МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Факультет **ФИЗИЧЕСКИЙ**

Кафедра **ФИЗИКО-ТЕХНИЧЕСКАЯ ИНФОРМАТИКА**

Направление подготовки **03.06.01 ФИЗИКА И АСТРОНОМИЯ**

Образовательная программа **01.04.01 — ПРИБОРЫ И МЕТОДЫ ЭКСПЕРИМЕНТАЛЬНОЙ ФИЗИКИ**

НАУЧНЫЙ ДОКЛАД ОБ ОСНОВНЫХ РЕЗУЛЬТАТАХ ПОДГОТОВЛЕННОЙ НАУЧНО–КВАЛИФИКАЦИОННОЙ РАБОТЫ (ДИССЕРТАЦИИ) АСПИРАНТА

Жадан Анастасия Андреевна

Тема работы **Моделирование детектора для супер-чарм-тау фабрики**

«К защите допущена»	
Декан ФФ НГУ	**Научный руководитель**
д.ф.-м.н.	к.ф.-м.н.
	с.н.с. лаб. 3-0 ИЯФ СО РАН
Блинов В.Е. /_____	Сухарев А.М. /_____
«_____»_____2022	«_____»_____2022

Дата защиты«_____»_____2022

Новосибирск, 2022

Содержание

Основные обозначения 4

Введение 5

1 Супер–чарм–тау фабрика 7

1.1 Физическая программа 7

1.2 Детектор 8

2 Программное обеспечение для экспериментов ФВЭ 10

2.1 Современные программные средства ФВЭ 10

2.2 Программное обеспечение детектора СЧТФ 12

3 Геометрия детектора 14

3.1 Описание геометрии 14

3.1.1 Нестандартные геометрические формы 17

3.1.2 Тестовая геометрия 21

3.2 Валидация геометрии 21

3.2.1 MaterialScan 22

3.2.2 Geantino-сканирование 22

4 Моделирование 24

4.1 Чувствительные детекторы 24

4.2 User Actions 25

4.3 Модель данных 27

5 Визуализация детектора СЧТФ и событий в нём 30

5.1 GeoDisplay 30

5.2 EventDisplay 31

5.3 WebEventDisplay 35

5.3.1 Фреймворк Phoenix 35

5.3.2 Сервер 36

5.3.3 Клиент 37

5.3.4 Запуск веб-приложения 40

5.3.5 Планы развития WebEventDisplay 42

Заключение 43

Список литературы 44

Термины и обозначения

ФВЭ — физика высоких энергий
СЧТФ — супер-чарм-тау фабрика
Хит:

1. Взаимодействие частицы с веществом детектора в моделировании;
2. «Сырой» хит — срабатывание электроники детектора в результате взаимодействия, результат оцифровки;
3. Реконструированное срабатывание.

ПО — программное обеспечение
jobOption — файл задания параметров работы программы
Модель данных — унифицированное представление данных внутри различных компонент ПО ФВЭ

Введение

Одним из перспективных направлений работы ИЯФ СО РАН является проект супер-чарм-тау фабрики [1]. Супер–чарм–тау фабрика — это симметричный электрон–позитронный коллайдер с высокой светимостью, объём получаемых данных которого будет на два порядка превышать набор данных аналогичного эксперимента предыдущего поколения BESIII[2, 3]. Кроме того эта установка будет работать в широком диапазоне энергий, что предоставляет возможность иметь богатую физическую программу: изучение процессов рождения очарованных кварков и тау–лептонов, что позволит исследовать физические явления, не описываемые Стандартной Моделью.

Супер-чарм-тау фабрика будет оснащена универсальным магнитным детектором частиц[4]. Важнейшим этапом при разработке проекта детектора является его моделирование. Моделирование должно дать оценку ожидаемых параметров детектора и позволить выбрать его наиболее оптимальную конфигурацию, в том числе набор и тип регистрирующих подсистем.

Настоящая работа посвящена созданию средств для моделирования детектора супер-чарм-тау фабрики и проведения будущих экспериментов, что включает в себя программу полного моделирования и ряд вспомогательных программ, необходимых для завершения проекта детектора и, в дальнейшем, его постройки и эксплуатации.

Основной акцент работы сделан на следующих задачах различных направлений:

1. реализовать описание геометрии подсистем детектора по предоставленным специалистами предварительным вариантам описания подсистем детектора;

2. разработать и реализовать компоненты для описания регистрирующих объёмов детектора в моделировании;

3. реализовать инструменты визуализации геометрии детектора и событий в нём, полученных на различных этапах обработки данных.

В первой главе представлена краткая информация по физической программе проекта СЧТФ и подсистемам детектора, во второй главе рассмот-

рено существующее ПО, используемое в экспериментах по ФВЭ. Третья глава включает в себя описание геометрии подсистем детектора, четвёртая глава посвящена компонентам моделирования. Наконец, в пятой главе описаны созданные инструменты визуализации.

1 Супер–чарм–тау фабрика

1.1 Физическая программа

Основными отличительными характеристиками коллайдера являются высокая светимость 10^{35} см$^{-2}$с$^{-1}$ и широкий диапазон энергий от 2 до 6 ГэВ в системе центра масс. Такой диапазон энергий позволяет изучать рождение частиц, принадлежащих семейству чармониев, очарованных мезонов и барионов, тау-лептонов.

Предполагаемое распределение интегральной светимости за один экспериментальный сезон (10^7с), и энергия супер-чарм-тау фабрики приведены в Таблице 1.

E, ГэВ	L, фб$^{-1}$		
3,097	300	J/ψ	Спектроскопия состояний из легких кварков, редкие распады
3,554	50	порог $e^+e^- \to \tau^+\tau^-$	Прецизионное измерение распадов τ-лептонов
3,686	150	ψ(2S)	Спектроскопия состояний из легких кварков, спектроскопия чармония
3,770	300	ψ(3770)	Исследование свойств D-мезонов
4,170	100	ψ(4160)	Исследование свойств D_s -мезонов
4,650	100	максимум $\sigma(e^+e^- \to \Lambda_c^+\Lambda_c^-)$	Исследование свойств Λ_c -барионов

Таблица 1: Энергия работы СЧТФ и предполагаемая интегральная светимость за один экспериментальный сезон.

Такая большая статистика данных позволяет иметь обширную программу по изучению физики

- чармония и чармониеподобных состояний;

- состояний из легких кварков;
- очарованных мезонов;
- очарованных барионов;
- τ-лептона,

а также двухфотонной физики и измерения сечения процесса $e^+e^- \to$ адроны.

1.2 Детектор

Детектор супер-чарм-тау фабрики будет включать в себя стандартный набор элементов и подсистем (Рис. 1):

1. вакуумная камера, в центре которой будут сталкиваться пучки;
2. финальный фокус — система фокусировки и направления пучков непосредственно перед местом столкновения. Является частью ускорителя, глубоко встроен в детектор;
3. внутренний трековый детектор для восстановления исходной вершины события;
4. дрейфовая камера — основной детектор треков, служит для измерения импульса частиц и идентификации частиц по потере энергии;
5. система идентификации частиц — детектор черенковского излучения;
6. электромагнитный калориметр для регистрации и измерения энергии заряженных частиц и фотонов;
7. сверхпроводящая катушка для создания аксиального магнитного поля в детекторе порядка 1,5 Тл, позволяющее измерять импульс заряженных частиц;
8. ярмо магнита с расположенной в его зазорах мюонной системой для разделения мюонов и адронов, а также отделения космических мюонов.

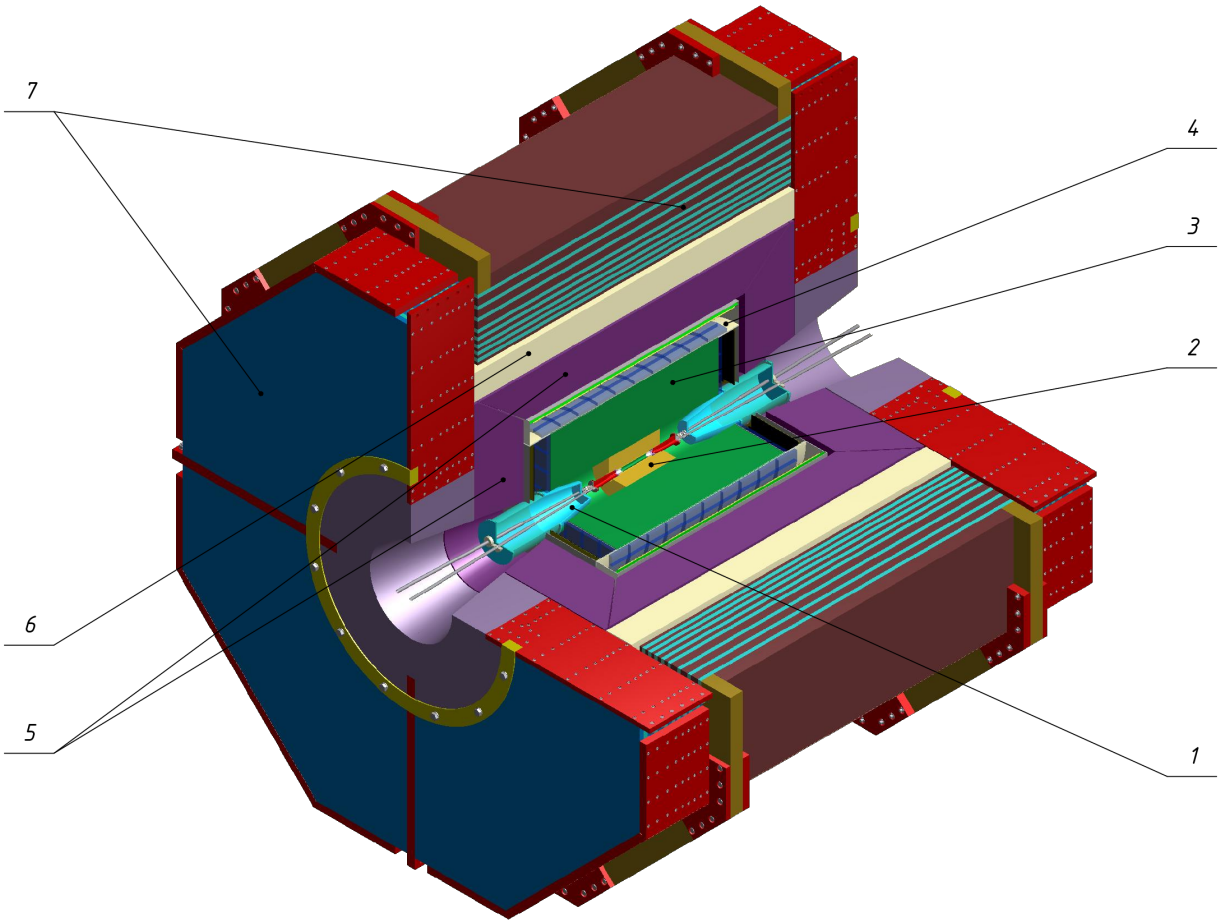


Рис. 1: Подсистемы детектора СЧТФ: 1 — вакуумная камера и финальный фокус, 2 — внутренний трековый детектор, 3 — дрейфовая камера, 4 — система идентификации частиц, 5 — калориметр, 6 — сверхпроводящая катушка, 7 — мюонная система и ярмо магнита.

2 Программное обеспечение для экспериментов ФВЭ

2.1 Современные программные средства ФВЭ

Программное обеспечение современных экспериментов по ФВЭ, как правило, организуется с использованием программных сред — фреймфорков, для решения универсальным образом типовых задач, таких, как чтение и запись данных на различных стадиях обработки, реализация описания геометрии детектора, которое должно быть одинаковым в моделировании и при процессе реконструкции, разделение данных и алгоритмов и т. д. Существует множество программных пакетов и библиотек, предоставляющее богатый инструментарий для решения перечисленных задач.

Рассмотрим существующее программное обеспечение, используемое в нашем проекте.

ROOT

ROOT[5, 6] — это фреймворк, используемый для хранения и анализа данных, размер которых может достигать петабайт. ROOT позволяет сохранять данные, включая любой C++ объект, в файл в сжатом бинарном виде, совершать математическую и статистическую обработку содержимого файлов, визуализировать данные, например, отображать геометрические объекты и строить гистограммы. ROOT применяется в большинстве исследований ФВЭ.

DD4hep

DD4hep[7] — фреймворк, предоставляющий инструментарий для полного описания геометрии детектора, используемых материалов, визуализации и т. д. Используя DD4hep, можно обеспечить единственный источник информации о геометрии детектора, применяемой для моделирования, визуализации, реконструкции и т. д.[8]. В DD4hep используются и расширяются существующие геометрические пакеты ROOT, поддерживается интеграция с Geant4 и Gaudi[9]. Описание геометрии детектора является важным элементом ПО в экспериментах ФВЭ, так как от него зависят все последующие полученные результаты моделирования, реконструкции и анализа.

Geant4

Geant4[10] — это программный пакет, включающий в себя инструменты для моделирования взаимодействия элементарных частиц с веществом, внутри которого они движутся, на основе метода Монте-Карло. Этот пакет предоставляет широкий функционал от генерации частиц до хранения и анализа данных, содержит большой набор физических моделей, описывающих бóльшую часть известных взаимодействий частиц с веществом, и который продолжает расширяться, развивается и поддерживается международным сообществом физиков и программистов[11]. Области применения Geant4 включают в себя эксперименты и проекты ФВЭ, физику ускорителей, ядерную физику, а также космонавтику и медицину[12].

Gaudi

Gaudi[17, 18] — фреймворк, используемый для конфигурации приложений для обработки данных на различных этапах (моделирование, оцифровка и т. д.) для экспериментов по ФВЭ. Он предоставляет для этого все необходимые компоненты и интерфейсы, которые позволяют простым способом настроить фреймворк и описать алгоритмы обработки в зависимости от данных и эксперимента. Gaudi много лет используется в ПО таких экспериментов, как ATLAS[13, 14] и LHCb[15, 16].

Алгоритмы (Algorithms) Gaudi используются для обработки событий последовательно одно за другим, создавая и обрабатывая объекты, хранящиеся в обменнике (Transient Event Store, TES). Обменник содержит данные, которые доступны из любой части фреймворка. При этом хранящиеся в нём данные являются неизменяемыми. Обмен данными между алгоритмами осуществляется через это хранилище. С помощью сервисов (Services) осуществляется доступ к различным частям фреймворка, например, ToolSvc предоставляет доступ к инструментам (Tool). Инструменты используются в качестве подалгоритмов, вызов которых можно делать для обработки только определённых событий, подходящих под некоторые условия, следовательно, можно создавать только необходимые в этой ситуации инструменты. За управление перечисленными компонентами отвечает менеджер приложения (Application Manager).

PODIO

PODIO[19] — C++ библиотека для создания и управления моделью данных событий в ФВЭ. PODIO обеспечивает высокую производительность при работе с данными за счёт использования простых структур данных (plain-old-data) и отсутствия глубокой иерархии объектов и виртуального наследования[20]. Генерация классов модели данных делается на основе их описания в yaml-файле. PODIO используется при разработке ПО Future Circular Collider (FCC)[21, 22].

2.2 Программное обеспечение детектора СЧТФ

Программное обеспечение детектора СЧТФ организовано с применением фреймворка, который называется Aurora[27, 28]. Для ускорения процесса разработки фреймворка используются программное обеспечение и пакеты, уже разработанные исследовательскими группами международного научного сообщества, и которые в данный момент являются практически стандартными для решения некоторых задач, так и новые, активно развивающиеся и поддерживаемые проекты.

Фреймворк Aurora основан на фреймворке Gaudi. DD4Hep используется для описания геометрии детектора. Geant4 применяется для моделирования процессов, происходящих в детекторе при прохождении частицы через него. ROOT используется для хранения данных, их визуализации и для физического анализа. Основа для некоторых пакетов заимствована из ПО FCC[23], анализ частично заимствован из ПО Belle II[24, 25, 26]. Для генерации событий используется большой набор пакетов: EvtGen[29, 30], Taulola[31, 32], PHOTOS[33, 34], Pythia[35, 36], MCGPJ[37], BabaYaga[38, 39], BBBREM[40], KKMC[41] и др.

Aurora обеспечивает взаимодействие всех компонент ПО. Входные данные обрабатываются алгоритмом, генерируя на выходе новые данные, которые могут быть далее переданы следующему алгоритму, формируя таким образом цепочку обработки. При этом данные могут быть сохранены на любом из шагов цепочки для промежуточного анализа в ROOT-файл в виде ROOT-коллекции (далее — коллекции). Взаимодействие всех компонент программного обеспечения детектора представлено на Рис. 2.

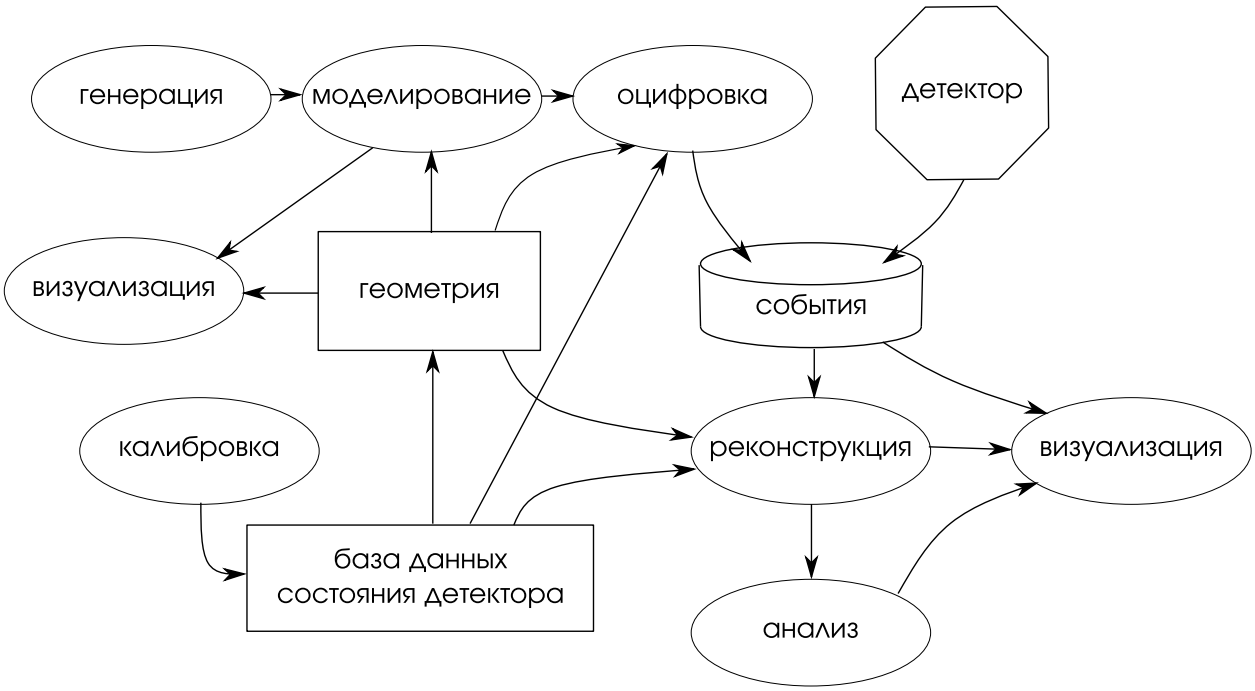


Рис. 2: Взаимодействие компонент ПО детектора.

Полная последовательность обработки данных следующая: сначала события генерируются заданным в настройках генератором, затем полученные данные передаются в моделирование. Результаты моделирования — наборы смоделированных хитов — обрабатываются модулем оцифровки, выдавая данные, эквивалентные тем, что будут считываться с электроники детектора. Такие данные называются «сырые» хиты. Они передаются на вход алгоритмам реконструкции для восстановления данных об исходном событии. Алгоритм реконструкции должен быть способен принимать на вход как смоделированные, так в будущем и реальные данные, записанные с детектора.

Используется операционная система Scientific Linux 7 с архитектурой x86_64, основной компилятор GCC9, для конфигурации и скриптов используется Python3. Система сборки пакетов Aurora и конфигурирования рабочей среды были сделаны на основе аналогичной части ПО эксперимента ATLAS. Система сборки внешних пакетов ПО, как специализированных средств ФВЭ (см. Раздел 2.1), так и пакетов общего назначения, версии которых могут отличаться от системных, использует LCGCMake[42].

3 Геометрия детектора

3.1 Описание геометрии

Описание геометрии каждой подсистемы детектора представляет собой отдельный пакет в фреймворке Aurora. В XML-файлах содержатся параметры и константы, используемые для построения геометрии. Важно, чтобы при построении геометрии формировалась правильная вложенность объёмов, то есть их иерархия. Логика взаимного расположения реализована на C++.

Структура пакета с описанием геометрии подсистемы:

- директория python, в которой расположен скрипт, возвращающий имя XML-файла Имя_подсистемыGeom_def.xml с параметрами геометрии, например, DriftChamberGeom_def.xml;
- директория share. В данной директории расположены файлы, используемые для автоматического тестирования геометрии. Их содержимое сравнивается с выходом утилиты MaterialScan (см. Подраздел 3.2.1). Данные тесты предназначены для отслеживания изменений в описании геометрии подсистемы, например, сдвигов, изменений материалов и т. д., как преднамеренных, так и возникших в результате ошибки;
- директория src. В ней расположен упомянутый выше C++ файл с реализацией построения геометрии с правильной иерархией объёмов;
- директория xml. Содержит два XML-файла:
  - Имя_подсистемыGeom_def.xml. Этот файл содержит константы для построения геометрии, параметры для её отображения, структуру для кодирования идентификаторов объёмов и ссылку на файл Имя_подсистемыGeom_geo.xml, расположенный в этой же директории;
  - Имя_подсистемыGeom_geo.xml содержит описание геометрии на основе данных из файла Имя_подсистемыGeom_def.xml: форма, расположение, материалы объёмов, является ли объём чув-

ствительным и т. д. Также указаны другие вспомогательные константы, которые используются для построения геометрии на C++.

- CMakeLists.txt — файл с инструкциями для сборки пакета с помощью cmake.

Вновь добавленные и уже существовавшие описания геометрии приведены в соответствие с данной структурой.

На следующих рисунках представлено отображение геометрии в утилите GeoDisplay (см. Раздел 5.1) для финального фокуса (Рис. 3) и подсистем детектора:

- вакуумная камера (Рис. 4);
- внутренний трековый детектор TPC — время-проекционная камера — один из вариантов внутреннего трекового детектора (Рис. 5);
- простая дрейфовая камера (Рис. 6);
- система идентификации частиц ФАРИЧ (Рис. 7);
- калориметр (Рис. 8). Описание реализовано экспертом подсистемы;
- сверхпроводящая катушка (Рис. 9).
- мюонная система и ярмо (Рис. 10).

В рамках данной работы было реализовано описание геометрии вакуумной камеры, финального фокуса, дрейфовой камеры и катушки, обновлено описание геометрии для TPC в соответствии с описанной выше структурой. Был сделан шаблон и подробная инструкция для описания геометрии ещё одного варианта внутреннего трекового детектора — CmuRWELL, разрабатываемого для детектора СЧТФ исследовательской группой INFN (Италия).

Кроме того, для катушки и дрейфовой камеры было добавлено описание используемых материалов. Для катушки это пропорциональная смесь стабилизатора из сплава алюминия и проводников из сплава ниобий-титан, представляющая собой в первом приближении материал сверхпроводящего

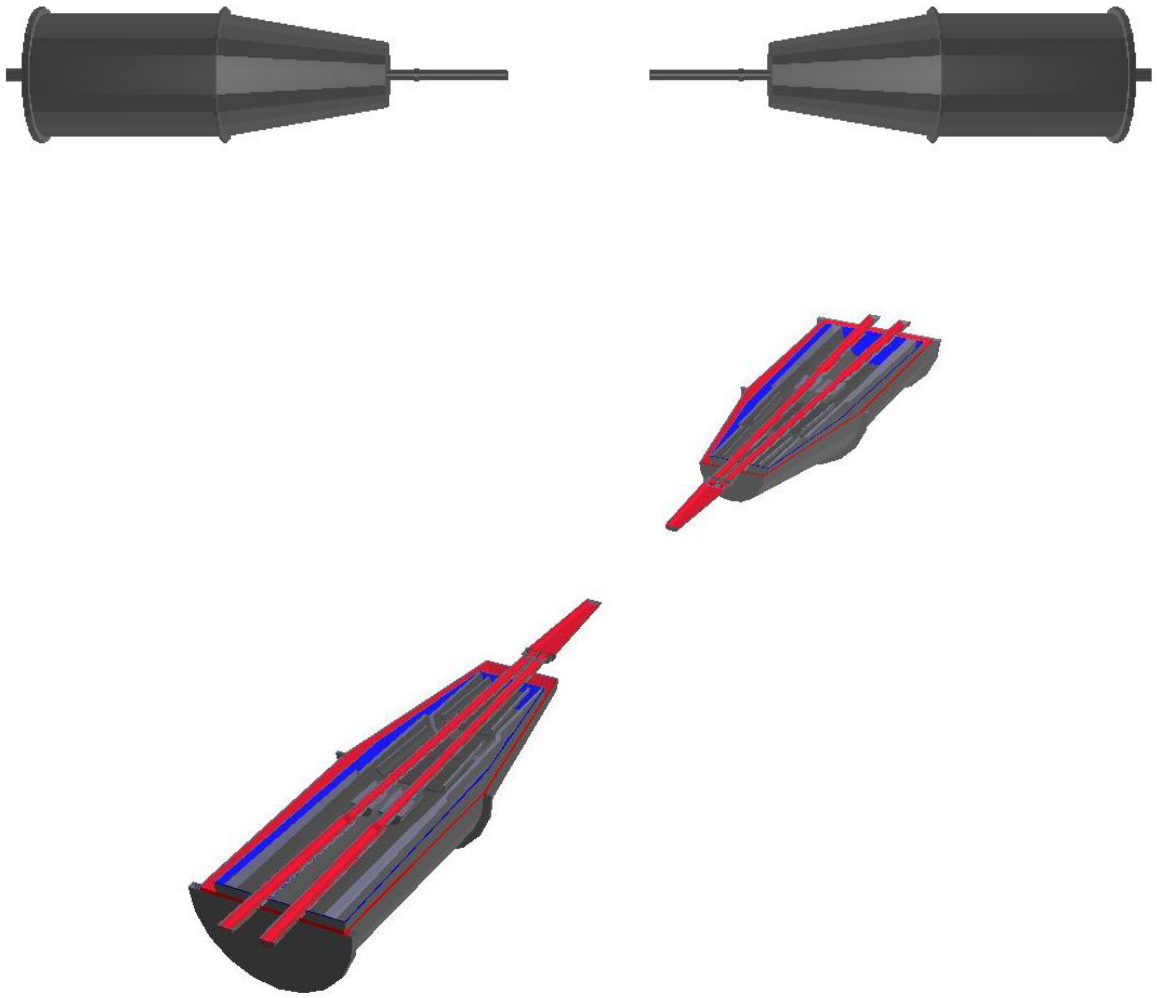


Рис. 3: Геометрия финального фокуса.

кабеля. Для дрейфовой камеры это материал, который равномерно распределён по всему внутреннему объёму, соответствуя пропорциональной смеси материала проволочной структуры: анодные проволочки из вольфрамрениевого сплава, полевые проволочки из алюминия, пины крепления из золочёного алюминия; и газовой смеси гелий-изобутана в пропорции 80/20.

Для каждой подсистемы детектора есть как минимум один вариант реализации. Часть подсистем имеет альтернативные варианты, оценка которых будет сделана на основе анализа результатов моделирования. Решение о том, какой вариант подсистемы будет реализован в окончательной версии проекта детектора, будет основываться на оценке соотношения трудозатрат и эффективности.

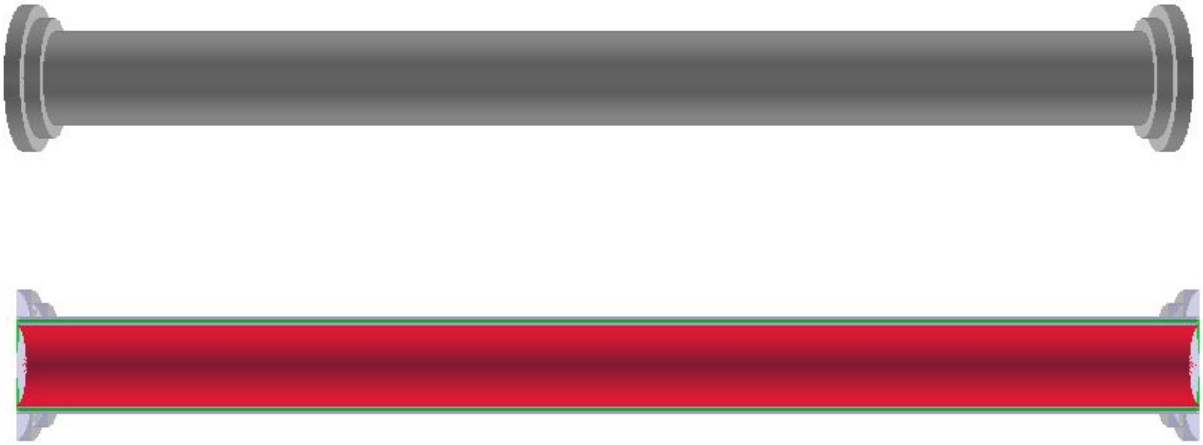


Рис. 4: Геометрия вакуумной камеры.

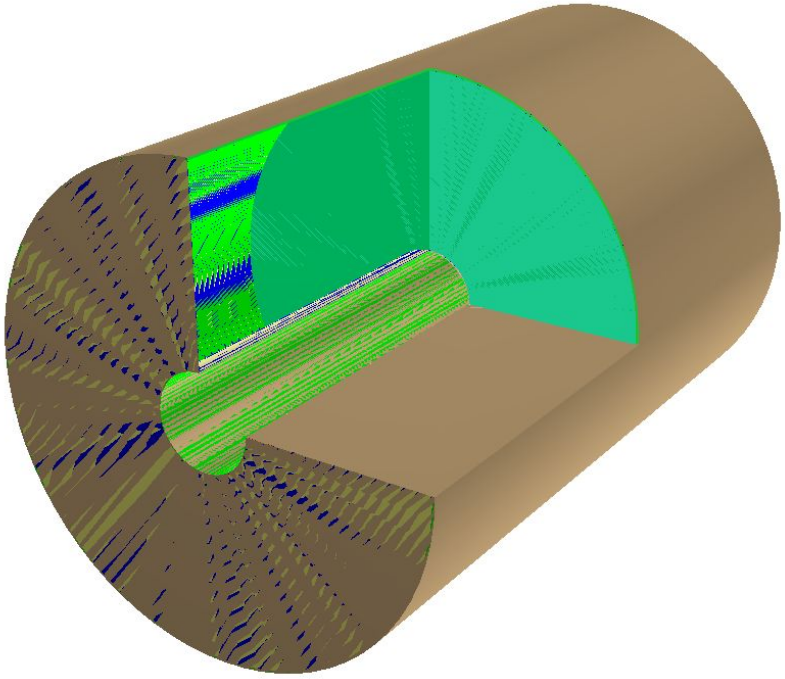


Рис. 5: Геометрия внутреннего трекового детектора в варианте TPC.

3.1.1 Нестандартные геометрические формы

В DD4Hep отсутствуют необходимые для описания геометрические фигуры: наклонный Polycone и пирамида с закруглёнными боковыми сторонами.

Наклонный Polycone — `Sctau_RotatedPolycone` — был реализован на основе базового Polycone. Параметры для `Sctau_RotatedPolycone`, задаваемые в XML-файле, аналогичны параметрам Polycone, но теперь под-

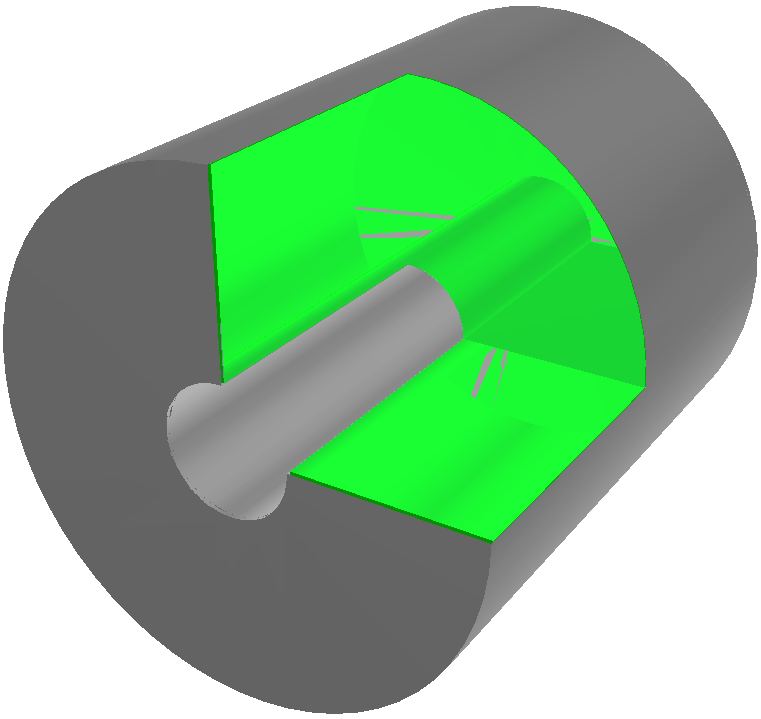


Рис. 6: Геометрия дрейфовой камеры.

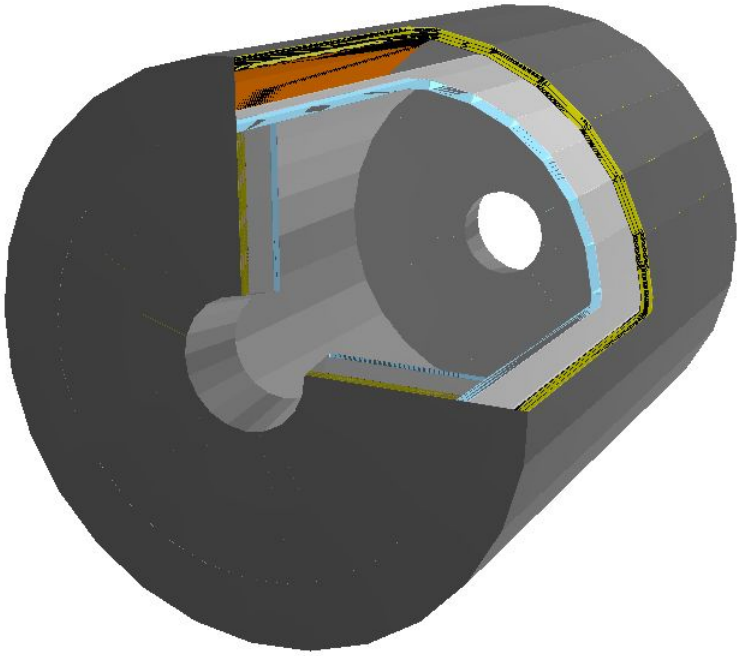


Рис. 7: Геометрия системы идентификации частиц в варианте ФАРИЧ.

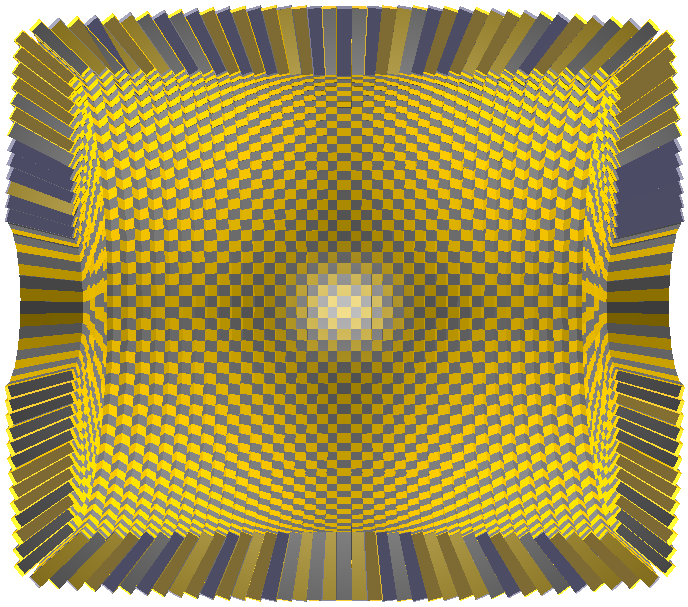


Рис. 8: Геометрия калориметра.

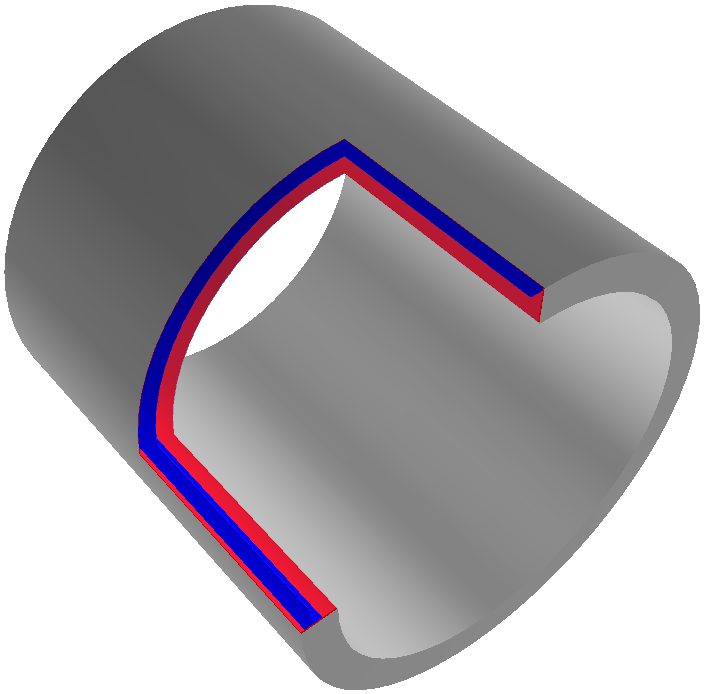


Рис. 9: Геометрия сверхпроводящей катушки.

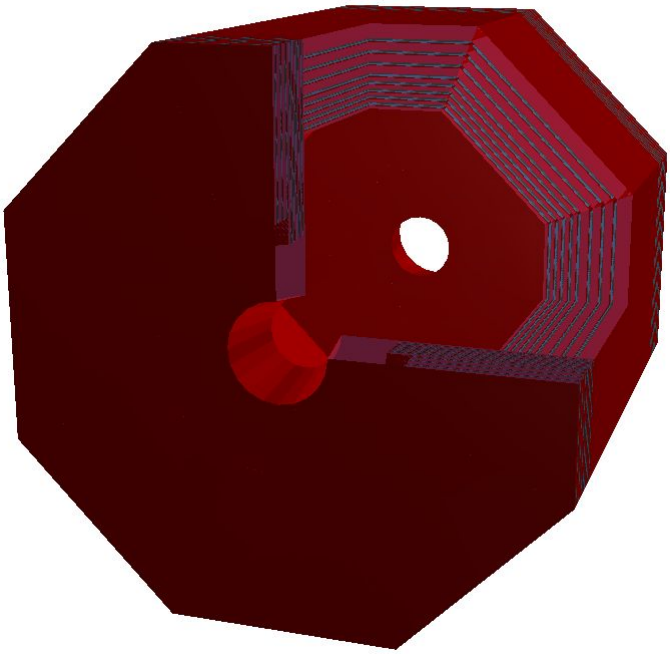


Рис. 10: Геометрия мюонной системы и ярма.

держивается вращение. Эта фигура используется в описания геометрии финального фокуса для описания каналов, по которым движутся пучки к месту столкновения.

Пирамида с закруглёнными боковыми сторонами — `Sctau_RoundTrap` — реализована путём объединения трапеции и двух трубок по бокам трапеции. При этом длина трубок превышает длину боковой стороны трапеции. Затем берётся пересечение получившегося объёма с коробкой, толщина и высота которой равны толщине и высоте трапеции соответственно, длина равна полуторной длине нижнего основания. Увеличенная длина необходима для сохранения закруглений на нижнем основании. Обрезание объёма такой коробкой даёт нам ровные плоскости на верхнем и нижнем основаниях трапеции. Для описания данной фигуры в XML-файле необходимо задать толщину, высоту и длины верхнего и нижнего оснований. В геометрии финального фокуса трапеция с закруглёнными боковыми сторонами используется для описания компенсирующих соленоидов. Пример данной формы — адаптер финального фокуса (Рис. 11).

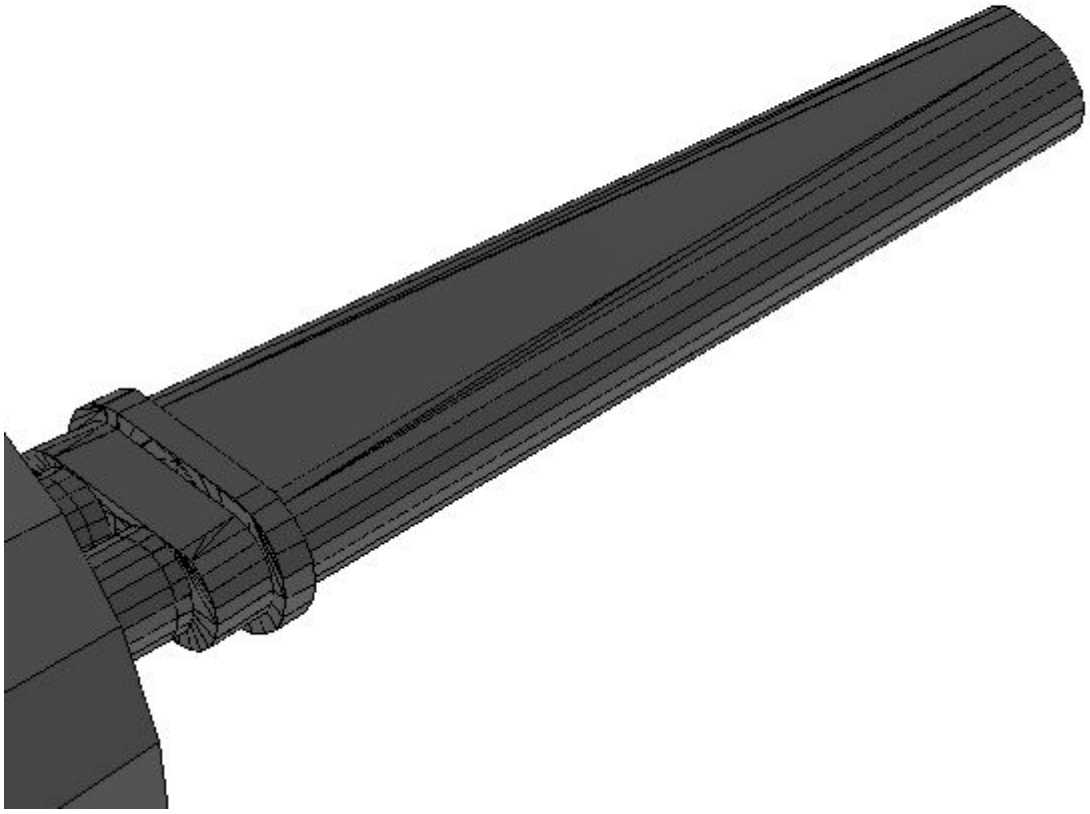


Рис. 11: Пирамида с закруглёнными боковыми сторонами.

3.1.2 Тестовая геометрия

При переходе на одну из новых версий ROOT утилита EventDisplay (см. Раздел 5.2) не заканчивала работу корректно. В процессе исследования было обнаружено, что причиной было присутствие в отображении подсистем, геометрии которых содержат геометрические формы Polycone и Polyhedra. Кроме того, можно было увидеть артефакты при отображении геометрий этих подсистем в утилите GeoDisplay (см. Раздел 5.1). Для понимания этой проблемы были разработаны пакеты тестовой геометрии на основе Polycone и Polyhedra — `TestPolycone` и `TestPolyhedra`.

3.2 Валидация геометрии

Валидация геометрии проводится на двух уровнях: DD4hep и Geant4. Делается это с целью убедиться, что геометрия детектора, описанная в

DD4hep, правильно конвертируется в геометрические объекты Geant4. В этом случае можно гарантировать, что геометрия одинаковая, и моделирование событий будет верным.

3.2.1 MaterialScan

Проверка геометрии на уровне DD4hep проведена с использованием модифицированной DD4Hep утилиты MaterialScan, которая печатает материалы и их толщины при движении из заданной начальной точки в определённом направлении. Как было сказано ранее, выходные данные этой утилиты используются для автоматического тестирования геометрии. Пример печати утилиты в консоль представлен на Рис. 12.

3.2.2 Geantino-сканирование

Для валидации геометрии на уровне Geant4 было реализовано geantino-сканирование. Geantino — специальная частица Geant4. Она не имеет физических аналогов, не участвует в физических взаимодействиях и служит только для проверки геометрии детектора в Geant4. Результат сканирования отображается пошагово в консоли. Вывод сканирования содержит имя объёма, через который прошла частица, длину пути по этому объёму, название материала объёма, текущие координаты. Geantino-сканирование позволяет получить детальную картину распределения вещества в Geant4-модели детекторе и узнать, например, сколько радиационных длин перед системами, которые чувствительны к веществу перед ними. Пример печати geantino-сканирования в консоль представлен на Рис. 13.

Помимо этого, сохраняется файл с событиями, содержащий траекторию частицы, который можно посмотреть в утилите EventDisplay (см. Раздел 5.2).

```
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------
+ Material scan between: x_0 = (   0.00,   0.00,   0.00) [cm] and x_1 = ( 300.00,   0.00,   0.00) [cm] :
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|     \   Material          Atomic                  Radiation   Interaction              Path   Integrated  Integrated    Material
| Num. \  Name         Number/Z   Mass/A  Density    Length      Length    Thickness   Length      X0         Lambda      Endpoint
| Layer \                       [g/mole] [g/cm3]     [cm]        [cm]        [cm]      [cm]      [cm]        [cm]    (     cm,     cm,     cm)
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------
|   1 Vacuum                7   14.784   0.0000 3.66346e+11  8.63978e+11      1.500     1.50    0.000000    0.000000  (   1.50,   0.00,   0.00)
|    Path: /world_volume_1/CentralTubeOut_0/CentralTubePara_0/CentralTubeIn_0/CentralTubeVac_0
|
|   2 Beryllium             4    9.012   1.8480     35.2760      39.4488      0.100     1.60    0.002835    0.002535  (   1.60,   0.00,   0.00)
|    Path: /world_volume_1/CentralTubeOut_0/CentralTubePara_0/CentralTubeIn_0
|
|   3 Paraffin              5   10.376   0.9300     48.2235      72.5155      0.100     1.70    0.004908    0.003914  (   1.70,   0.00,   0.00)
|    Path: /world_volume_1/CentralTubeOut_0/CentralTubePara_0
'
```

Рис. 12: Вывод утилиты MaterialScan.

```
ParticleGun.Pdg...  DEBUG  -> geantino
ParticleGun.Pdg...  DEBUG    P   = (1500,0,9.18485e-14,1500), Mass = -9.18485e-14 MeV
SctRNGSvc            INFO Creating engine SimG4Alg/DEFAULT
SimG4Alg.Scanni...   INFO CentralTubeVac Vacuum; Full path = 15; Coordinates = 0 0 0
SimG4Alg.Scanni...   INFO CentralTubeIn Beryllium; Full path = 1; Coordinates = 15 0 9.18485e-16
SimG4Alg.Scanni...   INFO CentralTubePara Paraffin; Full path = 1; Coordinates = 16 0 9.79717e-16
```

Рис. 13: Вывод geantino-сканирования.

4 Моделирование

4.1 Чувствительные детекторы

При прохождении через детектор частицы взаимодействуют с веществом подсистем детектора, оставляя в нём энергию. Для того чтобы Geant4 мог регистрировать такое взаимодействие, необходимо определить объёмы как чувствительные.

Изначально для сохранения результатов моделирования пользователю для каждой выходной коллекции необходимо было задавать соответствующий отдельный инструмент для сохранения данных. Также необходимо вручную прописывать все параметры, которые отличались для каждой выходной коллекции, например, имена коллекций. Затем все эти инструменты необходимо было добавлять в алгоритм моделирования.

Такой подход имеет несколько недостатков:

- дублирование кода. Так как настройка инструментов сохранения для различных выходных коллекций схожа, это увеличивало объём файла jobOption;
- вероятность ошибки при настройке инструмента сохранения, например, неправильное указание подсистемы, для которой сохранять данные;
- накладные расходы по сохранению данных: при таком подходе данные сначала сохранялись Geant4, а затем перекладывались в обменник Gaudi для дальнейшей записи в выходной файл.

Для решения перечисленных проблем было предложено реализовать чувствительные детекторы как инструменты фреймворка Gaudi. Эта реализация включала в себя несколько шагов:

- разработка архитектуры чувствительных детекторов;
- реализация инструментов Gaudi;
- реализация конкретных чувствительных детекторов;
- интеграция в алгоритм моделирования.

При разработке архитектуры преследовались следующие цели: унификация, уменьшение накладных расходов и ускорение расчётов, упрощение для пользователя настройки существующих и добавления новых чувствительных детекторов.

Новый класс чувствительного детектора `SimG4GenericSD` является наследником чувствительного детектора Geant4. Он содержит указатель на соответствующий Gaudi инструмент. Последний в свою очередь хранит указатель на чувствительный детектор, к которому он относится, и определяет его поведение. Благодаря этому, каждый чувствительный детектор сохраняет данные напрямую в обменник Gaudi без промежуточного сохранения результатов моделирования в Geant4.

Были реализованы базовый Gaudi инструмент для чувствительного детектора `SimG4SensitiveDetectorBaseTool` и два конкретных Gaudi инструмента чувствительных детекторов: `SimG4GenericSDTool` и `SimG4EmptySDTool`. Первый сохраняет хиты для чувствительных объёмов, второй можно использовать, если необходимо, чтобы данные для подсистемы, которая включена в моделирование, не сохранялись. Если пользователь хочет добавить свой тип чувствительного детектора, то ему необходимо создать только описание своего класса Gaudi инструмента чувствительного детектора с добавлением своих изменений и сразу использовать его, остальные взаимодействия компонент автоматизированы. Архитектура чувствительных детекторов представлена на Рис. 14.

Интеграция в алгоритм моделирования сделана посредством инструмента `SimG4SensitiveDetectorMasterTool`, который содержит в себе все заданные в настройках чувствительные детекторы и обеспечивает взаимодействие с ними.

Резюмируя сказанное, разработанная архитектура реализует поставленные цели.

Все скрипты моделирования приведены в соответствие с описанными изменениями.

4.2 User Actions

User Actions — это инструмент Geant4, с помощью которого пользователь может взаимодействовать с процессом моделирования на различных

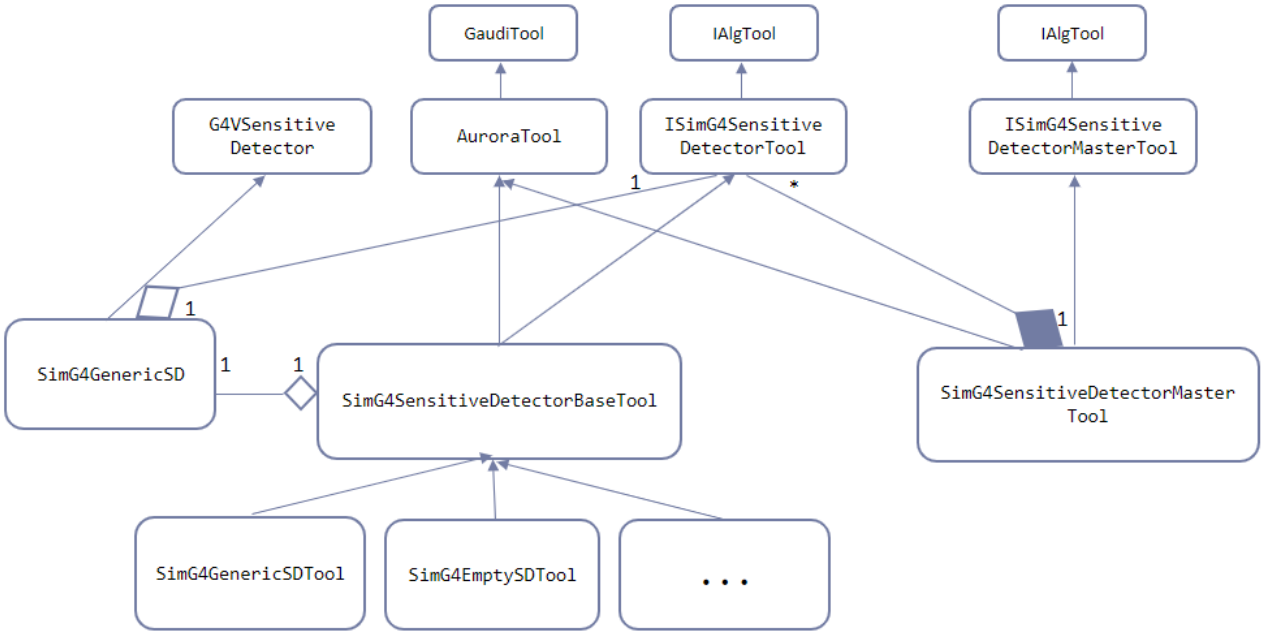


Рис. 14: Архитектура чувствительных детекторов.

этапах: в конце и начале события, трека частицы или шага моделирования. Для этого необходимо реализовать свой класс, который наследуется от одного из абстрактных классов User Action в зависимости от этапа, на котором необходимо получить доступ к данным для дальнейших действий с ними, например, печати дополнительной информации.

Причины изменения реализации используемых User Action (унаследованной из ПО FCC) в нашем фреймворке на реализацию User Action как инструментов фреймворка Gaudi схожи с причинами перепроектирования кода чувствительных детекторов: унификация, упрощение и расширение возможности настройки пользователем, уменьшение накладных расходов при сохранении данных.

При реализации User Action как инструментов фреймворка Gaudi были сделаны:

- изменение существующей архитектуры;
- реализация инструментов Gaudi;
- интеграция в алгоритм моделирования.

Были реализованы `UserEventAction`, `UserTrackingAction` и `UserSteppingAction` — User Action на этапе события, трека и шага. На Рис. 15

представлена полная архитектура для `UserEventAction`. Для `UserTrackingAction` и `UserSteppingAction` архитектура аналогичная.

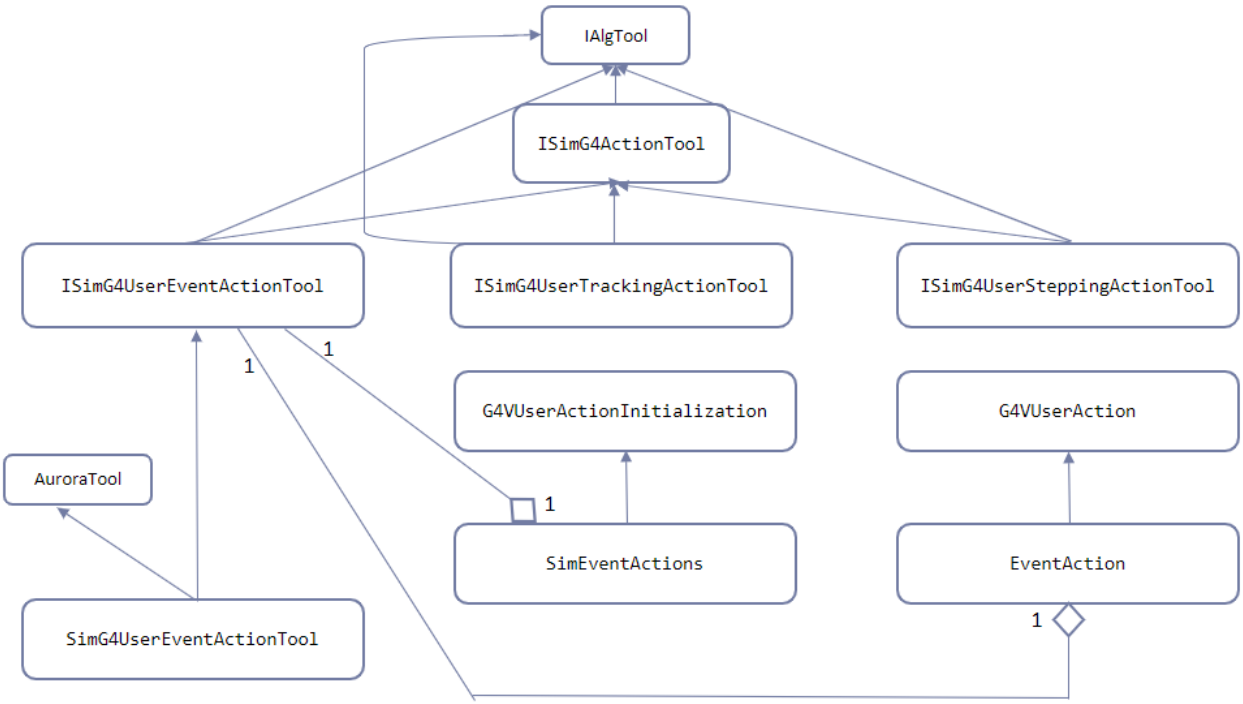


Рис. 15: Архитектура UserEventAction.

Теперь пользователю нужно создать только один User Action инструмент, являющийся наследником одного из существующих, например, `UserTrackingActionTool`, с описанием его особенностей, так как вся остальная логика взаимодействий уже реализована за него. В дополнение к этому, в настройках jobOption можно одновременно задавать несколько различных User Action, но не более одного каждого типа. В предыдущем варианте до изменения архитектуры пользователь должен был реализовывать всю цепочку сам.

Аналогично чувствительным детекторам, данные сохраняются напрямую в обменник Gaudi без промежуточного сохранения в Geant4.

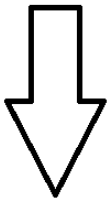
Все скрипты моделирования приведены в соответствие с описанными изменениями.

4.3 Модель данных

Для обмена данными между различными компонентами они (данные) должны иметь одинаковое представление. Поэтому для их хранения и об-

мена ими разрабатывается единая модель данных. Её структура описана в yaml-файле[43], на основе которого с помощью PODIO генерируются C++ классы (Рис. 16).

```yaml
sct::G4Hit:
  Description: "A track hit and with its global pre step and post step positions."
  Author: "A.Zhadan"
  Members:
    - unsigned long long cellId   // Cell id
    - double energy               // Energy
    - double globalTime           // Global time
    - double localTime            // Local time
    - sct::Point preStepPosition  // The pre step point in global frame
    - sct::Point postStepPosition // The post step point in global frame
    - int trackId                 // Track Id from Geant4 of the particle that leave the hit
    - int pdgId                   // PDG Id of the particle that leave the hit
    - sct::LorentzVector momentum // Momentum of the particle that leave the hit
```



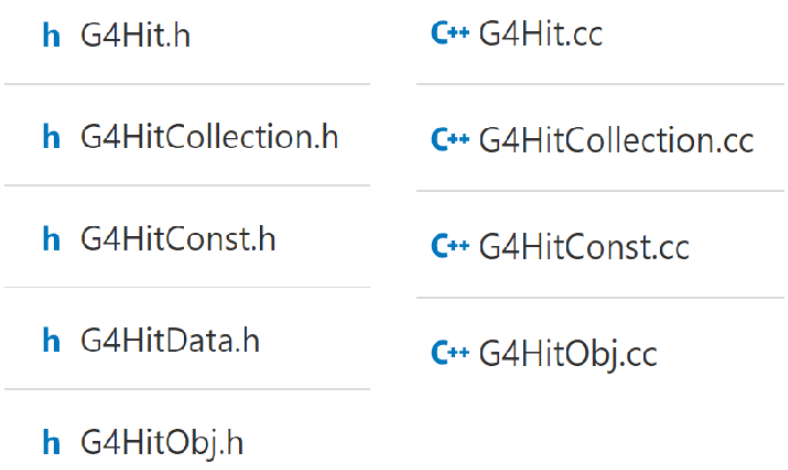


Рис. 16: Генерация C++ классов для типа данных G4Hit из описания модели данных в yaml-файле.

При реализации чувствительных детекторов был разработан и интегрирован в алгоритм моделирования специализированный тип данных — G4Hit

— используемый для хранения информации о хите из моделирования Geant4. Он заменил несколько других типов данных, которые до этого применялись для этой цели одновременно. Это позволило ещё больше упростить и унифицировать код, так как теперь вся необходимая информация хранится только в одной структуре. Кроме того, по сравнению с использованными до этого типами данных, в типе данных G4Hit добавлены новые поля, что увеличивает объём информации для анализа, который получается на выходе. В дальнейшем часть этой информации может быть опущена, так как ПО активно развивается, и модель данных может модифицироваться.

Поля G4Hit:

- unsigned long long cellId — идентификатор чувствительного объёма;
- double energy — энергия на текущем шаге;
- double globalTime — время, прошедшее с начала события;
- double localTime — время с начала текущего трека;
- sct::Point preStepPosition — координаты в начале шага;
- sct::Point postStepPosition — координаты в конце шага;
- int trackId — идентификатор трека частицы из Geant4, которая оставила хит;
- int pdgId — PDG-идентификатор частицы, оставившей хит;
- sct::LorentzVector momentum — импульс частицы.

В типе G4Hit используются уже описанные в yaml-файле вспомогательные структуры sct::Point для хранения координат и sct::LorentzVector для хранения импульса частицы, оставившей данный хит. Использование собственных структур позволяет быть уверенными, что формат данных, единицы измерения сохраняемой в них информации будут правильными и представлены в нужном нам виде.

5 Визуализация детектора СЧТФ и событий в нём

Визуализация является одним из главных инструментов физика. Без неё трудно полноценно оценить корректность описания геометрии, результатов моделирования, оцифровки и реконструкции событий в детекторе.

5.1 GeoDisplay

Утилита GeoDisplay используется для отображения геометрии подсистем детектора, указанных в командной строке при запуске утилиты. Пример отображения представлен на Рис. 17.

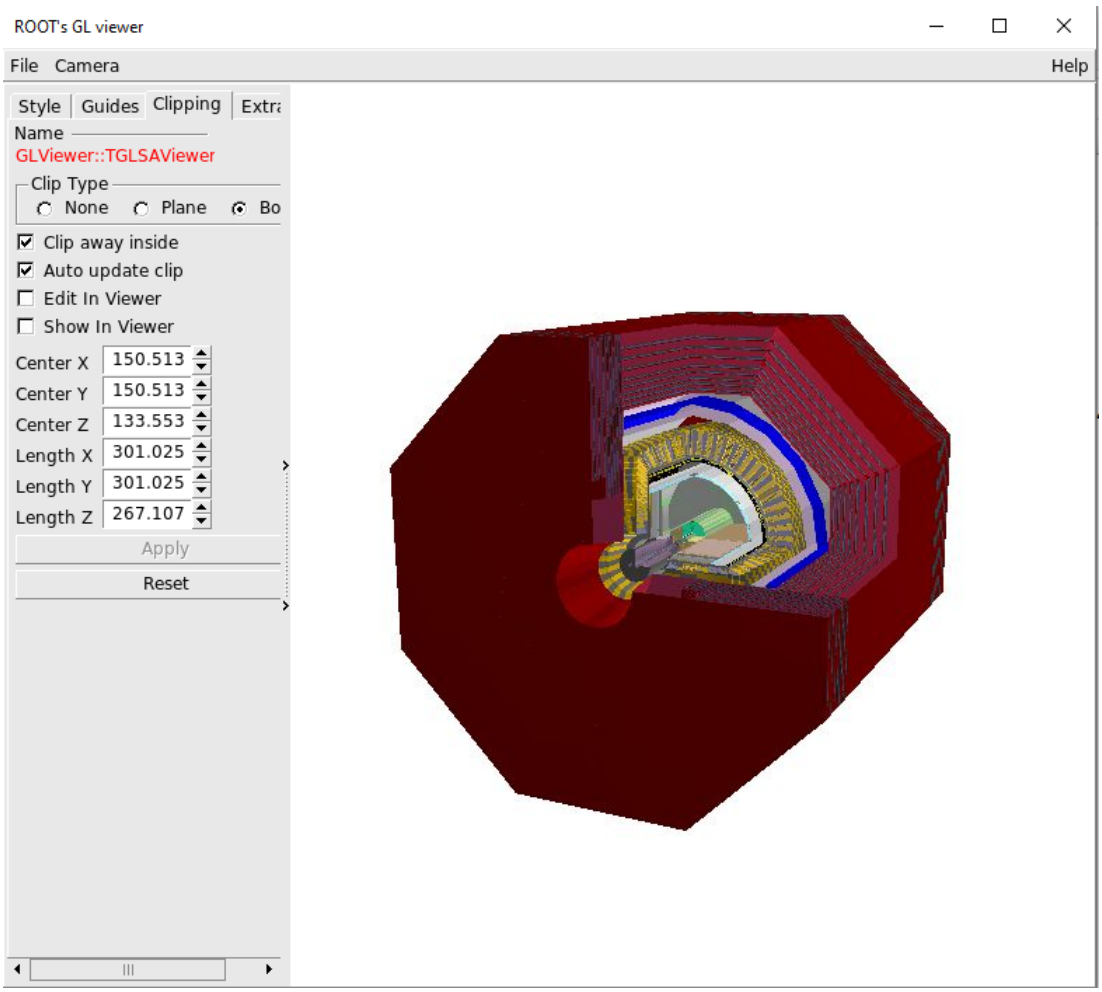


Рис. 17: Отображение геометрии детектора в утилите GeoDisplay.

Эта утилита унаследована из DD4hep и практически не нуждалась в доработке. Единственным улучшением было добавление возможности настройки максимального уровня иерархии для отображения. Причиной для

этого послужила сложная вложенная структура геометрии финального фокуса. Для отображения всей структуры необходим седьмой видимый уровень вложенности, по умолчанию он равен четырём.

5.2 EventDisplay

Утилита EventDisplay используется для отображения сгенерированных событий (траектории частиц и хиты), читая их из файла. Пример отображения представлен на Рис. 18.

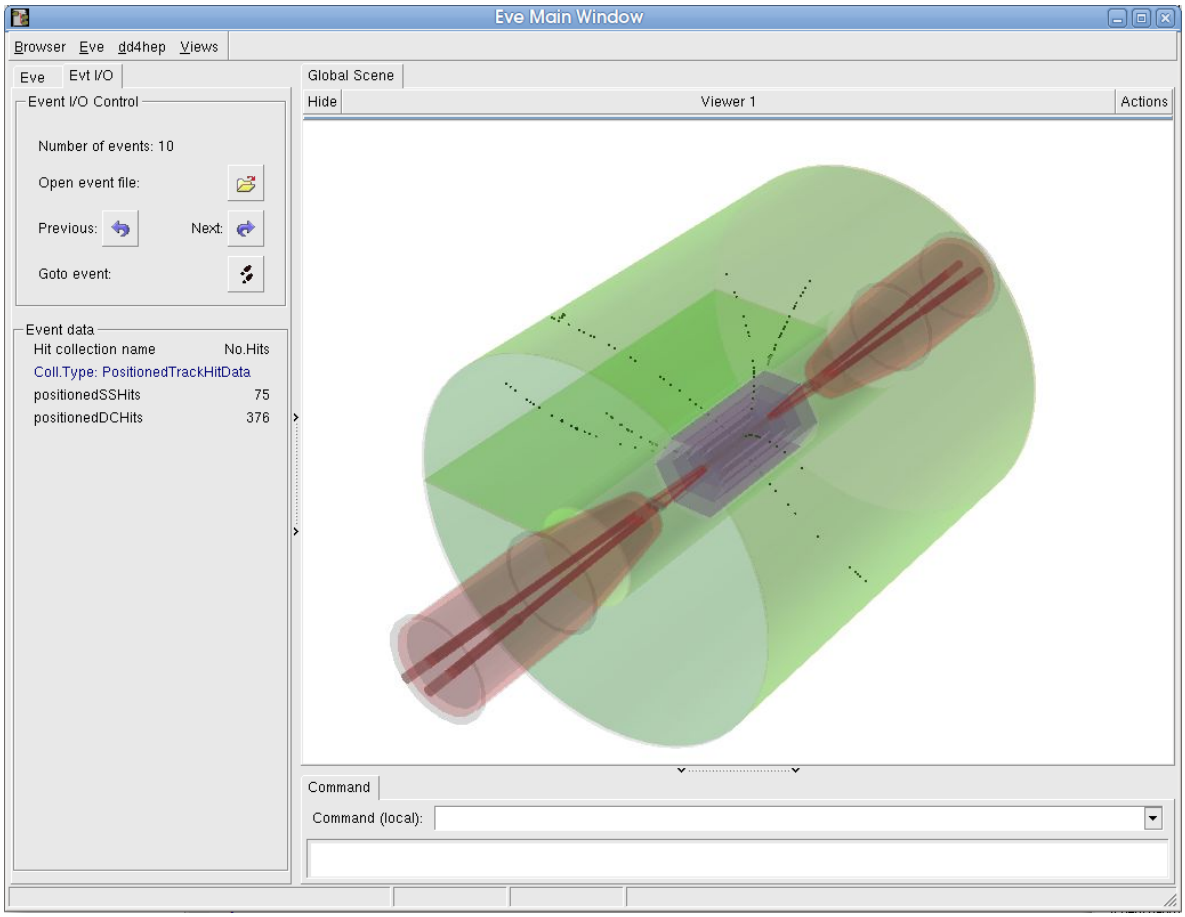


Рис. 18: Отображение события — адронного распада $\psi(4040)$ в центральной части детектора — в утилите EventDisplay.

Утилита основана на компоненте DD4Hep — DDEve, но разработка этой компоненты приостановлена командой DD4hep, поэтому исходная утилита требовала множество изменений, прежде чем можно было начать её использовать. Сперва было решено адаптировать её, не изменяя исходники DD4Hep. Так был добавлен новый класс EventDisplay — наследник исходного Display, который и использовался уже для внесения изменений. Рас-

смотрим их подробнее:

1. Первым изменением было добавление возможности загрузки нескольких XML-файлов с описанием геометрии подсистем, так как описание каждой подсистемы сделано в отдельном пакете с отдельным XML-файлом. В исходном варианте подразумевалось, что описание всех подсистем сделано в одном файле.

2. Добавлена возможность настройки обработчика файлов событий и событий в нём (EventHandler) по имени из конфигурационного файла утилиты EventDisplay.

3. Реализован собственный EventHandler – PodioEventHandler — и компонента, которую PodioEventHandler использует для чтения данных из входного файла с событиями, основанная на PODIO. Новый EventHandler имеет ряд улучшений по сравнению с тем, что был унаследован из ПО FCC и использовался по умолчанию:

   - Была добавлена возможность перехода по событиям вперёд и назад по нажатию соответствующих кнопок интерфейса. В дополнение была добавлена база для реализации перехода к произвольному номеру события.
   - Реализовано отображение сгенерированных при моделировании первичных частиц путём добавления соответствующего обработчика данных. Трек заряженной частицы в магнитном поле отображается спиралью, конечная точка траектории частицы находится на расстоянии трёх метров от начальной координаты.
   - Имена коллекций для отображения определяются автоматически из входного файла.
   - Имена коллекций группируются по типу коллекции, полученных в процессе моделирования, например, коллекции хитов, коллекции частиц. Пример группировки представлен на Рис. 19.

4. В зависимости от типа частицы её траектория отображается соответствующим цветом, например, фотоны — жёлтым, мюоны — голубым и т. д. Пример отображения представлен на Рис. 20.

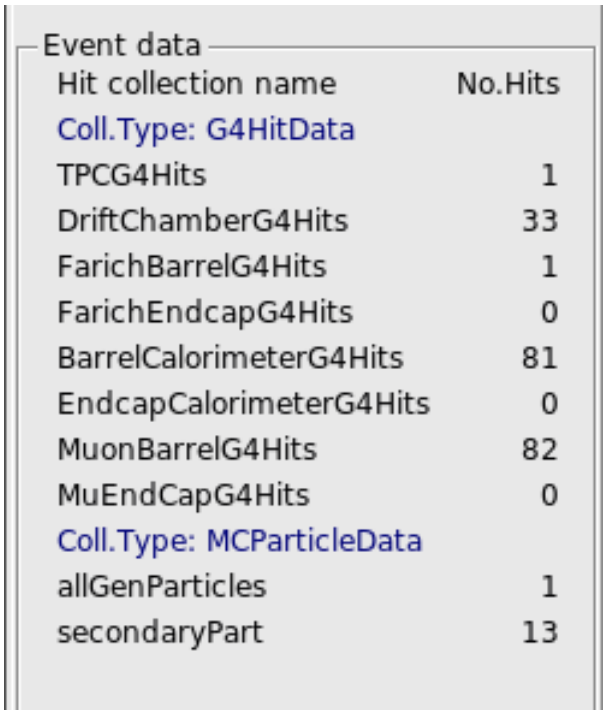


Рис. 19: Группировка имён коллекций по их типу в EventDisplay.

5. Добавлен параметр утилиты для настройки списка коллекций для отображения в EventDisplay с помощью ключа в командной строке.

6. Добавлен новый тип отображения хитов, представляющий собой линию, соединяющую точки. Таким отображением можно грубо представить траекторию частицы на основе хитов, оставленных в подсистеме.

7. Исправлен ряд ошибок.

Перейдём к входным параметрам утилиты EventDisplay:

- имя конфигурационного файла с настройками отображения, например, видимый уровень вложенности геометрии, вид отображения коллекций и т. д.;

- список подсистем детектора для отображения их геометрии. По умолчанию будут загружены геометрии вакуумной камеры, дрейфовой камеры, время-проекционной камеры и катушки;

- имя файла с данными коллекций для отображения;

- список имён коллекций для отображения. По умолчанию отображаются все коллекции, присутствующие во входном файле с данными.

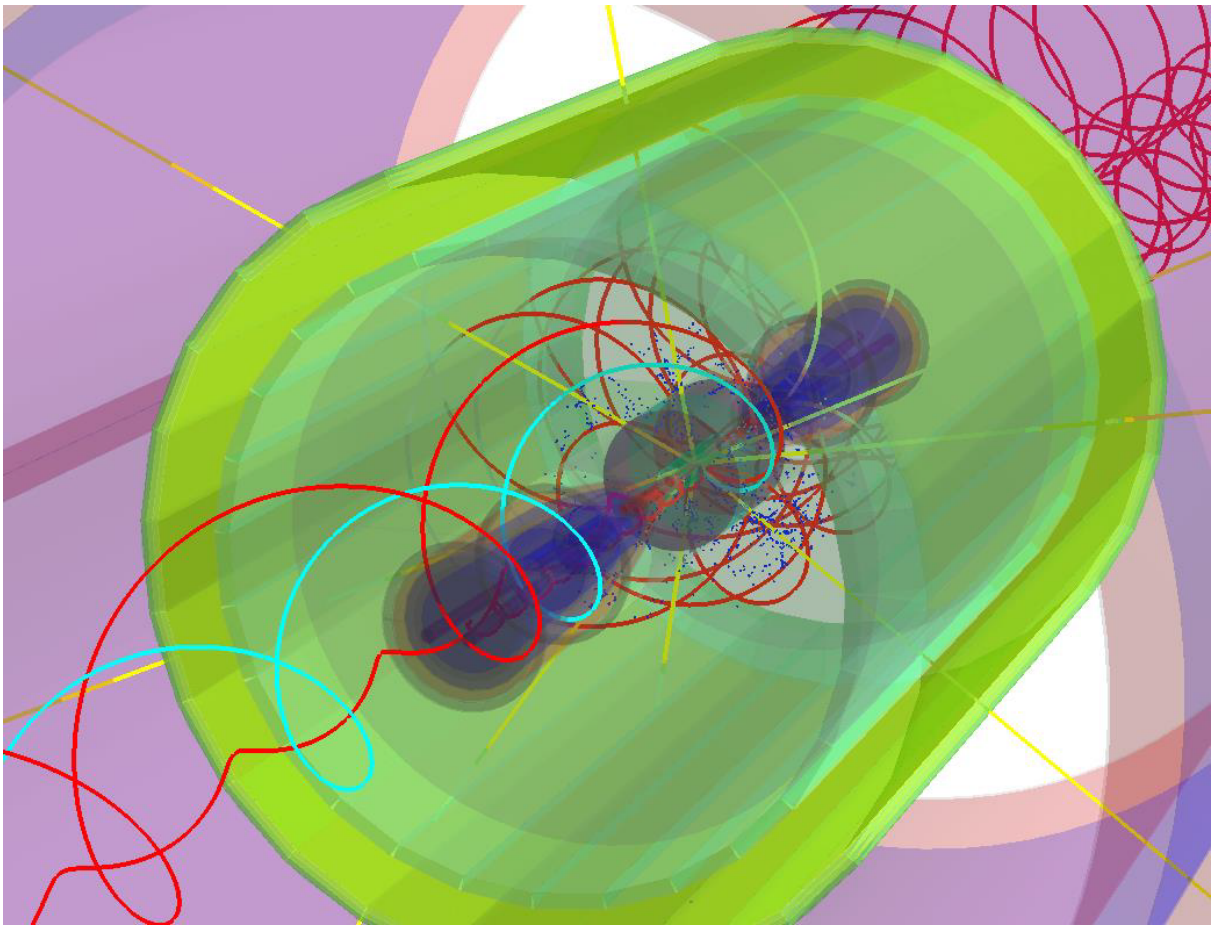


Рис. 20: Пример отображения траектории частиц разными цветами в зависимости от типа: жёлтый — фотоны, голубой — мюоны, красный — пионы, точки — хиты, оставленные частицами в подсистеме.

Как упоминалось ранее, все перечисленные модификации были сделаны без изменения исходных файлов компоненты DDEve. Вследствие этого, даже небольшое изменение требовало проделывания значительного объёма работы по добавлению собственной реализации и переписыванию кода. А дальнейшие запросы по расширению возможностей визуализации, например, подсветка сработавшего кристалла калориметра или добавление флажков в интерфейс EventDisplay, чтобы во время работы включать и отключать коллекции для отображения, требовали более глобальных изменений исходного кода. Поэтому было решено отказаться от расширения DD4Hep компоненты DDEve. Итак, на базе исходной компоненты была создана собственная компонента, чтобы вносить изменения уже в исходный код, таким образом упростив и ускорив разработку. Собственная компонента была успешно добавлена в ПО и протестирована.

Однако работа по развитию EventDisplay была временно приостановлена, так как даже реализация собственной компоненты не могла упростить решение некоторых запросов к визуализации. Объём некоторых требуемых изменений оказался слишком большим, к тому же осталась не решена имевшая место неприятная проблема — нестабильная работа 3D-графики на основе OpenGL при удалённой работе. Дальнейшая деятельность была направлена на поиск альтернативных инструментов.

5.3 WebEventDisplay

5.3.1 Фреймворк Phoenix

При поиске альтернатив утилите EventDisplay был обнаружен фреймворк Phoenix[44], написанный на TypeScript, использующий для отрисовывания библиотеку three.js. Он может быть использован для отображения трёхмерной графики в веб-интерфейсе, является официальным веб-приложением визуализации эксперимента ATLAS и поддерживается HEP Software Foundation (HSF)[45].

Достоинствами Phoenix являются его активное развитие, независимость от ПО пользователя, универсальность для любого эксперимента ФВЭ, быстрая и простая настройка отображения необходимой информации и его расширение в зависимости от эксперимента. Таким образом, было решено использовать данный фреймворк для реализации собственного веб-приложения для визуализации. В качестве базовой версии был взят пример веб-приложения на основе Angular[46] с официального GitHub проекта Phoenix[47], и была начата разработка веб-приложения для отображения геометрии и событий в детекторе СЧТФ.

WebEventDisplay представлен в виде клиент-серверной архитектуры. Серверная часть отвечает за подготовку данных: их форматирование, фильтрацию и отправку только необходимых для запрашиваемых параметров отображения. Клиент преобразует полученный ответ сервера в формат, обрабатываемый Phoenix.

5.3.2 Сервер

Для поиска оптимального варианта между простотой и скоростью реализации и достаточной функциональностью было реализовано несколько вариантов сервера.

В качестве первого варианта был написан простой сервер на nodejs, который подготавливает информацию для отображения и обрабатывает запросы со стороны Phoenix:

1. обработка параметров командной строки:
    - порт, на котором необходимо поднять сервер;
    - имя файла со списком имён подсистем и путями до файлов с геометрией, которые необходимо отобразить;
    - имя файла событий, которые необходимо отобразить.
2. подготовка URL файлов с описанием геометрии подсистем или файла с событиями;
3. по запросу клиента по URL возвращение требуемого файла.

Затем было принято решение реализовать сервер на Python, так как Python позволяет более простым образом осуществлять подготовку и обработку входных и выходных данных, в том числе большого объёма, код получается более компактный, проще осуществлять взаимодействие с остальными компонентами фреймворка Aurora.

Были рассмотрены веб-фреймворки Django[48], Flask[49] и FastAPI[50]. Основным вариантом стал сервер, написанный на Python с использованием фреймворка FastAPI. Он проще, чем Django, который избыточен и включает в себя много функционала не нужного для решения требуемой задачи. В отличие от Flask, FastAPI не требует изучения сторонних модулей для расширения функциональности, необходимые компоненты уже включены, код проще и выше его читаемость, за счёт асинхронности выше скорость обработки запросов.

Часть функционала схожа с функционалом первого сервера на nodejs. Входные параметры аналогичны параметрам для nodejs сервера, FastAPI

сервер также подготавливает URL файлов с описанием геометрии и соответствующие им имена подсистем, но теперь обработка содержимого файла с событиями лежит на стороне сервера, а не клиента. По запросу клиента сервер возвращает данные для отображения события, для которого клиент отправил запрос. Данные передаются в JSON формате.

Для обработки содержимого файла событий используются вспомогательные модули — `WEDHelpers` — каждый из которых отвечает за чтение, обработку своего типа данных, например, полученных в процессе моделирования G4Hit (см. Раздел 4.3) или «сырых» хитов, и подготовку выходных JSON данных.

В рамках работы был добавлен вспомогательный модуль для обработки G4Hit данных. Для каждой коллекции назначается свой цвет отображения. Эта часть WebEventDisplay находится в процессе разработки, и в промежуточном варианте реализации `WEDHelpers` каждый модуль сохраняет выходные данные в отдельный файл.

5.3.3 Клиент

В качестве шаблона был использован пример веб-приложения с использованием Phoenix. В него была добавлена компонента, отвечающая за загрузку и отображение данных детектора СЧТФ.

Пример отображения геометрии детектора и событий представлен на Рис. 21.

С помощью JSROOT[51] каждая геометрия подсистемы вычитывается из отдельного ROOT-файла по URL, предоставленным сервером, и затем загруженные данные обрабатываются Phoenix. Отдельные файлы используются для того, чтобы отображение каждой подсистемы можно было включать и выключать независимо от других. Таким образом, можно загружать сразу и несколько вариантов одной и той же подсистемы, и включать для отображения тот из вариантов, который необходим в данном случае. На Рис. 22 голубым цветом выделено меню по управлению отображением геометрии детектора. При отображении геометрии используются цвета, которые задаются в XML-файлах с описанием геометрии (см. Раздел 3.1).

Были решены проблемы визуализации калориметра и финального фокуса. Калориметр состоит из большого количества кристаллов, как следствие,

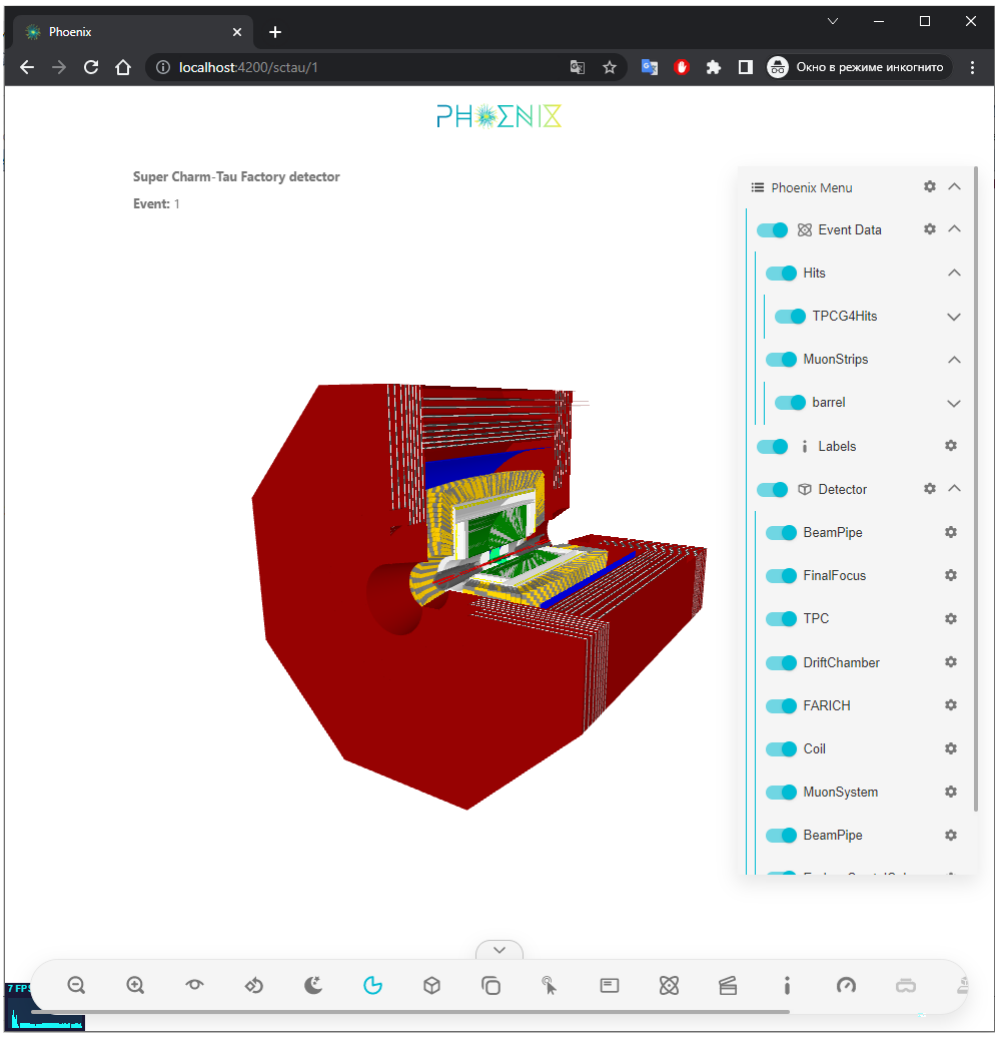


Рис. 21: Отображение геометрии детектора и события в веб-приложении.

содержит и большое количество объёмов. Для отображения калориметра было установлено наибольшее возможное количество отображаемых узлов геометрии, в противном случае он не отображался. С визуализацией финального фокуса была проблема, как и в GeoDisplay — отображались не все уровни вложенности на экране. Эта проблема была решена аналогичным образом — настройкой максимального видимого уровня вложенности геометрии. Были включены настройки для отрисовки поверхностей геометрии не только снаружи, но и внутри. Необходимо это, например, при применении разреза.

Для чтения событий из файла был добавлен собственный загрузчик событий.

В первом варианте, когда сервер был написан на nodejs и обработка содержимого файла была на стороне клиента, чтение файла и извлечение данных было реализовано с помощью JSROOT, поскольку стандартные

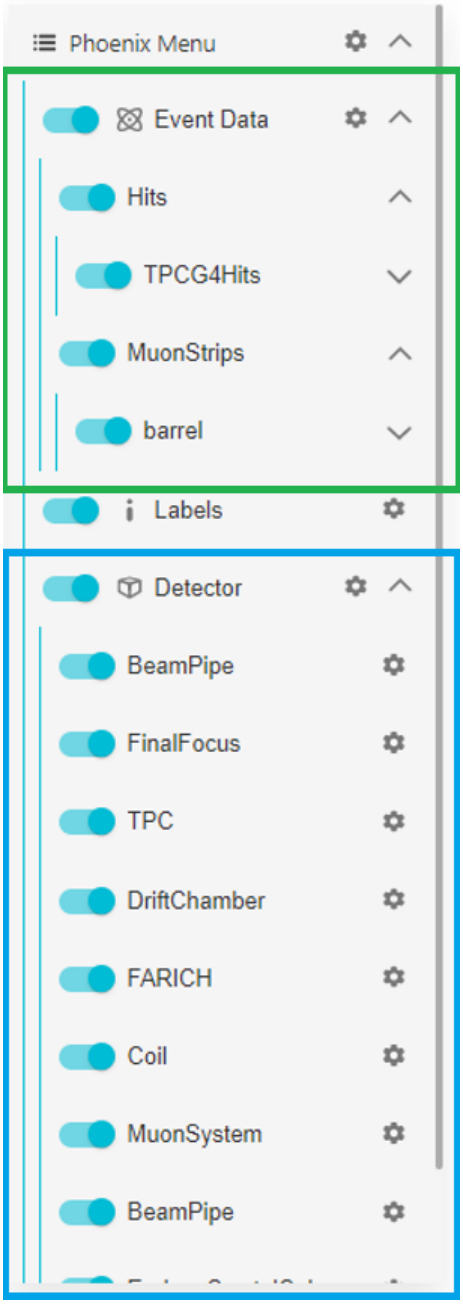


Рис. 22: Меню управления отображением: зелёный — меню событий, голубой — меню геометрии.

загрузчики Phoenix не могут вычитывать данные из древовидной структуры, в которой хранятся выходные данные моделирования, оцифровки и реконструкции. Так как необходимо отображать произвольный файл с событиями, была реализована передача файла с событиями по URL, а не размещение его локально, как во всех примерах, приведённых в Phoenix. Таким образом, на стороне клиента из файла читается дерево с помощью JSROOT, затем вычитанные данные преобразуются в формат JSON, с которым уже работает Phoenix. Были проверены возможности чтения произ-

вольного события, а также нескольких событий параллельно.

При переходе на FastAPI-сервер загрузчик событий был изменён: он отвечает за интерпретацию JSON данных, пришедших с сервера, и их структуризацию для дальнейшей обработки Phonix. Так, каждый хит сохраняется в коллекцию соответствующей подсистемы, в которой его оставила частица. На данный момент окончательный формат JSON данных, приходящих с сервера, не утверждён и будет модифицироваться с целью максимального уменьшения размера данных.

Хиты моделирования и реконструкции отображаются набором точек, возможно отображение «сырых» хитов: кристаллов калориметра и сцинтилляционных счётчиков мюонной системы (Рис. 23). Как и геометрия детектора, отображения разных коллекций события могут быть включены и отключены независимо. На Рис. 22 зелёным цветом выделено меню по управлению отображением событий.

Для перехода по событиям в файле на данный момент используется механизм разрешения URL. В URL текущей веб-страницы указан номер события, которое в данный момент отображается (Рис. 24). В дальнейшем для перехода по событиям будет добавлен соответствующий элемент пользовательского интерфейса.

5.3.4 Запуск веб-приложения

Для запуска веб-приложения WebEventDisplay был разработан соответствующий скрипт. Его входные параметры в текущем варианте реализации:

1. порт, на котором необходимо поднять сервер;
2. список подсистем детектора для отображения;
3. имя файла событий, которые необходимо отобразить.

Геометрия детектора может модифицироваться, а также пользователь может запрашивать различные комбинации подсистем, в том числе различные варианты одной и той же подсистемы. Поэтому геометрия каждой подсистемы генерируется и сохраняется в отдельный ROOT-файл во временной директории пользователя средствами DD4Hep. При этом в текстовый файл (один из параметров командной строки при запуске сервера,

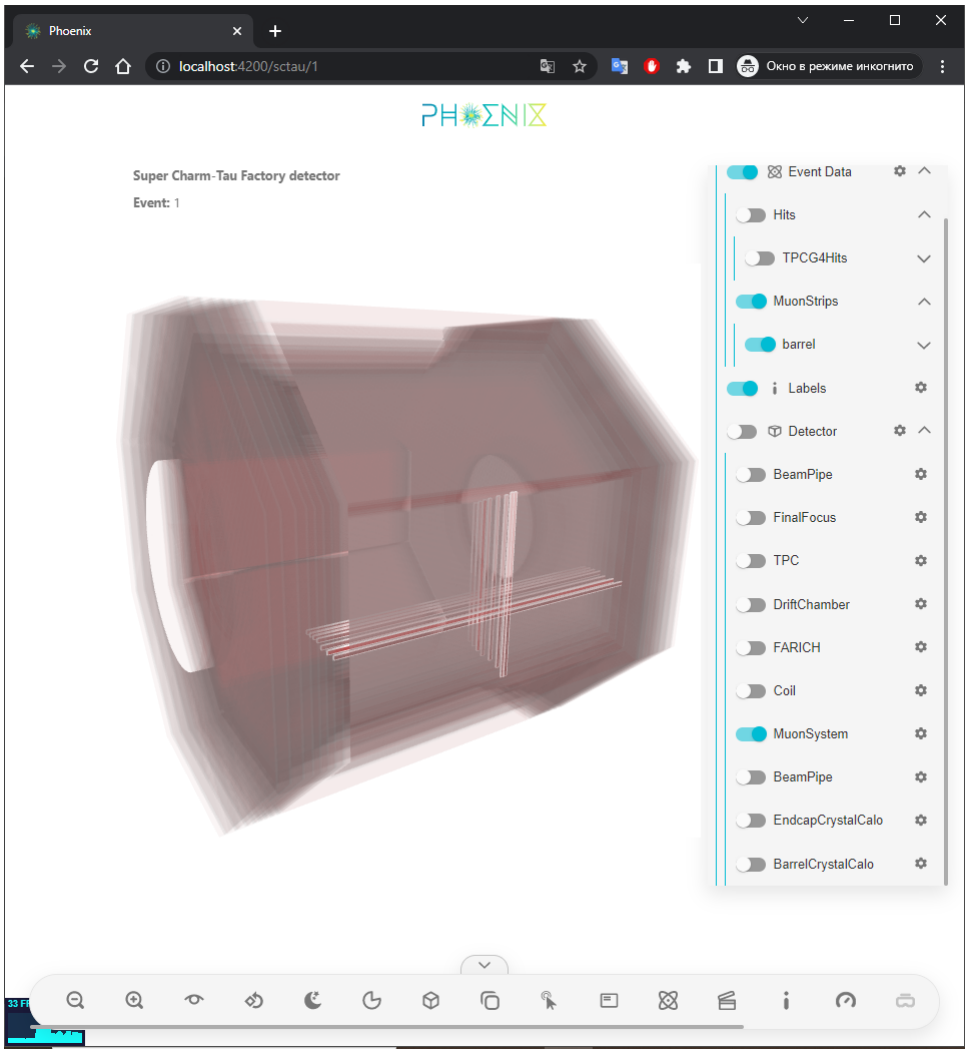


Рис. 23: Пример визуализации события в мюонной системе в веб-приложении. Включено отображение только геометрии мюонной системы с прозрачностью 2% и «сырых» хитов мюонной системы.

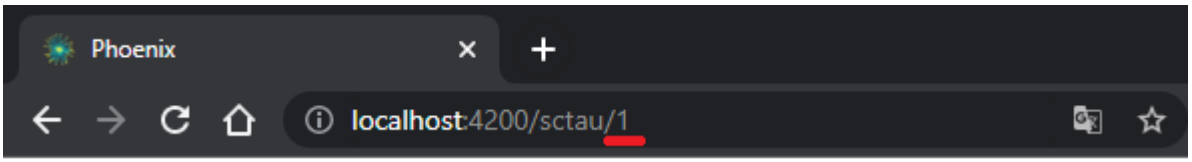


Рис. 24: URL веб-страницы с отображением первого события. Подчёркнутая часть URL отвечает за номер текущего события.

см. Подраздел 5.3.2) записывается путь до ROOT-файла и имя подсистемы, которая содержится в ROOT-файле. Это делается только один раз при запуске приложения.

Затем запускается сервер с указанными ранее параметрами. При остановке сервера все сгенерированные файлы геометрии и промежуточные файлы с данными событий удаляются автоматически.

5.3.5 Планы развития WebEventDisplay

WebEventDisplay активно развивается. В ближайшие планы входит:

- реализация вспомогательных модулей для обработки всех типов данных для отображения и их поддержка в загрузчике событий: «сырые» хиты оставшихся подсистем, хиты реконструкции, треки частиц моделирования и реконструкции;
- реализация взаимодействия вспомогательных модулей между собой для записи выходных данных в один файл;
- расширение пользовательского интерфейса, добавление новых элементов управления, например, кнопки для перехода по событиям;
- реализация упрощённых вариантов геометрии подсистем для увеличения производительности в случае, когда не нужно детальное отображение геометрии и достаточно только контуров подсистем.

Заключение

Одна из актуальных задач, стоящих перед ИЯФ — проект детектора супер-чарм-тау фабрики. Для разработки необходимо моделирование детектора, которое даст возможность рассмотреть и проанализировать различные варианты его устройства, определить ожидаемые параметры, и на основе этого выбрать оптимальный вариант реализации.

В настоящей работе были реализованы полные и базовые варианты описания геометрии для нескольких подсистем и элементов детектора. Была проведена валидация геометрии на разных уровнях, создано средство проверки геометрии в Geant4. Для моделирования были реализованы пакет регистрирующих объёмов детектора и тип данных для хранения информации о результатах моделировании, а также пакет взаимодействия с данными на различных этапах моделирования. Были реализованы инструменты визуализации для отображения геометрии детектора — GeoDisplay, и событий — EventDisplay, а также разрабатывается новое веб-приложение визуализации — WebEventDisplay — на основе современных подходов.

Перечисленные результаты интегрированы в общее программное обеспечение детектора супер-чарм-тау фабрики и используются участниками проекта. Работы будут продолжаться.

Список литературы

[1] Концептуальный проект часть вторая (коллайдер, инжектор) [Электронный ресурс]. URL: https://ctd.inp.nsk.su/wiki/images/f/f3/CDR2_ScTau_ru_vol2.pdf (дата обращения: 15.05.2022)

[2] Yi-Fang Wang. The Construction of the BESIII Experiment // International Journal of Modern Physics A. – 2006. – Vol. 21. – № 27. – P. 5371-5380.

[3] Chang-Zheng Yuan. The BESIII physics programme / Chang-Zheng Yuan, Stephen Lars Olsen // Nature Reviews Physics. – 2019. – Vol. 1. – № 8. – P. 480-494

[4] Концептуальный проект часть первая (физическая программа, детектор) [Электронный ресурс]. URL: https://ctd.inp.nsk.su/wiki/images/8/8b/CDR2_ScTau_ru_vol1.pdf (дата обращения: 15.05.2022)

[5] Brun R. ROOT — An object oriented data analysis framework / R. Brun, F. Rademakers // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. – 1997. – Vol. 389. – Issues 1–2. – P. 81-86.

[6] Antcheva I. ROOT — A C++ framework for petabyte data storage, statistical analysis and visualization / I. Antcheva, M. Ballintijn, B. Bellenot и др. // Computer Physics Communications. – 2009. – Vol. 180. – Issue 12. – P. 2499-2512.

[7] Frank M. DD4hep: A Detector Description Toolkit for High Energy Physics Experiments/ M. Frank, F. Gaede, C. Grefe, P. Mato // Journal of Physics: Conference Series. – 2014. – Vol. 513. – Issue 2.

[8] Petrič M. Detector Simulations with DD4hep / M. Petrič, M. Frank, F. Gaede и др. // Journal of Physics: Conference Series. –2017. – Vol. 898.

[9] Clemencic M. Using DD4hep through Gaudi for new experiments and LHCb / M Clemencic, A. Karachaliou // Journal of Physics: Conference Series. – 2015. – Vol. 664.

[10] Agostinelli S. Geant4 — a simulation toolkit / S. Agostinelli, J. Allison, K. Amako и др. // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. – 2003. – Vol. 506. – Issue 30. – P. 250-303.

[11] Allison J. Recent developments in Geant4 / J.Allisonab, K.Amakoca, J.Apostolakis и др. // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. – 2016. – Vol. 835. – P. 186-225.

[12] Allison J. Geant4 developments and applications / J. Allison, K. Amako, J. Apostolakis и др. // 2006 IEEE Transactions on Nuclear Science. – 2006. – Vol. 53. – № 1. – P. 270-278.

[13] Эксперимент ATLAS [Электронный ресурс].
URL: https://atlas.cern/ (дата обращения: 15.05.2022)

[14] The ATLAS Collaboration. The ATLAS Experiment at the CERN Large Hadron Collider / The ATLAS Collaboration, G. Aad, E. Abat и др. // Journal of Instrumentation. – 2008. – Vol. 3. – № 8. – P. S08003.

[15] Эксперимент LHCb [Электронный ресурс].
URL: https://lhcb-outreach.web.cern.ch/ (дата обращения: 15.05.2022)

[16] Belyaev I. The history of LHCb / I. Belyaev, G. Carboni, N. Harnew и др. // The European Physical Journal H. – 2021. – Vol. 46. – № 1.

[17] Фреймворк Gaudi [Электронный ресурс].
URL: https://gaudi-framework.readthedocs.io/en/latest/ (дата обращения: 15.05.2022)

[18] Barrand G. GAUDI — A software architecture and framework for building HEP data processing applications / G.Barranda, I.Belyaevb, P.Binko и др. // Computer Physics Communications. – 2001. – Vol. 140. – P. 45–55.

[19] Gaede F. PODIO: An Event-Data-Model Toolkit for High Energy Physics Experiments / F. Gaede, B. Hegner, P. Mato // Journal of Physics: Conference Series. – 2017. – Vol. 898. – P. 072039 (P. 1-6).

[20] Gaede F. PODIO: recent developments in the Plain Old Data EDM toolkit / F. Gaede, B. Hegner G. A. Stewart // 24th International Conference on Computing in High Energy and Nuclear Physics. – Adelaide, Australia, 2019. – P. 05024.

[21] Future Circular Collider [Электронный ресурс].
URL: https://home.cern/science/accelerators/future-circular-collider (дата обращения: 15.05.2022)

[22] Suarez R.G. The Future Circular Collider (FCC) at CERN [Электронный ресурс]. URL: https://arxiv.org/abs/2204.10029 (дата обращения: 27.05.2022)

[23] ПО Future Circular Collider [Электронный ресурс].
URL: http://hep-fcc.github.io/FCCSW/ (дата обращения: 15.05.2022)

[24] Эксперимент Belle II [Электронный ресурс].
URL: https://www.belle2.org/ (дата обращения: 26.05.2022)

[25] Robertson S.H. The Belle II Experiment // Journal of Physics: Conference Series. – 2019. – Vol. 1271. – № 1. – P. 012011.

[26] Kuhr T. The Belle II Core Software / T. Kuhr, C. Pulvermacher, M. Ritter и др. // Computing and Software for Big Science. – 2018. – Vol. 3. – № 1.

[27] Belozyorova M. Software framework for the Super Charm-Tau factory detector project / M. Belozyorova, D. Maksimov, G. Razuvaev и др. // Proceedings of the 25th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2021). – Virtual Event hosted by CERN, 2021. – Vol. 251.

[28] Belozyorova M. Computing Environment for the Super-Charm-Tau Factory Detector Project/ M. Belozyorova, D. Maksimov, G. Razuvaev и др. // Proceedings of the 9th International Conference

«Distributed Computing and Grid Technologies in Science and Education» (GRID'2021). – Dubna, Russia, 2021.

[29] EvtGen [Электронный ресурс].
URL: https://evtgen.hepforge.org/ (дата обращения: 26.05.2022)

[30] Lange D.J. The EvtGen particle decay simulation package / Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. – 2001. – Vol. 462. – Issues 1-2. – P. 152-155.

[31] Tauola, [Электронный ресурс].
URL: http://tauolapp.web.cern.ch/tauolapp/ (дата обращения: 26.05.2022)

[32] Shekhovtsova O. Study of the tau meson decay modes with Monte Carlo generator TAUOLA. Status and perspectives // Nuclear and Particle Physics Proceedings. – 2015. – Vol. 260. – P. 52-55.

[33] PHOTOS [Электронный ресурс].
URL: http://photospp.web.cern.ch/photospp/ (дата обращения: 26.05.2022)

[34] Golonka P. PHOTOS Monte Carlo: a precision tool for QED correctionsin Z and W decays / P. Golonka, Z. Was // The European Physical Journal C. – 2006. – Vol. 45. – № 1. – P. 97-107.

[35] Pythia [Электронный ресурс].
URL: https://pythia.org/ (дата обращения: 26.05.2022)

[36] Sjöstrand T. The Pythia event generator: Past, present and future // Computer Physics Communications. – 2020. – Vol. 246. – P. 106910.

[37] ArbuzovA.B. Monte-Carlo generator for e^+e^- annihilation into lepton and hadron pairs with precise radiative corrections / A.B. Arbuzov, G.V. Fedotovich, F.V. Ignatov и др. // The European Physical Journal C. – 2006. – Vol. 46. – № 3. – P. 689–703.

[38] Carloni Calame C.M. The BABAYAGA event generator / C.M. Carloni Calame, G. Montagna, O. Nicrosini и др. // SIGHAD03 - Worskhop on Hadronic Cross Section at Low Energy. – Pisa, Italy, 8-10 October, 2003.

[39] Balossini G. Matching perturbative and parton shower corrections to Bhabha process at flavour factories / G. Balossini, C. M. Carloni Calame, G. Montagna и др. // Nuclear Physics B. – 2006. – Vol. 758. – № 1-2. – P. 227-253.

[40] Kleiss R. BBBREM — Monte Carlo simulation of radiative Bhabha scattering in the very forward direction / R. Kleiss, H. Burkhardt // Computer Physics Communications. – 1994. – Vol. 81. – № 3. – P. 372–380.

[41] Jadach S. The precision Monte Carlo event generator for two-fermion final states in collisions / S. Jadach, B.F.L. Ward, Z. Was // Computer Physics Communications. – 2000. – Vol. 130. – № 3. – P. 260-325.

[42] LCGCMake [Электронный ресурс].
URL: https://gitlab.cern.ch/sft/lcgcmake (дата обращения: 15.05.2022)

[43] Язык Yaml [Электронный ресурс].
URL:https://yaml.org/ (дата обращения: 15.05.2022)

[44] Moyse E. The Phoenix event display framework / E. Moyse, F. Ali, E. Cortina и др. // Proceedings of the 25th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2021). – Virtual Event hosted by CERN, 2021. – Vol. 251.

[45] HEP Software Foundation [Электронный ресурс].
URL: https://hepsoftwarefoundation.org/projects.html (дата обращения: 15.05.2022)

[46] Angular [Электронный ресурс].
URL: https://angular.io/ (дата обращения: 26.05.2022)

[47] Git ареймворка Phoenix [Электронный ресурс].
URL: https://github.com/HSF/phoenix (дата обращения: 15.05.2022)

[48] Веб-фреймворк Django [Электронный ресурс].
URL: https://www.djangoproject.com/ (дата обращения: 15.05.2022)

[49] Веб-фреймворк Flask [Электронный ресурс].
URL: https://palletsprojects.com/p/flask/ (дата обращения: 15.05.2022)

[50] Веб-фреймворк FastAPI [Электронный ресурс].
URL: https://fastapi.tiangolo.com/ (дата обращения: 15.05.2022)

[51] JavaScript ROOT [Электронный ресурс].
URL: https://root.cern/js/ (дата обращения: 26.05.2022)