме того, будем считать это взаимодействие точечным – т.е. до взаимодействия частица проходит вещество без изменений, а в момент взаимодействия мгновенно изменяет свои параметры.



Алгоритм моделирования

Базовый алгоритм при наличии нескольких типов взаимодействия

Пусть возможны m разных вариантов точечного взаимодействия частицы с веществом. Тогда каждый тип взаимодействия описывается своей длиной свободного пробега λ_i $(i = 1 \dots m)$.

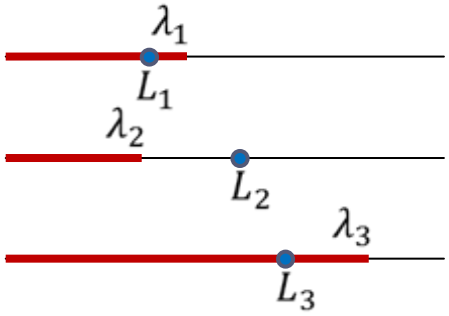
На каждом шаге моделирования выбирается тот тип взаимодействия, который произойдет первым (не обязательно у него самая маленькая λ).

Будет выбран этот процесс

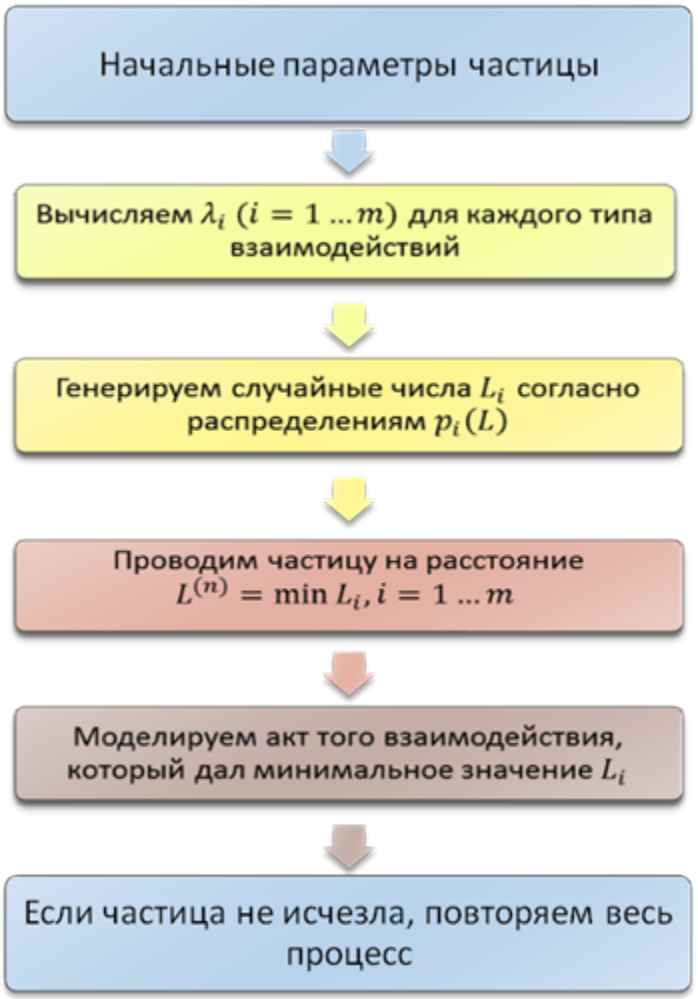
Комптоновское рассеяние

Рождение пары

Фотопоглощение



Алгоритм моделирования



Алгоритм моделирования в Geant4

В Geant4 используется описанный алгоритм со следующими дополнениями:

•Кроме точечных типов взаимодействия определены и непрерывные взаимодействия (например, ионизация).

•Для каждой частицы определяются два типа точечных взаимодействий: для движущейся и для покоящейся частицы

•Добавлены специальные типы «взаимодействий», которые не меняют параметры частицы, но ограничивают длину шага моделирования (например, не позволяют шагу переходить границу между разными веществами).

Пример: электромагнитные процессы в Geant4

Список электромагнитных процессов с момента первого выпуска Geant4 значительно расширился согласно потребностям пользователей:

•учет особенностей взаимодействий при высоких энергиях, для экспериментов на LHC с космическими лучами

• учет особенностей взаимодействий при низких энергиях, для медицинских, нейтринных и т.п. экспериментов

Для некоторых процессов существует несколько вариантов реализации.

С точки зрения ядра Geant4 все процессы – это классы, в которых реализованы интерфейсные функции: GetPhysicalInteractionLength() и DoIt().

Электромагнитные процессы

- Multiple scattering
- Bremsstrahlung
- Ionisation
- Annihilation
- Photoelectric effect
- Compton scattering
- Rayleigh effect
- g conversion
- e^+e^- pair production
- Synchrotron radiation
- Transition radiation
- Cherenkov
- Refraction
- Reflection
- Absorption
- Scintillation
- Fluorescence
- Auger

Ядро Geant4

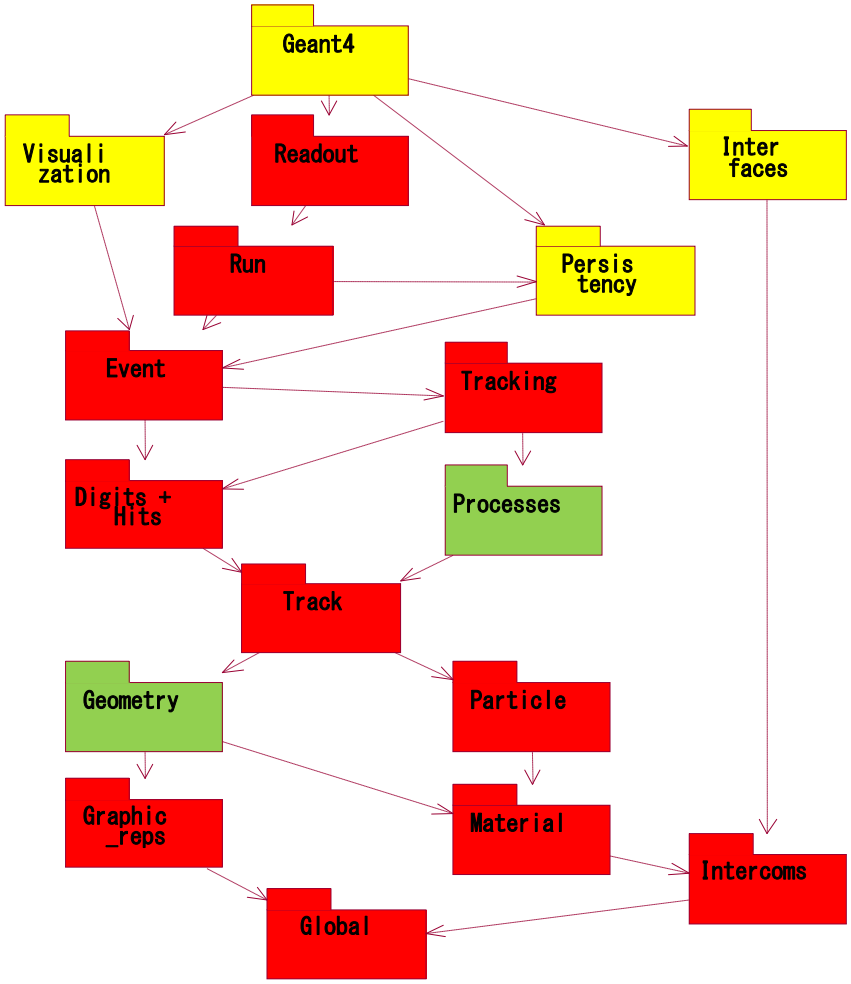
Программное ядро Geant4 состоит из 17 модулей, каждый из которых поддерживается отдельной группой. Специальная группа отвечает за межмодульный интерфейс.

Функции ядра Geant4:

•Управление заходами (runs), частицами (track, trajectory), шагами моделирования (steps), откликом детектора (hits, digits).

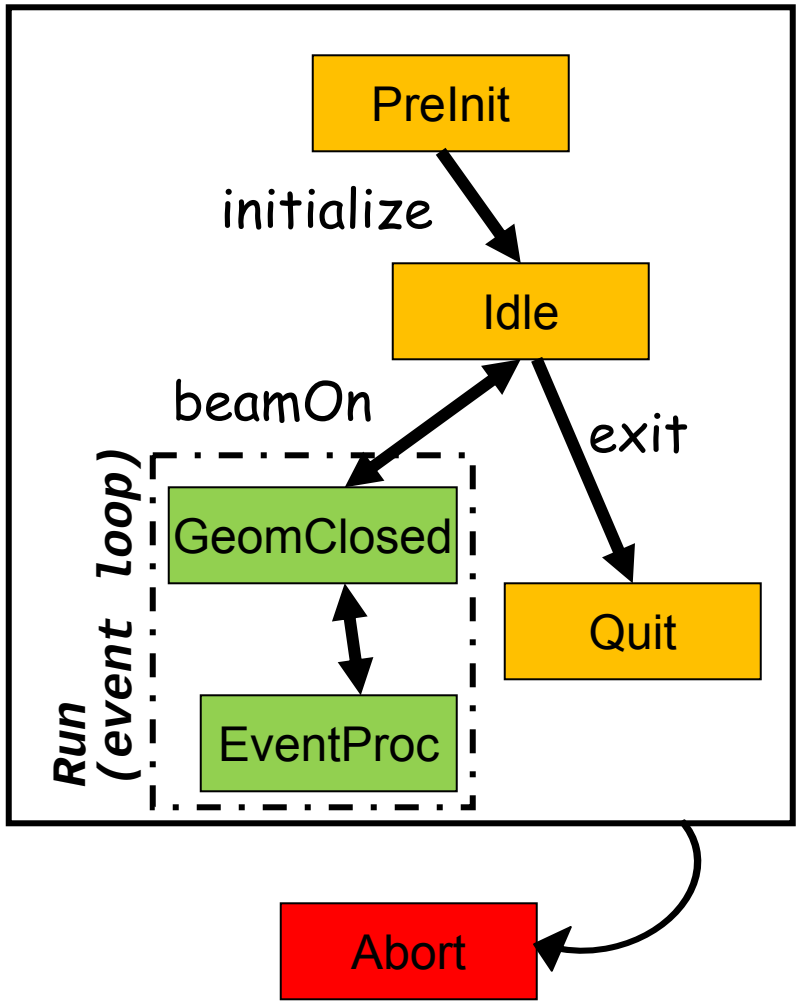
•Предоставление механизмов для работы с геометрическим представлением детектора.

•Предоставление механизмов для определения отдельных физических процессов.



Ядро Geant4 как конечный автомат

- G4State_PreInit
  Инициализация/определение вещества, геометрии, частиц, типов взаимодействия
- G4State_Idle
  Готовность к началу «захода»
- G4State_GeomClosed
  Оптимизация геометрии и готовность к моделированию события
- G4State_EventProc
  Моделирование одного события



Заход (Run)

Работа ядра Geant4 описывается иерархией заход-событие-трек-шаг.

Заход – множество событий, моделирование которых производилось в аналогичных условиях. При этом отдельные события в одном заходе статистически независимы.

В пределах одного захода пользователь не может:

- изменять геометрию детектора;
- изменять список или параметры возможных типов взаимодействий.

Понятие захода в Geant4 полностью аналогично понятию захода в эксперименте, означающего множество событий, зарегистрированных при одинаковых условиях набора данных.

В начале захода ядро Geant4 выполняет

Событие (Event)

Событие – коллекция всей информации, полученной при моделировании всех частиц, соответствующих одному запуску первичного генератора.

Процесс моделирования одного события:

1. Моделирование каждого события начинается с запуска первичного генератора, который создает первичные частицы и помещает их в стек.
2. Ядро Geant4 берет из стека по одной частице и моделирует ее прохождение через вещество детектора. Если в процессе моделирования появились вторичные частицы, они помещаются в стек.
3. Моделирование события заканчивается, когда в стеке не остается ни одной частицы.

Трек (Track)

Трек описывает мгновенное состояние одной частицы:

- в любой конкретный момент трек обладает рядом характеристик: положение, импульс,…;
- по мере движения частицы характеристики трека меняются;
- трек не является коллекцией шагов моделирования.

Трек создается или в первичном генераторе, или при рождении новой частицы в результате взаимодействия существующих частиц с веществом.

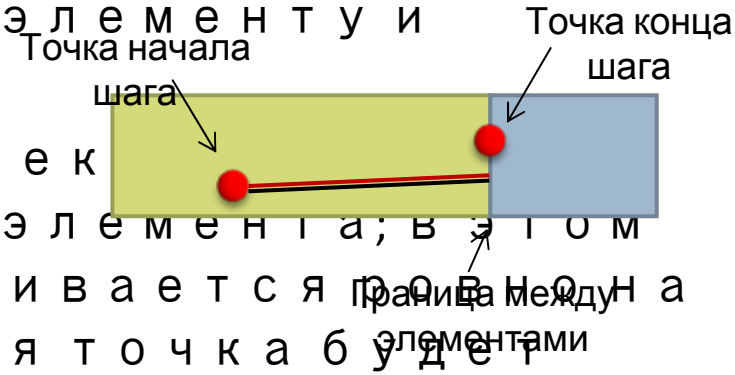
Трек удаляется в следующих случаях:

- частица вылетает из области моделирования (world volume);
- частица «умирает» (распадается,

Шаг моделирования (Step)

Минимальной единицей моделирования в Geant4 является шаг моделирования (Step):

- шаг описывается двумя точками: началом и концом
- шаг содержит всю информацию о том, что произошло с частицей на протяжении шага: энерговыделение, время пролета,...;
- точки начала и конца шага соответствуют некоторому геометрическому элементу и веществу;
- шаг не может пересекать границу геометрического элемента; в этом случае шаг заканчивается на границе и конечная точка будет соответствовать уже новому геометрическому элементу

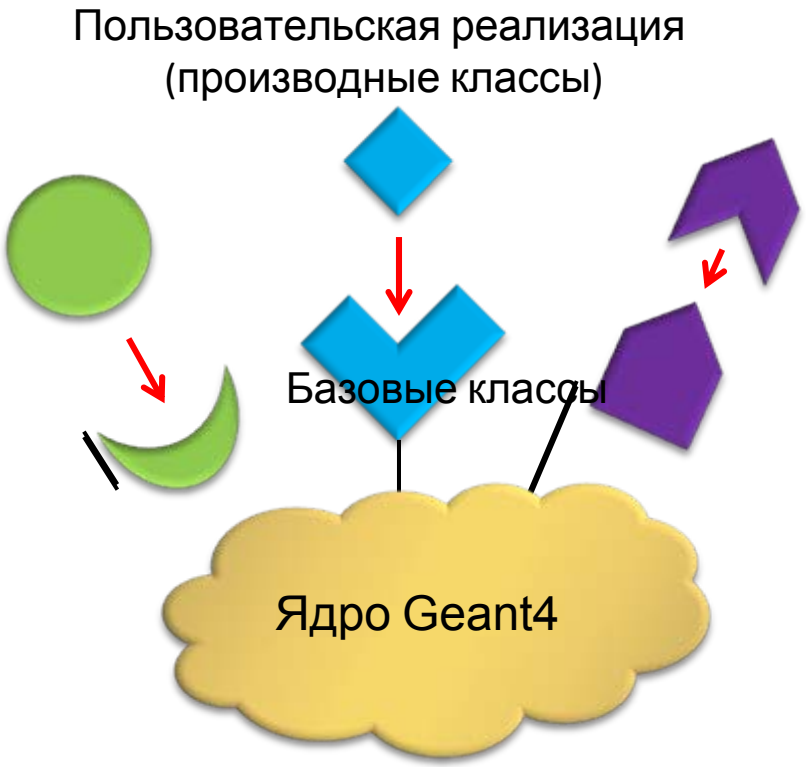


Благодаря тому, что шаг останавливается на границе, Geant4 позволяет моделировать такие взаимодействия, как преломление, переходное излучение,...

Взаимодействие пользователя и ядра Geant4

Для взаимодействия пользователя и ядра Geant4 используется механизм объектно-ориентированного программирования – наследование.

1. Все элементы моделирования, в которые может потребоваться вмешательство пользователя, выделены в виде базовых классов.
2. При необходимости, пользователь реализует производный класс, в который добавляет необходимую функциональность.
3. Ядро Geant4 работает с пользовательскими классами через интерфейс базового класса.
4. Те элементы, которые должны быть определены пользователем (например, описание геометрии детектора), выделены в ядре Geant4 в виде абстрактных базовых классов.



Основные базовые классы

Этап инициализации

Следующие классы позволяют управлять процессом инициализации:

- *G4VUserDetectorConstruction*
- *G4VUserPhysicsList*

Этап моделирования

Следующие классы позволяют управлять процессом моделирования:

- *G4VUserPrimaryGeneratorAction*
- G4UserRunAction
- G4UserEventAction
- G4UserTrackingAction
- G4UserStackingAction
- G4UserSteppingAction

Базовые классы, которые обязательно должны быть реализованы пользователем

G4VUserDetectorConstruction
описание геометрии детектора

G4VUserPhysicsList
перечисление возможных процессов взаимодействия

G4VUserPrimaryGeneratorAction
первичный генератор

Структура main()

Поскольку Geant4 – это библиотека (toolkit), пользователь сам должен написать основную программу m

Пользователь должен:

- создать G4RunManager
- передать ему пользовательские реализации базовых классов G4VUserDetectorConstruction, G4VUserPhysicsList, G4VUserPrimaryGeneratorAction
- запустить процесс моделирования

Пользователь может:

- передать G4RunManager пользовательские реализации других, необязательных, базовых классов
- создать графический или текстовый интерфейс

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

  // Set mandatory user initialization classes
  MyDetectorConstruction* detector = new MyDetectorConstruction;
  runManager->SetUserInitialization(detector);
  runManager->SetUserInitialization(new MyPhysicsList);

  // Set mandatory user action classes
  runManager->SetUserAction(new MyPrimaryGeneratorAction);

  // Set optional user action classes
  MyEventAction* eventAction = new MyEventAction();
  runManager->SetUserAction(eventAction);
  MyRunAction* runAction = new MyRunAction();
  runManager->SetUserAction(runAction);

  // Initialize G4 kernel and run one event
  runManager->Initialize();
  runManager->BeamOn(1);

 delete runManager();
}
```

Описание геометрии детектора (1)

Пользователь описывает геометрию детектора, реализуя класс, производный от абстрактного базового класса G4VUserDetectorConstruction. Пользователь должен реализовать функцию Construct(), в которой:

- определяются все используемые вещества;
- определяются все геометрические объемы, из которых состоит детектор;
- определяются и размещаются все элементы детектора;
- определяются чувствительные элементы детектора;
- формируется карта электромагнитных полей;
- определяются параметры визуализации элементов детектора.

File MyDetectorConstruction.hh:

```
class MyDetectorConstruction : public
G4VUserDetectorConstruction {
  public:
  …
  G4VPhysicalVolume* Construct();
  …
}
```

File MyDetectorConstruction.cc:

```
G4VPhysicalVolume* MyDetectorConstruction::Construct()
{
 // Implementation of the experimental set-up
 …
}
```

Пример описания используемых веществ

Вещество может быть определено как изотоп, химический элемент, молекула, или любая смесь перечисленных вариантов.

В Geant4 предопределено большое количество широко используемых веществ

```
PVPhysicalVolume*  MyDetectorConstruction::Construct()
{
  …
  // Define Xenon Gas

  density = 5.458*mg/cm3;
  pressure = 1*atmosphere;
  temperature = 293.15*kelvin;
  G4Material* xenon = new G4Material(name="XenonGas", z=54., a=131.29*g/mole, density,
                                                  kStateGas, temperature, pressure);

  // Define C9H10 scintillator

  G4Element* H  = new G4Element(name="Hydrogen", symbol="H", z=1., a=1.01*g/mole);
  G4Element* C  = new G4Element(name="Carbon", symbol="C", z=6., a=12.01*g/mole);

  G4Material* scintillator = new G4Material(name = "Scintillator", density = 1.032*g/cm3, numberOfComponents = 2);
  scintillator -> AddElement(C, numberOfAtoms = 9);
  scintillator -> AddElement(H, numberOfAtoms = 10);
  ...
}
```

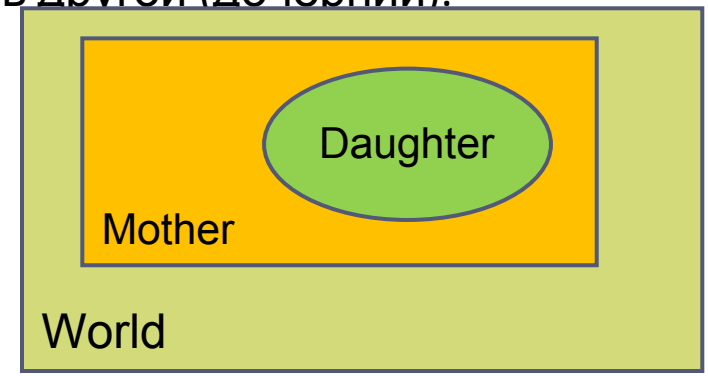
Описание геометрии детектора (2)

Описание геометрии элемента детектора производится на трех уровнях:

•описание геометрической формы и размера элемента – класс G4VSolid
•описание логического объема, для которого добавляется информация о веществе, из которого состоит элемент, наличии магнитного и электрического полей, пассивности или активности элемента,… – класс G4LogicalVolume
•описание физического объема, представляющего собой логический объем, помещенный в конкретную область пространства детектора – класс G4VPhysicalVolume

Элементы детектора не могут пересекаться частично, но один объем (материнский) может полностью содержать другой (дочерний).

Существует выделенный элемент детектора, world volume, который содержит все остальные элементы.



Пример описания геометрии детектора

геометрическая форма

логический объем

физический объем

материнский объем

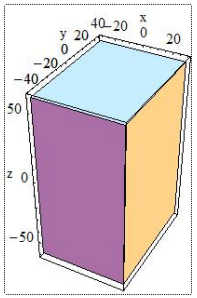
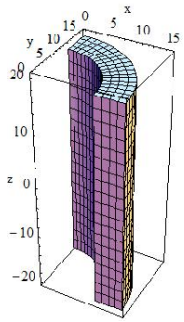
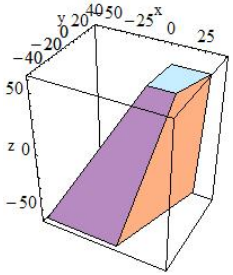
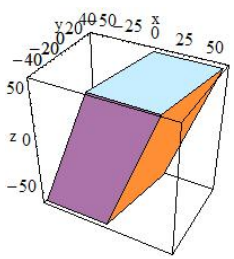
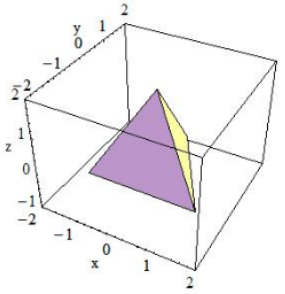
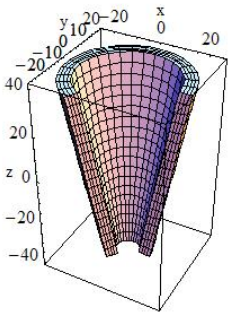
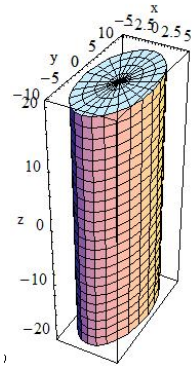
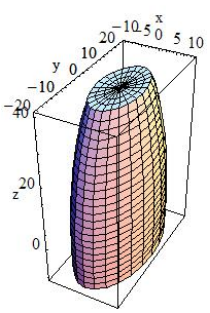
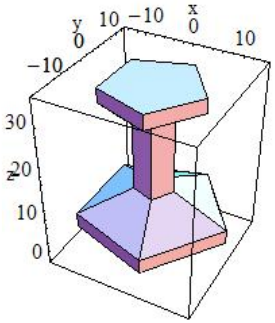
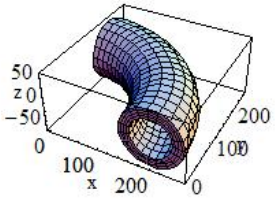
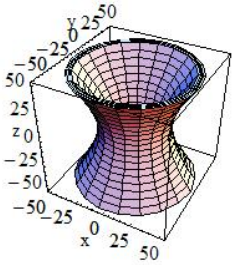
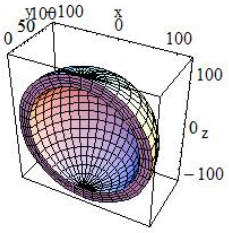
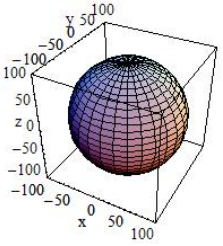
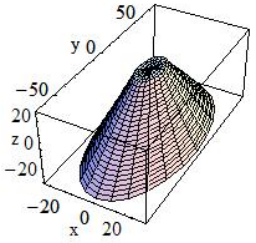
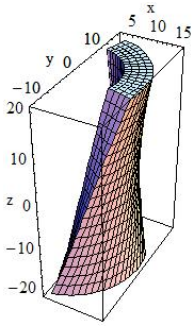
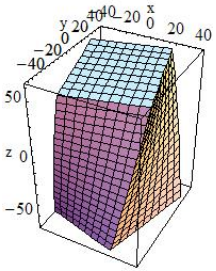
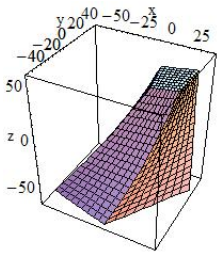
```
PVPhysicalVolume*  MyDetectorConstruction::Construct()
{
  …
 // Create world volume
 solidWorld = new G4Box(“World", halfWorldLength, halfWorldLength, halfWorldLength);
 logicWorld = new G4LogicalVolume(solidWorld, air, "World", 0, 0, 0);
 physicalWorld = new G4PVPlacement(0,                    //no rotation
                                   G4ThreeVector(),      // at (0,0,0)
                                   logicWorld,           // its logical volume
                                   "World",              // its name
                                   0,                    // its mother  volume
                                   false,                // no boolean operations
                                   0);                   // no magnetic field

 // Create target and place it to the world
 solidTarget = new G4Box(“Target", targetSize, targetSize, targetSize);
 logicTarget = new G4LogicalVolume(solidTarget, targetMaterial, "Target",0,0,0);
 physicalTarget = new G4PVPlacement(0,                   // no rotation
                                   positionTarget,       // at (x,y,z)
                                   logicTarget,          // its logical volume
                                   "Target",             // its name
                                   logicWorld,           // its mother  volume
                                   false,                // no boolean operations
                                   0);                   // no particular field

  ...
}
```

Примеры геометрических примитивов в Geant4

www.phys.nsu.ru

Перечисление списка возможных процессов взаимодействия

По умолчанию в Geant4 не определены никакие типы частиц и взаимодействий. Пользователь формирует списки частиц и перечисляет типы возможных взаимодействий, реализуя класс, производный от абстрактного базового класса G4VUserPhysicsList.

Пользователь должен реализовать функции ConstructParticles(), ConstructProcesses() и SetCuts().

Geant4 предоставляет множество уже готовых наборов частиц/процессов, которые можно использовать при реализации этих функций.

File MyPhysicsList.hh:

```
class MyPhysicsList : public G4VUserPhysicsList {
  public:
  …
  void ConstructParticles();
  void ConstructProcesses();
  void SetCuts();
  …
}
```

File MyPhysicsList.cc:

```
void MyPhysicsList::ConstructParticles()
{
 // Define all particles which can be created
 …
}
…
```

Пример перечисления списка возможных процессов

определение типов частиц

для каждого типа частицы необходимо перечислить, в каких взаимодействиях они могут участвовать

```
// Define all particles, which can occur in simulation
void  MyPhysicsList::ConstructParticles()
{
 G4Electron::ElectronDefinition();
 G4Positron::PositronDefinition();
 G4Gamma::GammaDefinition();
}

 // List, what can happen to these particles
void  MyPhysicsList::ConstructProcesses()
{
 …
 if (particleName == "gamma")
      {
                pManager->AddDiscreteProcess(new G4PhotoElectricEffect());
                pManager->AddDiscreteProcess(new G4ComptonScattering());
                pManager->AddDiscreteProcess(new G4GammaConversion());
      }
 else if (particleName == "e-")
      {
                pManager->AddProcess(new G4MultipleScattering(), -1, 1,1);
                pManager->AddProcess(new G4eIonisation(), -1, 2,2);
                pManager->AddProcess(new G4eBremsstrahlung(), -1,-1,3);
      }
  …
}
```

Первичный генератор

Первичный генератор формирует набор исходных частиц перед началом моделирования каждого события. Чтобы связать первичный генератор с ядром Geant4, пользователь реализует класс, производный от абстрактного базового класса G4VUserPrimaryGeneratorAction.

Пользователь должен реализовать функцию GeneratePrimaries(), в которой создать (положить в стек) все начальные частицы для данного события. Для каждой частицы задается ее тип, координаты и импульс (при необходимости – и другие параметры, например, спин).

File MyPrimaryGenerator.hh:

```
class MyPrimaryGenerator : public G4VUserPrimaryGeneratorAction
{
  public:
  …
  void GeneratePrimaries(G4Event*);
…
}
```

File MyPrimaryGenerator.cc:

```
void MyPrimaryGenerator::GeneratePrimaries()
{
 // Create primary particles
 …
}
…
```

Пример первичного генератора

Специальный инструмент для создания частиц

Указываем, что мы создаем электрон

Задаем параметры частицы

Создаем начальные частицы

```
MyPrimaryGenerator::MyPrimaryGenerator()
{

  G4int numberOfParticles = 1;
  particleGun = new G4ParticleGun (numberOfParticles);

  G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
  G4ParticleDefinition* particle = particleTable->FindParticle("e-");
  particleGun->SetParticleDefinition(particle);

  particleGun->SetParticlePosition(G4ThreeVector(x,y,z));
  particleGun->SetParticleMomentumDirection(G4ThreeVector(x,y,z));
  particleGun->SetParticleEnergy(energy);

}

void MyPrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
{
  particleGun->GeneratePrimaryVertex(anEvent);
}
```

Основные необязательные к реализации интерфейсные классы

Базовый Класс	Функция	Что позволяет сделать
G4UserRunAction	BeginOfRunAction(const G4Run*) EndOfRunAction(const G4Run*)	Выполнять действия по началу и концу захода, например, создавать и сохранять гистограммы
G4UserEventAction	BeginOfEventAction(const G4Event*) EndOfEventAction(const G4Event*)	Выполнять действия по началу и концу моделирования одного события, например, проводить анализ данных
G4UserTrackingAction	PreUserTrackingAction(const G4Track*) PostUserTrackingAction(const G4Track*)	Выполнять действия по началу и концу моделирования одной частицы, например, удалять неинтересные
G4UserSteppingAction	UserSteppingAction(const G4Step*)	Выполнять действия на каждом шаге моделирования, например, удалять частицы, сохранять промежуточные данные

Хиты

После того, как пользователь предоставил ядру перечисленные компоненты, Geant4 самостоятельно моделирует прохождение частиц через вещество детектора. Что делать с полученной информацией?

Geant4 предоставляет несколько механизмов сохранения результатов.

1. Используя UserSteppingAction(), самостоятельно сохранять всю интересующую информацию. Простой подход, но все требуется писать самому.
2. Использовать встроенный в Geant4 механизм сохранения результатов на сетке (scoring mesh). Удобный механизм для расчета поглощенной дозы, потоков частиц и т.п.
3. Определение активных областей детектора (sensitive detector) и самостоятельная обработка хитов. Наиболее гибкий механизм. Используется для детального моделирования, включая моделирование работы электроники.

Что почитать?

Geant4 – очень большой и сложный программный пакет. Для дальнейшего изучения Geant4 рекомендуется использовать следующие источники.

1.Описания на официальном сайте проекта
http://geant4.cern.ch/support/index.shtml

- Для начинающих пользователей: Application Developers' Guide
- Более детальные описания с техническими деталями: Toolkit Developers Guide, Physics Reference Manual

2.Программный код Geant4
http://www-geant4.kek.jp/LXR

3.Примеры, которые распространяются вместе с Geant4

- Большое число примеров, от простых до сложных, доступны как часть программного кода Geant4. Существуют примеры практически для каждой области применения пакета. Очень часто примеры используются как первоначальный каркас при разработке программы моделирования для нового эксперимента.