Направления развития вычислений, ПО и работы с данными в ИЯФ

А. Сухарев и др.

Выездное собрание «Дискуссионного клуба» в Разливе

27 августа 2026

ИЯФ СО РАН

Настоящее и будущее экспериментов по ФВЭ в ИЯФ

- ВЭПП-2000
  - КМД-3 ($\rightarrow$ 4)
  - СНД ($\rightarrow$?)

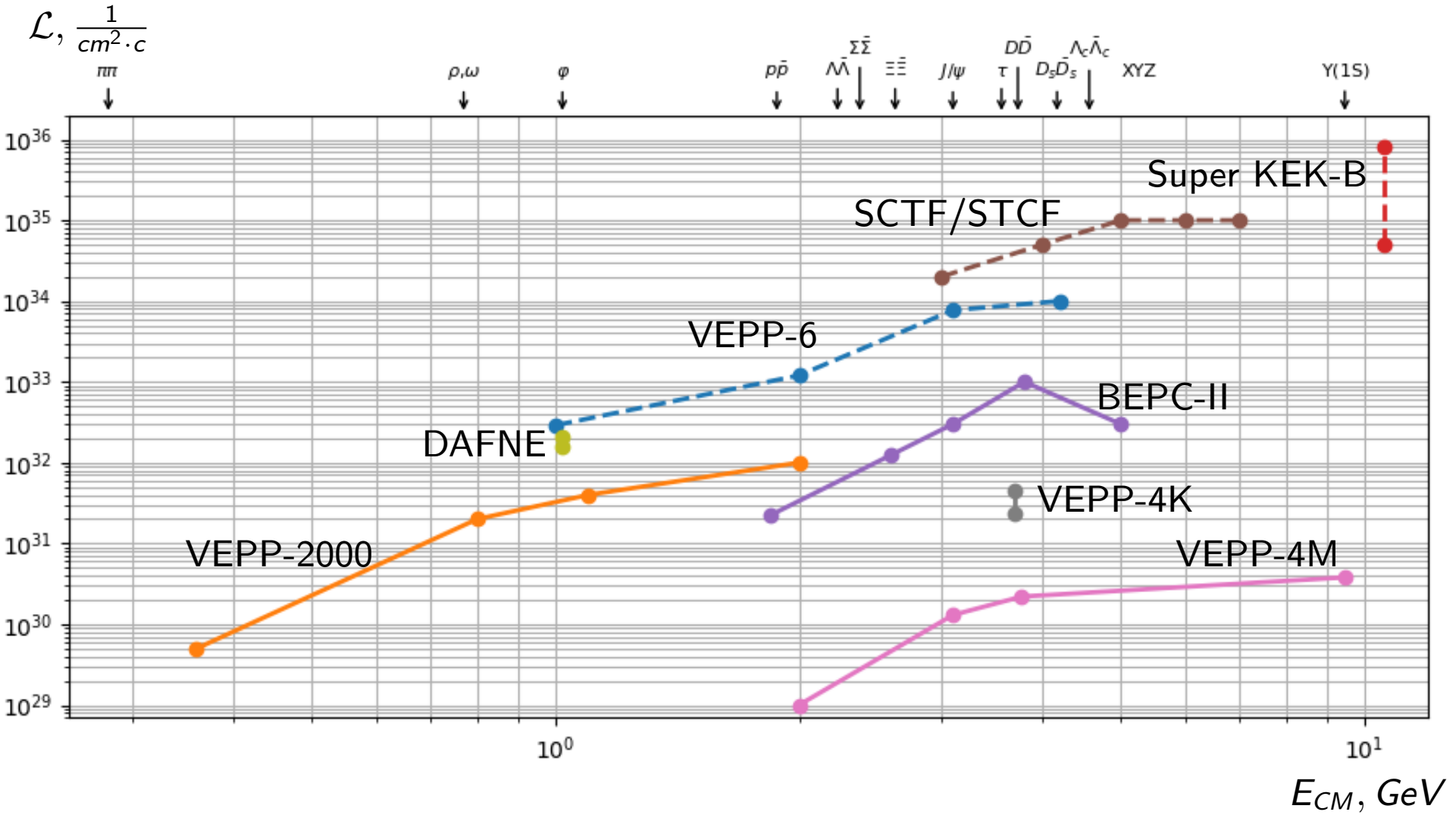
  $\gtrsim$ 10 лет
- ВЭПП-4М
  - КЕДР

  $\sim$ 1 год
- ВЭПП-6
  - детектор для ВЭПП-6

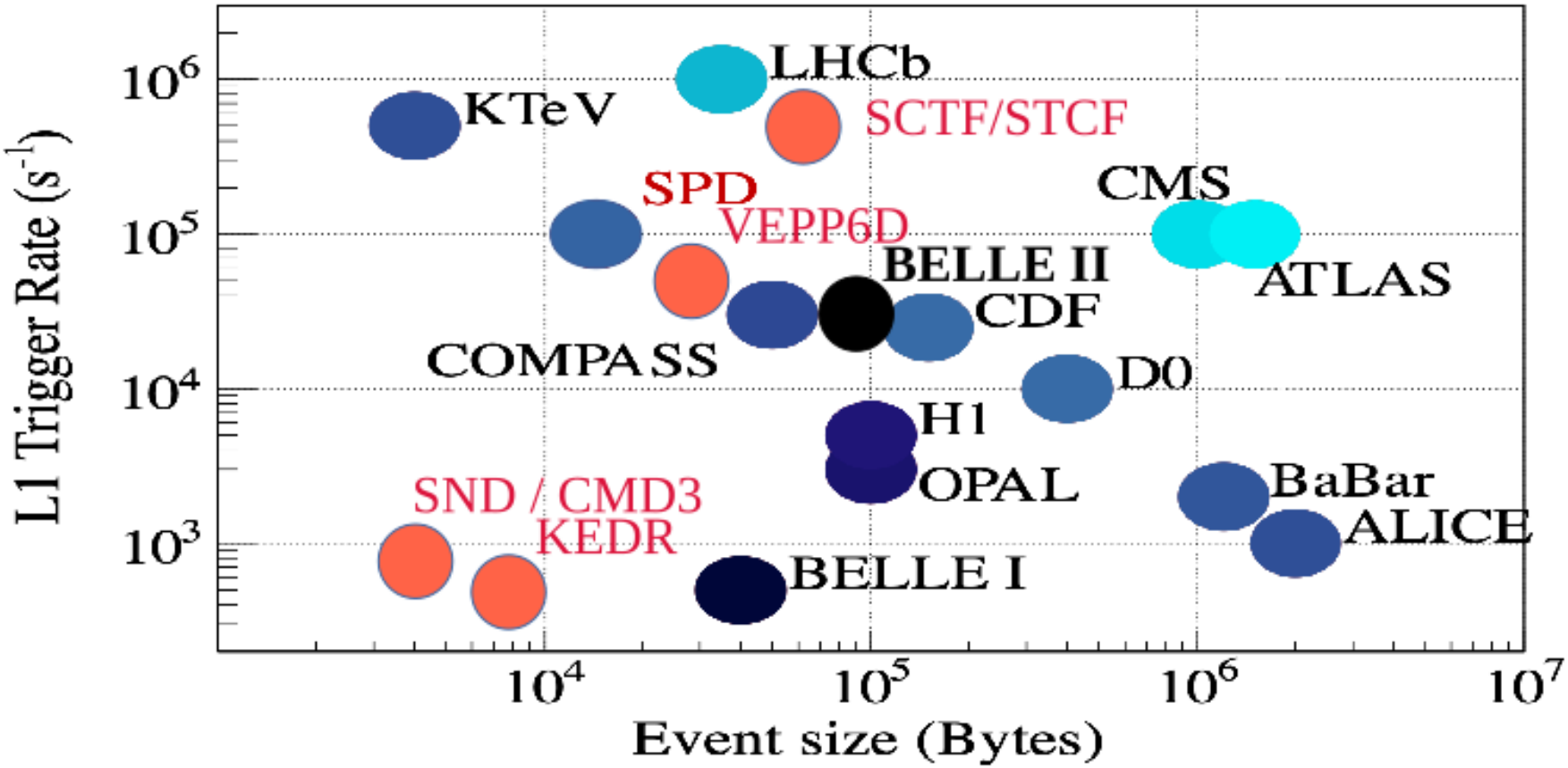
  $\gtrsim$ 10 лет

Три одновременно работающих эксперимента + участие во внешних коллаборациях

Электрон-позитронные коллайдеры в мире

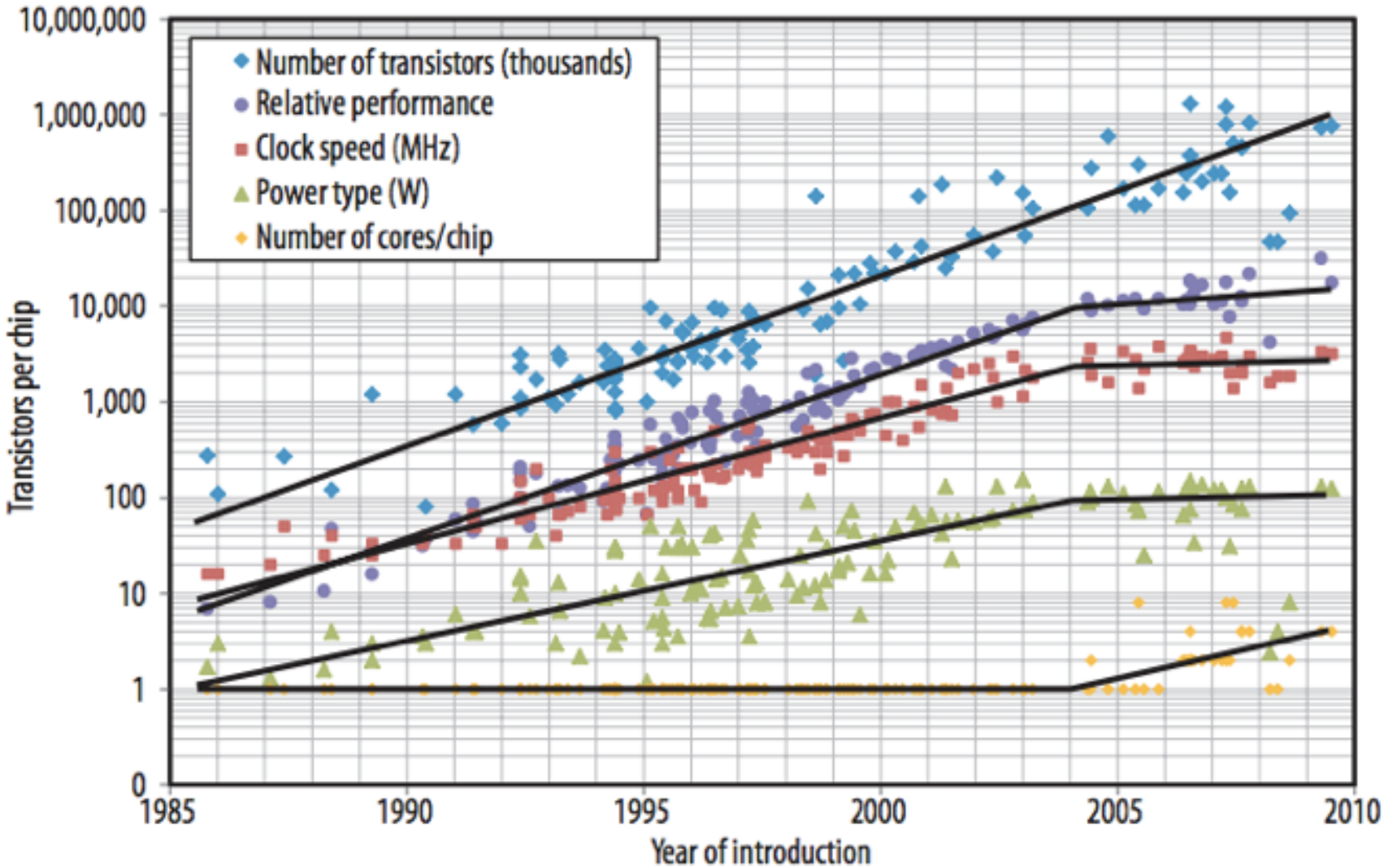


Потоки и объёмы экспериментальных данных



на основе FTCF2024-Guangzhou, Alexey Zhemchugov

Эволюция вычислительных мощностей



Fuller, Millett (2011)

Коллаборации и люди

В мире:

- BES III — $\sim$ 700 человек
- Belle II — $\sim$ 1000 человек
- MPD — 500+ человек
- SPD — 400+ человек

В ИЯФ:

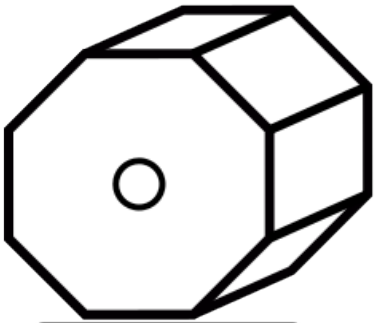
- КМД — $\sim$ 30 человек
- СНД — $\sim$ 30 человек
- КЕДР — $\sim$ 30 человек
- детектор для ВЭПП-6 — $\sim$ 15 человек

Ключевые принципы построения будущего программного обеспечения и вычислительных ресурсов

- Объединять усилия и опыт всех экспериментальных групп института,
- максимально использовать имеющиеся мировые наработки,
- при этом опираясь как проверенные временем решения, так и современные технологии,
- совместно развивать и разрабатывать ПО на общих ресурсах, что
  - позволит использовать и развивать эти ресурсы наиболее эффективно,
  - не отменяет приватных ресурсов (но в меру).

Программное обеспечение эксперимента ФВЭ

Компоненты ПО, потоки и типы данных



Модель данных

Данные об условиях эксперимента

Конфигурационные данные

Генерация

Сгенерированное событие

Геометрия

Моделирование

Смоделированное взаимодействие с детектором

Система сбора данных

Оцифровка

Смоделированные сигналы с детектора в виде, эквивалентном сырым данным

Сырые данные

Реконструкция

Восстановленное событие
Event summary data

Система хранения

Абстрагирование
Contraction
Derivation

Объекты анализа

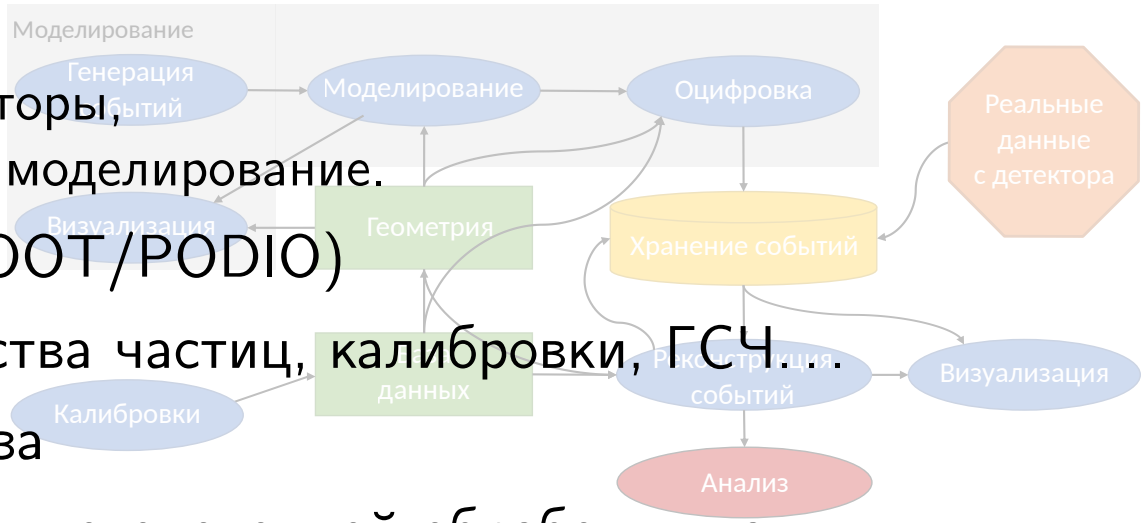
Система визуализации

Анализ

Программная среда Аврора

Изначально разрабатывалась для проекта SCTF, но достаточно универсальна

- Основана на Gaudi
- Многое почерпнуто из FCCSW, Athena и BASF2
- Геометрия (DD4hep, GeoModel(?))
- Моделирование:
  - первичные генераторы,
  - быстрое и полное моделирование.
- Хранение данных (ROOT/PODIO)
- Общие данные: свойства частиц, калибровки, ГСЧ...
- Имеется пакет анализа
- Разработка с учётом многопоточной обработки данных
- Модули: C++, интеграция и JobOptions: Python



Управление средой Аврора

- Система сборки и конфигурации построена по образу эксперимента ATLAS.
- Организованы полные ночные сборки.
- Разработаны средства для выпуска и управления релизами.
- Система `lcgcmake` используется для сборки внешних пакетов,
- Текущая рабочая среда AlmaLinux 8, gcc13 (C++17), Python3.
- Следующая планируемая среда AlmaLinux 9, gcc15 (C++20), Python3.
- ПО частично распространяется через CVMFS, планируется полный переход.

Принципы построения ПО на базе Gaudi

ПО реализует набор стандартных объектов. На каждом событии

- *Алгоритм* получает *Данные* и преобразует их
- *Данные* и *Метаданные* поступают в *Алгоритм* из других *Алгоритмов* и/или *Сервисов*
- *Алгоритмы* и *Сервисы* могут использовать *Инструменты* для каких-то специальных действий
- Вызов *Алгоритмов* в правильном порядке и передача данных между ними обеспечивается Gaudi

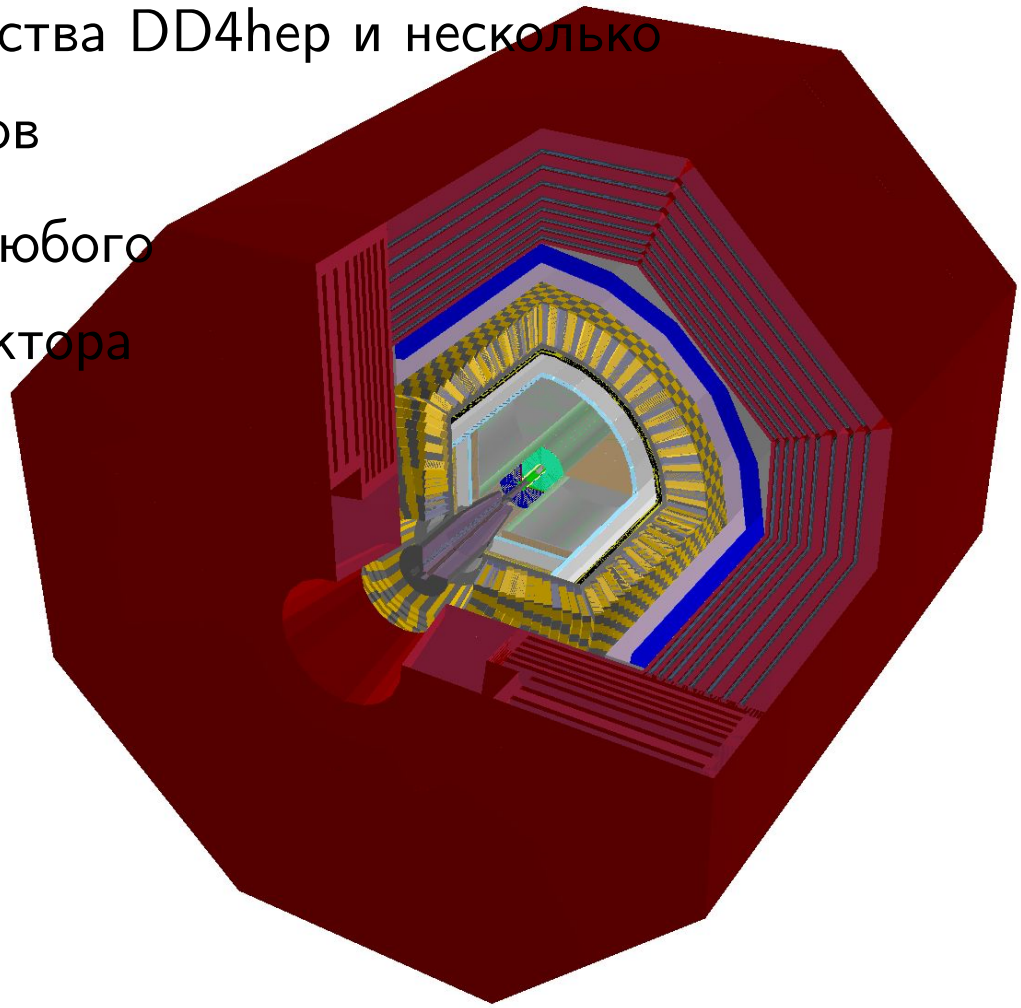
Примеры:

- **Algorithm**: генерация событий; реконструкция системы; совместная реконструкция
- **Service**: метаданные захода, события; калибровки
- **Tool**: расчёт положений кристаллов; наложение условий отбора

Средства работы с геометрией в Авроре

DD4hep

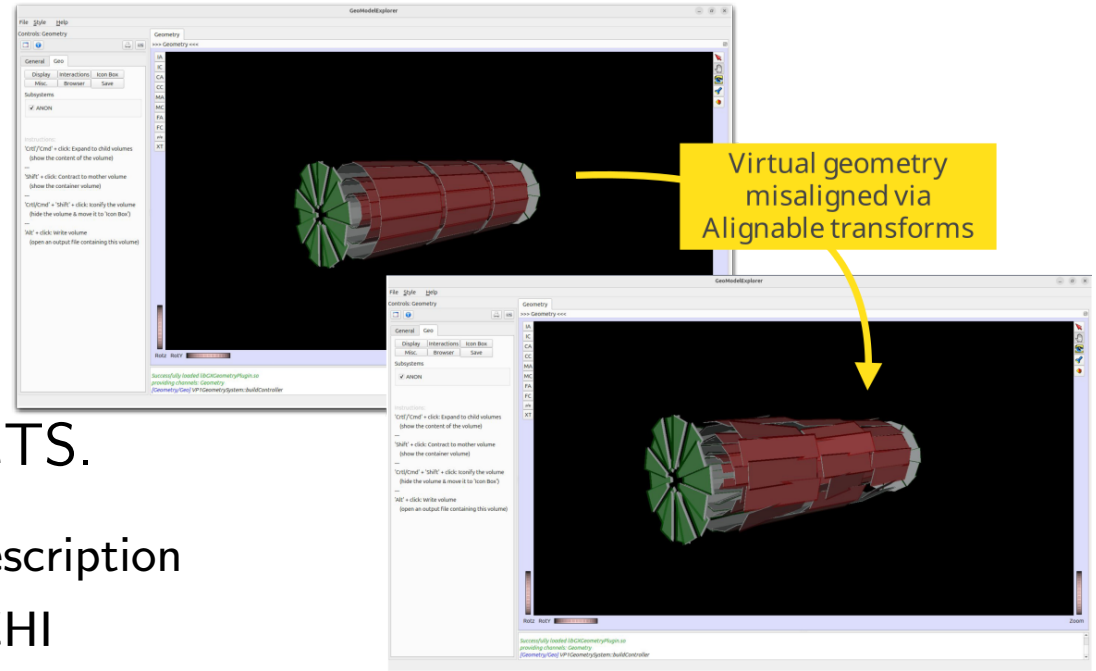
- Доступны стандартные средства DD4hep и несколько специализированных объёмов
- Достаточны для описания любого сколь угодно сложного детектора
- Имеются средства тестирования геометрии (пересечения объёмов, печать материалов . . .)
- Инструменты визуализации детектора



Средства работы с геометрией в Авроре

GeoModel (планируется)

- Разработан коллаборацией ATLAS, зрелость подтверждена более чем 20-летним опытом работы. Продолжает развиваться.
- Собственная среда разработки, тестирования и визуализации геометрии вне основного фреймворка.
- Готовые варианты компоновки могут отмечаться git-тэгами.
- Единый источник информации.
- Модульный дизайн и систему плагинов.
- Виртуальная геометрия (2D-поверхности), выставка систем.
- Интеграция с Geant4 и ACTS.



CHEP 2024, Updating the software description of the ATLAS Detector, R. M. BIANCHI

Первичные генераторы

Доступные генераторы:

- Физические генераторы:
  - EvtGen, Tauola, PHOTOS, Pythia, KKMC
- Технические генераторы
  - Particle gun, ...

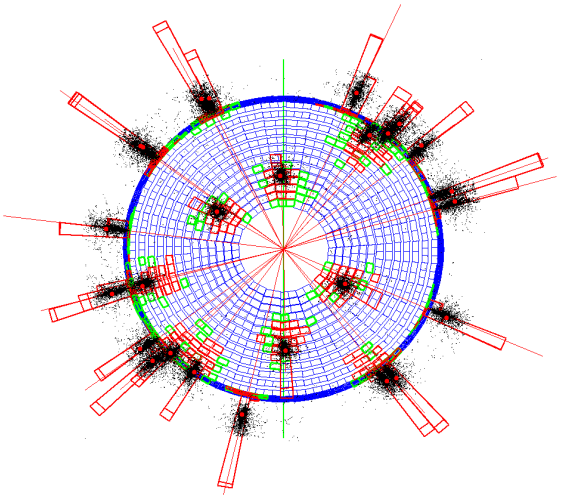
Список расширяется по необходимости

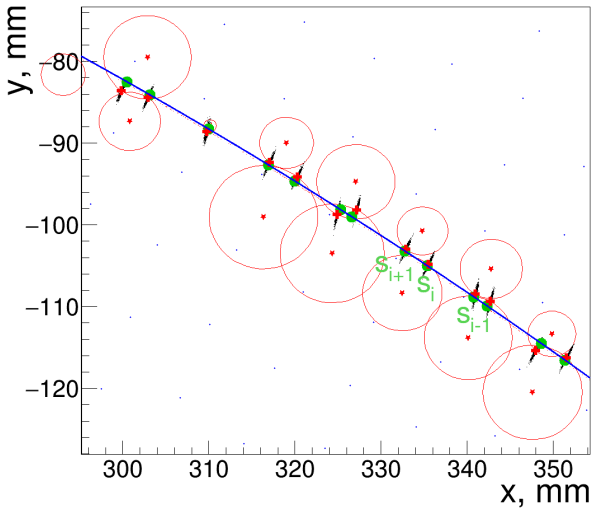
Моделирование

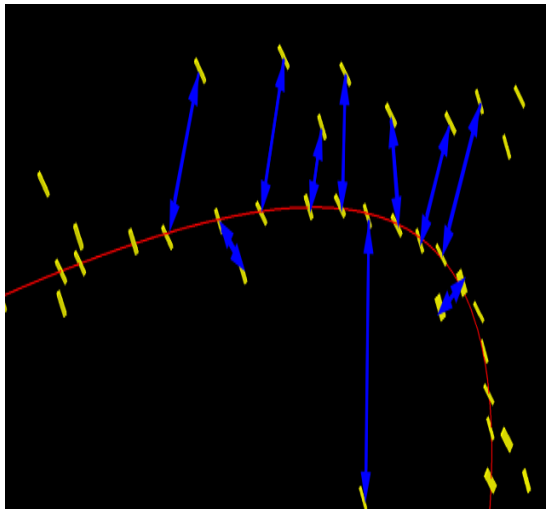
- Быстрое моделирование:
  - выдаёт результат, аналогичный полной реконструкции,
  - моделирование на уровне подсистем, доступны трекер, калориметр, PID и мюонная система,
  - используются параметрические модели или данные специального Geant4 моделирования подсистем,
  - вычисляются ассоциации между результатами работы подсистем.
- Полное моделирование:
  - используется Geant4,
  - на выходе коллекции G4Hit.

Оцифровка и реконструкция

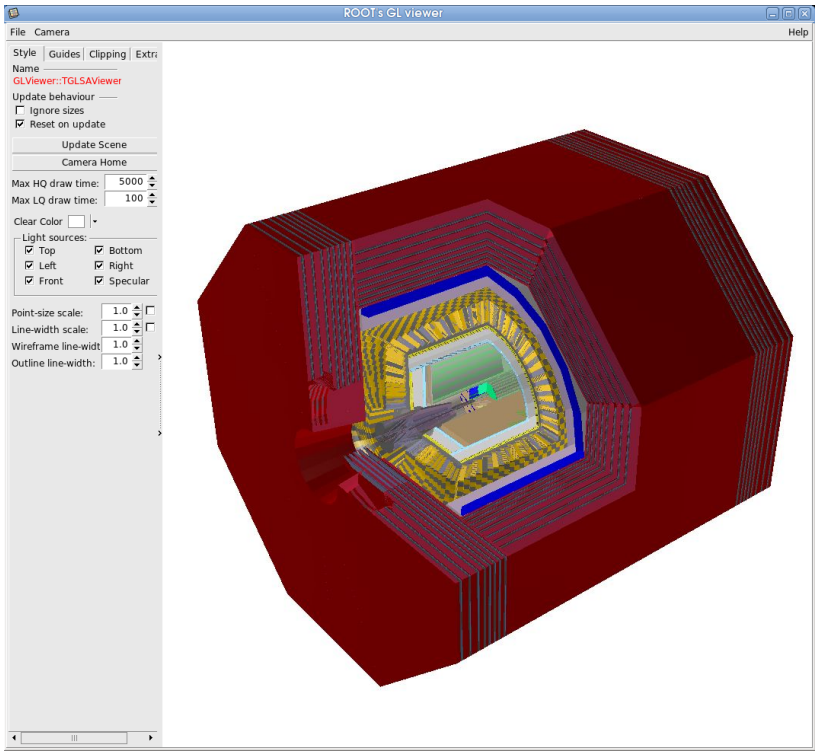
- Оцифровка — моделирование процессов формирования отклика детектора на основе имеющегося энерговыделения.
  - Основывается на специализированных работах в каждой подсистеме.
- Реконструкция:
  - отдельные подсистемы: (ДК), калориметр, мюонная система
  - совместная реконструкция: ассоциации между результатами реконструкции отдельных подсистем, индивидуальные и комбинированный PID.

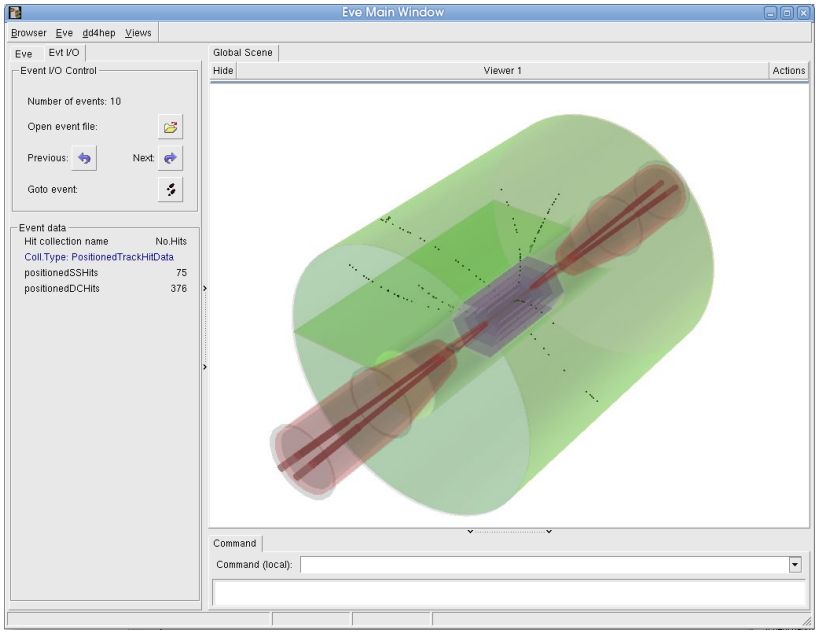






Визуализация



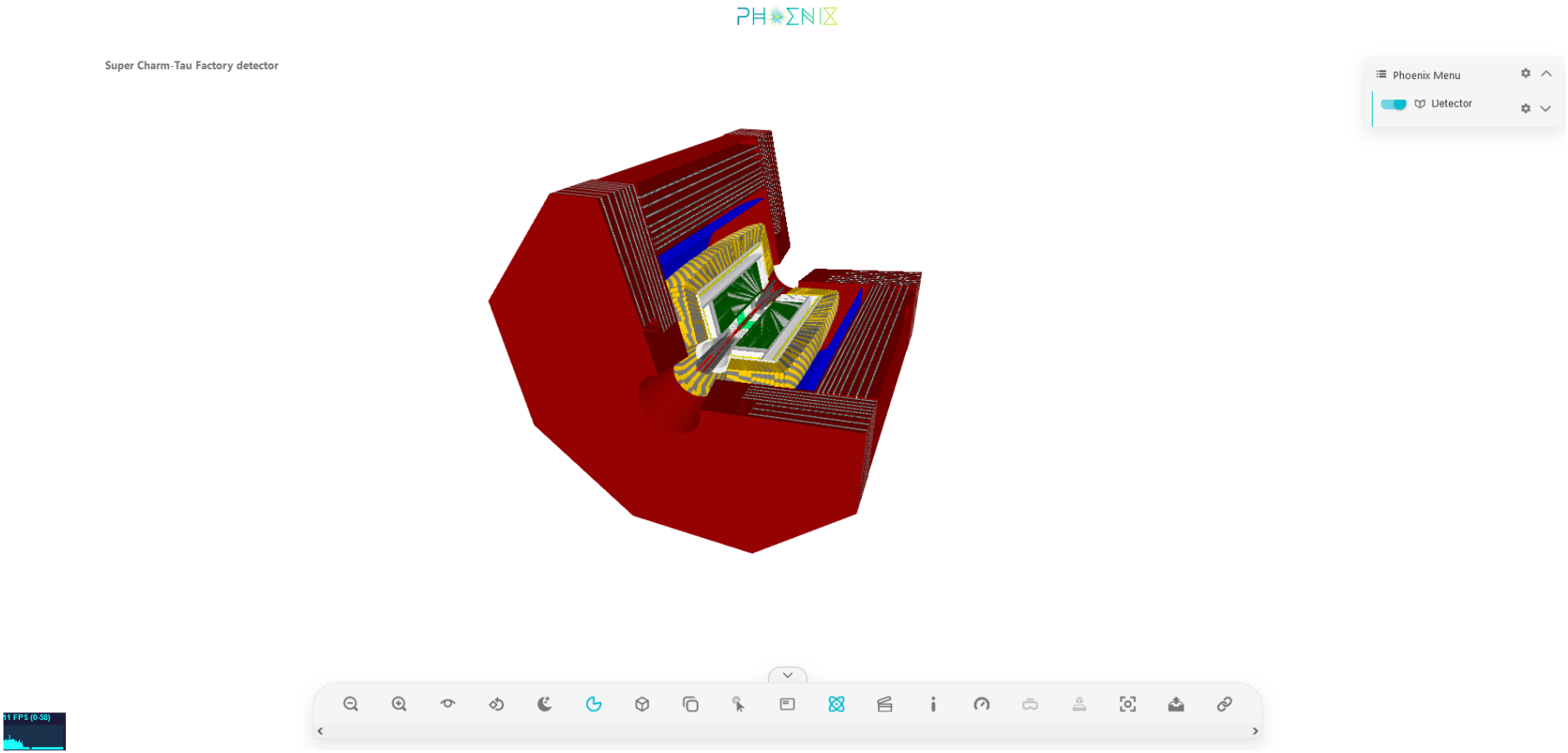


$\psi(4040) \rightarrow$ адроны

- Рисовалка геометрии стандартная из DD4Hep
- Рисовалка событий (на основе DD4Hep/DDEve) есть, но...

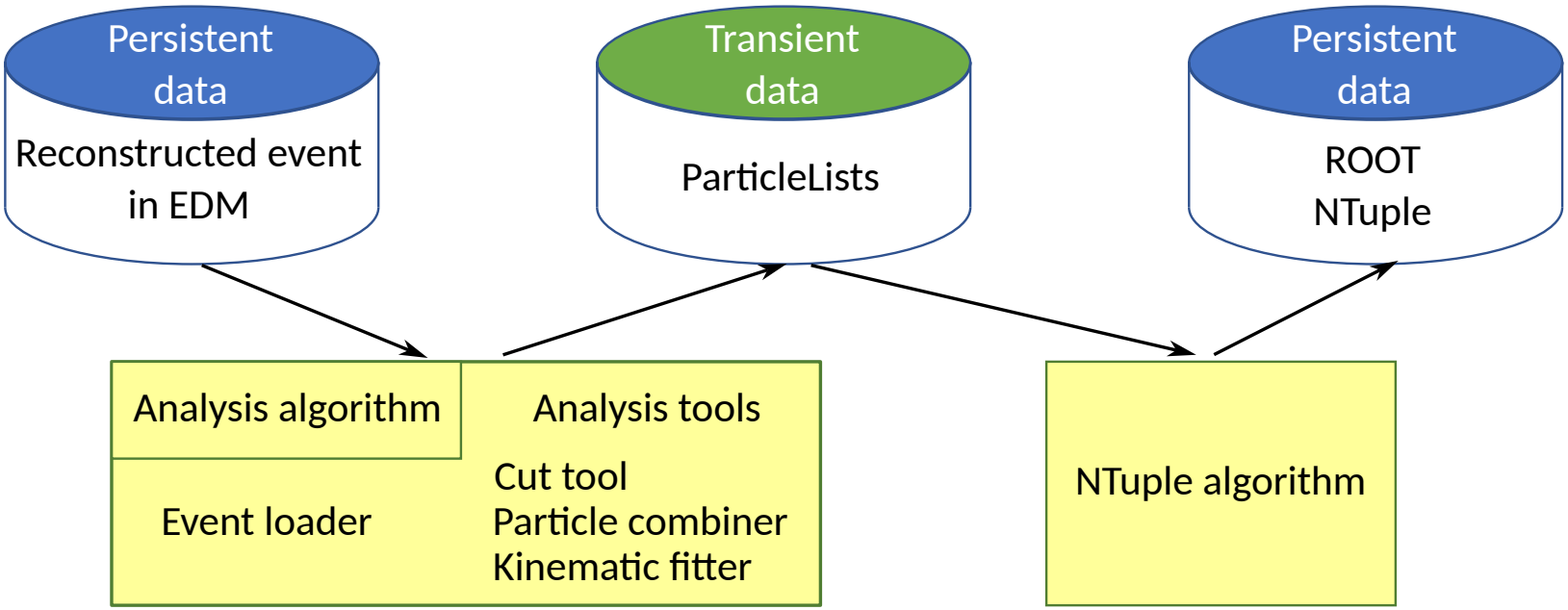
Система визуализации Phoenix

3D-картинка отображается в любом (современном) браузере, рисует как геометрию, так и события



Пакет анализа данных

- Следует подходам и решениям эксперимента Belle II к анализу



- обработка конечных частиц
- кинематические отборы
- восстановление распавшихся частиц
- расчёт кинематических параметров
- отбор частиц
- кинематические фиты
- сохранение результатов

Хранение данных: библиотека PODIO

PODIO — plain-old-data I/O.

- Библиотека на C++.
- Разработана как новая Модель данных для будущих экспериментов на основе опыта LHC и ILC.
- Использует простые структуры данных везде, где возможно, избегая глубокие и сложные иерархии наследования.
- Обеспечивает средства для создания связей между объектами и автоматического управления памятью.
- Потокобезопасна по построению.
- С 2024 года имеет стабильный API.
- По умолчанию поддерживает сохранение данных в ROOT, возможна работа с другими форматами, например, HDF5.

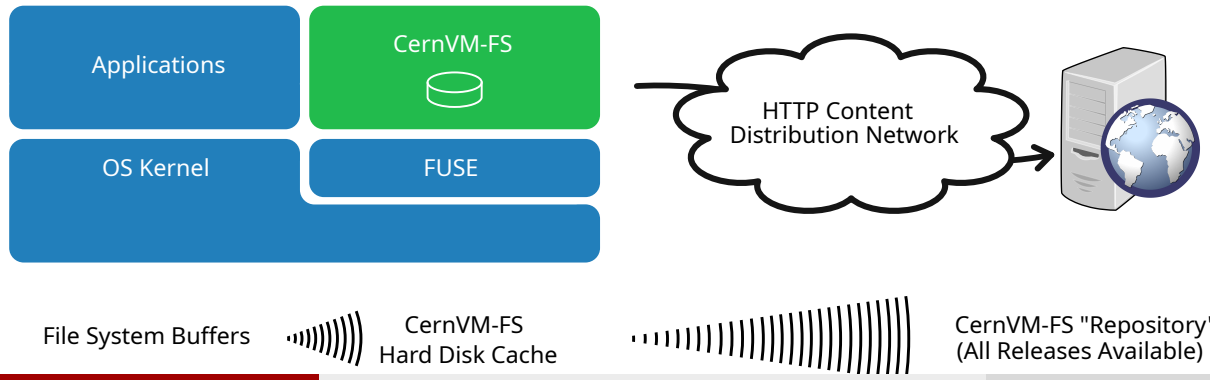
Распространение ПО

На данный момент:

- для внешнего ПО используется CernVM File System (CVMFS),
- для своего ПО — CephFS вычислительного кластера, в планах полный переход на CVMFS.

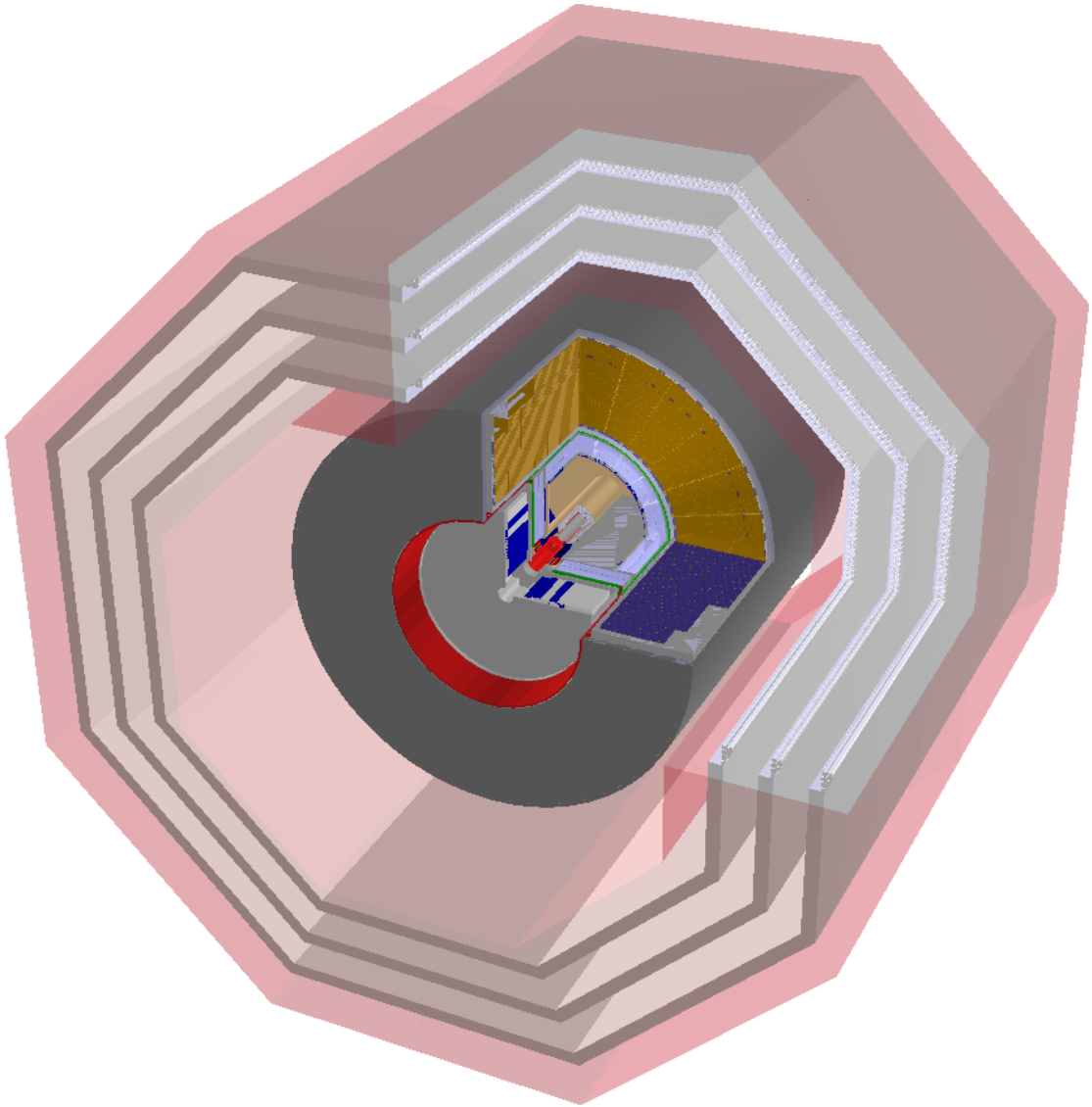
Файловая система CVMFS:

- работает через http, доступна только для чтения,
- данные и метаданные загружаются только по требованию и кэшируются,
- поможет, когда потребуется привлекать коллаборантов и/или считать в удалённых ВЦ,
- демонстрирует открытость эксперимента.



Применение для других экспериментов

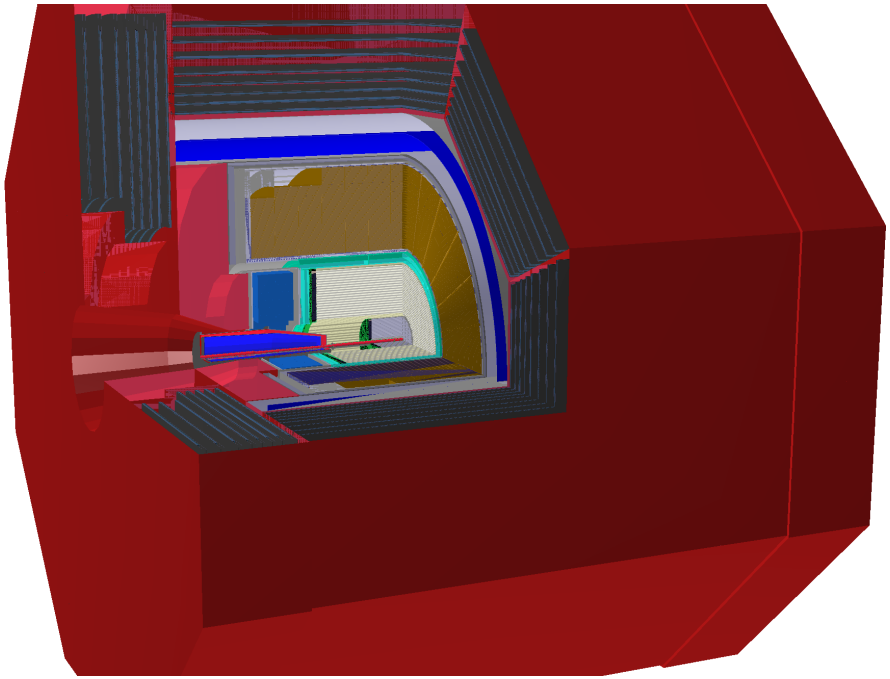
Детектор КЕДР



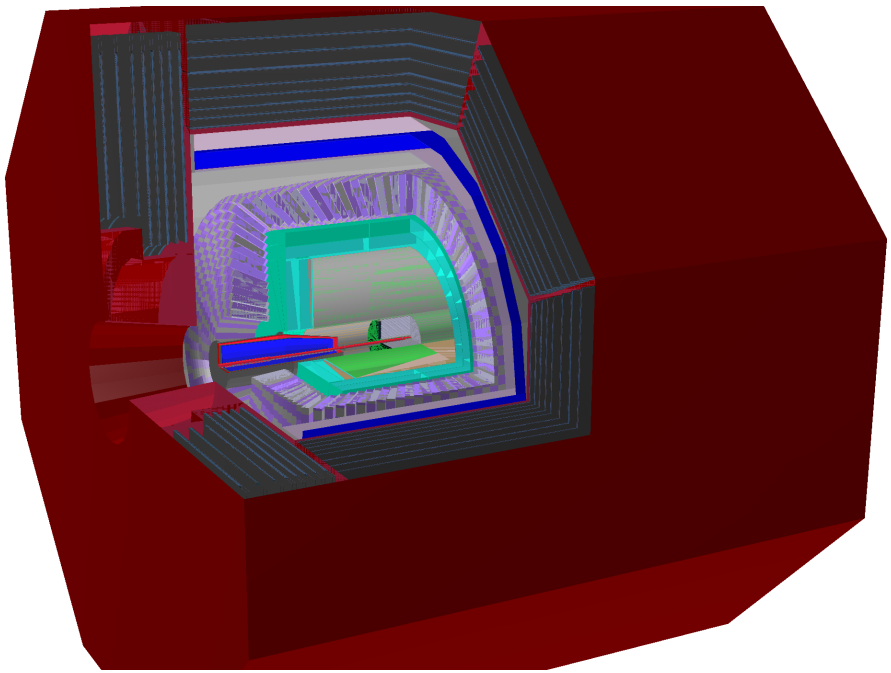
Применение для других экспериментов

Детектор для ВЭПП-6

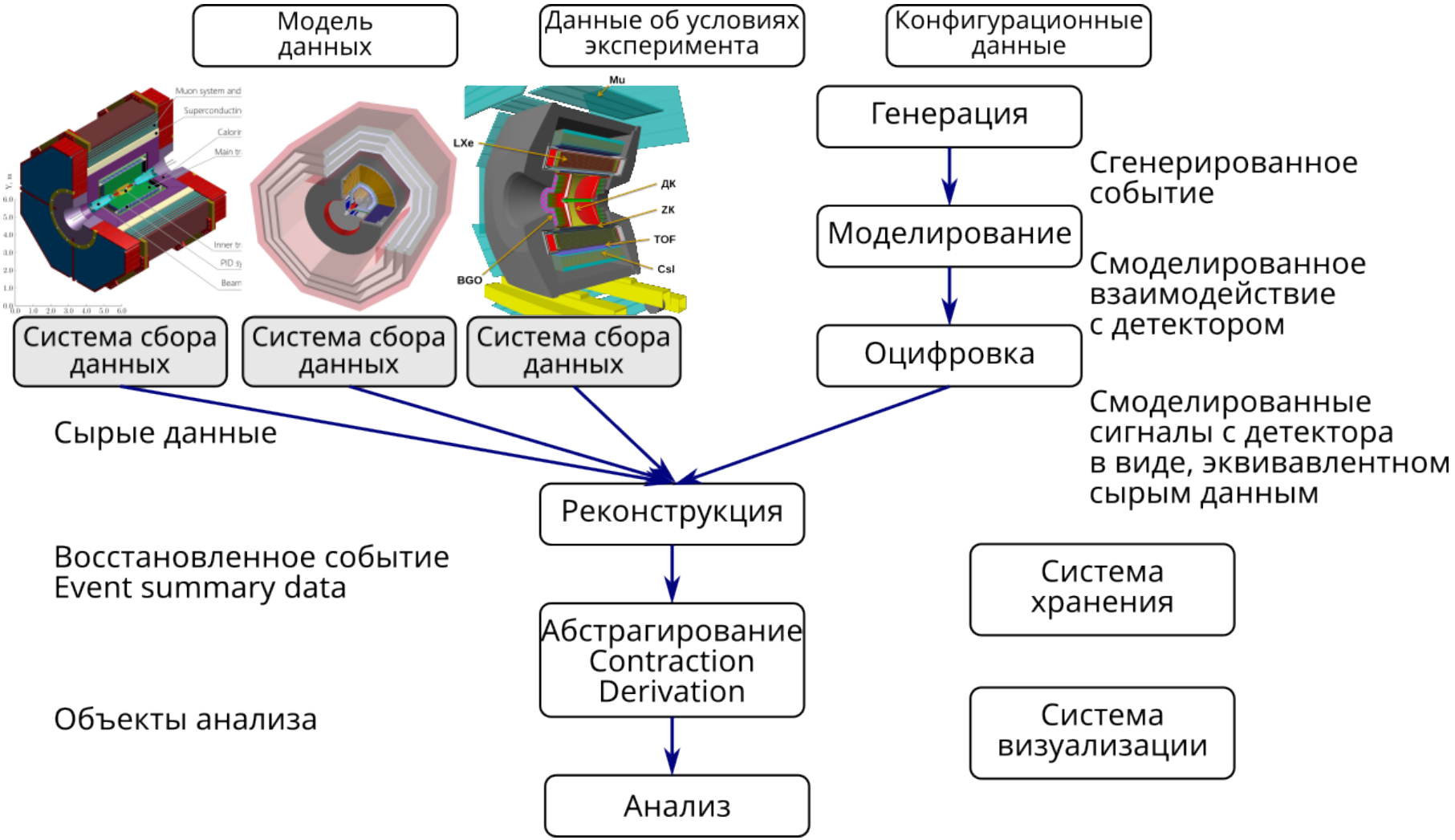
Фаза 1



Фаза 2



Аврора для разных экспериментов



Аврора

Что делается

Группа ПО:

- Адаптация к современным версиям внешнего ПО.
- Ужесточение проверок кода перед его принятием.
- Разделение на общую и экспериментоспецифичные части.
- Развитие служебных средств.
- Изучение вариантов геометрического движка.
- ConditionsDatabase — на аутсорсинг (Анисенков)?

Пользователи:

- Моделирование для ВЭПП-6 (Чиликин).
- Начаты работы по КМД-4 (Иванов).
- Работает группа черенковских детекторов (Анисимов).
- Начаты работы по КЕДР.

Аврора

Что делать

- Принятие общих подходов к построению ПО.
- Стабильный релиз Аврора, готовый к одновременному использованию группами разных экспериментов.
- Развитие инфраструктуры (CVMFS, CI).
- Участие специалистов по конкретным системам, по средствам обработки и анализа. Готовность следовать общим подходам.

Аврора

Что делать

- Принятие общих подходов к построению ПО.
- Стабильный релиз Аврора, готовый к одновременному использованию группами разных экспериментов.
- Развитие инфраструктуры (CVMFS, CI).
- Участие специалистов по конкретным системам, по средствам обработки и анализа. Готовность следовать общим подходам.

Спасибо за внимание

Программное обеспечение эксперимента ФВЭ

Любой эксперимент по ФВЭ требует полный набор соответствующего программного обеспечения:

- генераторы событий,
- параметрическое и полное моделирования детектора,
- алгоритмы реконструкции событий,
- модель данных события,
- интерфейс к базе данных условий эксперимента,
- интерфейс к системе хранения данных,
- алгоритмы анализа данных,
- система сборки и управления релизами.

Пример использования пакета анализа

Входные данные, алгоритм анализа

Создаём алгоритм анализа:

```
# Analysis algorithm.
analysis = AnalysisAlgorithm(
 'Analysis',
 eventLoader=event_loader,
 analysisTools=[cut1, combiner1, extrainfo1, fit1, combiner2, extrainfo2,
                fit2]
)
```

Алгоритм анализа последовательно вызывает инструменты анализа (AnalysisTool). Первым всегда вызывается загрузчик события, а остальная цепочка конфигурируется пользователем.

Пример использования пакета анализа

Загрузка события и отбор

Загрузчик события создаётся с помощью команды:

```
# Event loader.
event_loader = EventLoaderTool(
    'EventLoader',
    particleLists=['pi+', 'K+', 'gamma']
)
```

Аргумент particleLists задаёт списки частиц, которые нужно загрузить. Загружаются все частицы, на данном этапе не применяются критерии отбора. Античастицы загружаются автоматически. Критерии отбора применяются с помощью CutTool.

```
# Apply selection criteria for kaons.
cut1 = CutTool(
    'Cut1',
    outputParticleList='K+:sel',
    inputParticleLists=['K+'],
    cut='pt > 0.1'
)
```

Новый список `K+:sel` содержит Каоны из исходного списка `K+`, которые удовлетворяют условию $p_t > 0.1$ ГэВ/c.

Пример использования пакета анализа

Комбинации частиц

Составные частицы создаются при помощи ParticleCombinerTool из ранее созданных списков дочерних частиц:

```
# Reconstruct pi0 -> gamma gamma.
combiner1 = ParticleCombinerTool(
    'Combiner1',
    decayString='pi0 -> gamma gamma',
    cut='0.120 < M < 0.150'
)
```

На созданные частицы накладываются заданные критерии отбора:

```
# Reconstruct J/psi -> K+ K- pi0.
combiner2 = ParticleCombinerTool(
    'Combiner2',
    decayString='J/psi:channel1 -> K+:sel K-:sel pi0',
    cut='2.9 < M < 3.3 and p < 0.6'
)
```

Пример использования пакета анализа

ExtraInfo и кинематические фиты

ExtraInfoTool сохраняет значения заданных переменных в ExtraInfo для частиц из заданного списка. Например, можно сохранить значение инвариантной массы до фита:

```
# Save pi0 mass before fit.
extrainfo1 = ExtraInfoTool(
    'ExtraInfo1',
    particleList='pi0',
    extraInfo=['M', 'M']]
)
```

Первый аргумент в списке из двух — имя ExtraInfo, а второй — имя переменной.
KinematicFitterTool производит подгонку в массу для частиц из заданного списка:

```
# Perform mass fit for pi0.
fit1 = KinematicFitterTool(
    'Fit1',
    particleList='pi0'
)
```

Пример использования пакета анализа

Сохранение результатов

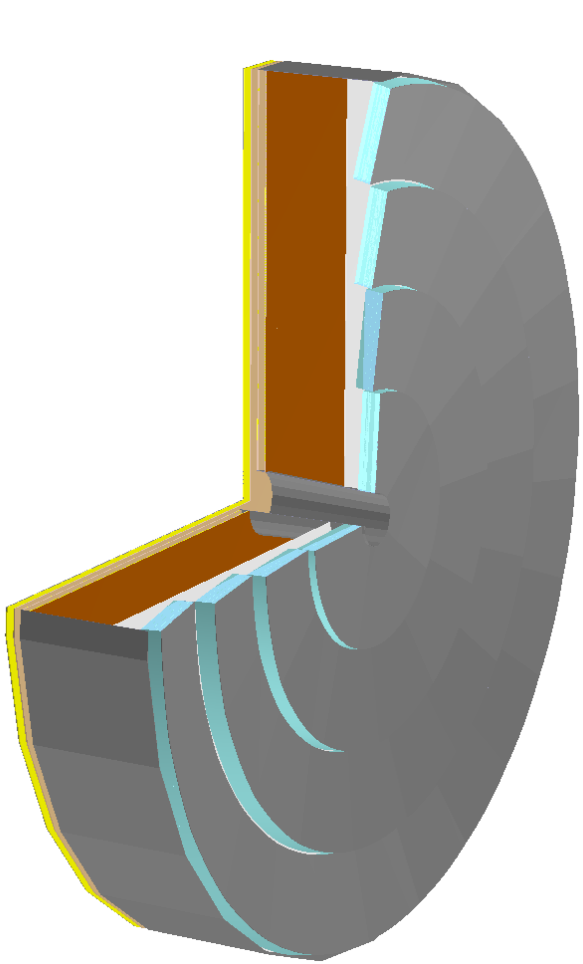
Реконструированные частицы сохраняются в выходной файл с помощью алгоритма NtupleAlg. Для самой частицы или её выбранных дочерних частиц задаются списки переменных, которые нужно сохранить.

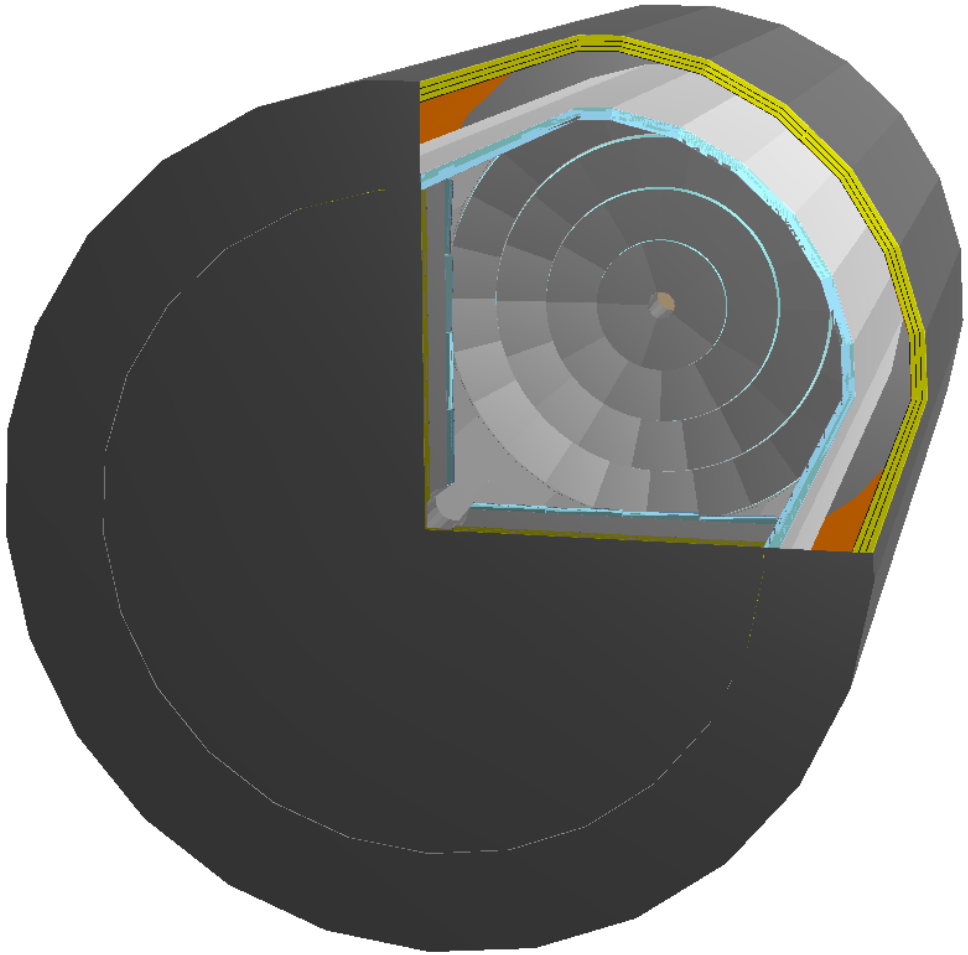
```
# Define aliases.
aliases = [['M_before_fit', 'extraInfo(M)'],
           ['p_before_fit', 'extraInfo(p)']]

# Create ntuple.
ntuple1 = NtupleAlg('NtupleAlg1',
    listName='J/psi:channel1',
    fileName='scttuple/jpsi1',
    tupleTitle='',
    vars=[['M_before_fit', 'p_before_fit', 'M', ''],
          ['E', 'px', 'py', 'pz', 'px_mc', 'py_mc', 'pz_mc', 'pidkpi',
           'J/psi -> ^K+ ^K- pi0'],
          ['E', 'px', 'py', 'pz', 'M_before_fit', 'M', 'J/psi -> K+ K- ^pi0'],
          ['px', 'py', 'pz', 'px_mc', 'py_mc', 'pz_mc',
           'J/psi -> K+ K- [ pi0 -> ^gamma ^gamma ]']],
    aliases=aliases
)
NTupleSvc(Output = [
    "scttuple DATAFILE='jpsi_kpkmpiz_reconstructed.root' OPT='NEW' TYP='ROOT'"
])
```

Применение для других экспериментов

Система FARICH для SPD (DD4hep)





Применение для других экспериментов

Система FARICH для SPD (GeoModel)