EventIndex

каталог событий

Осенняя Школа по информационным технологиям ОИЯИ

EventIndex

- Рассматриваем EventIndex в том виде в каком он был реализован для эксперимента Atlas на LHC, с прицелом на использование в установке SPD на коллайдере NICA.
- Несмотря на то что физические задачи этих экспериментов кардинально различны, системы обработки данных имеют сходные черты, что позволяет использовать похожие решения
- На SPD ожидается объемы данных того же порядка как на ATLAS при даже большем количестве событий
- Сходная модель данных событий и файловая организация
- Основное отличие: отсутствие триггера (но есть Online Filter)

Файлы

- Группы статистически эквивалентных событий хранятся в файлах на диске или на магнитной ленте.
  - события реальных данных, собранные при тех же условиях детектора
  - смоделированные события, созданные с помощью того же генератора и обработанные теми же версиями ПО
- Каждый файл обычно содержит от 1000 до 10000 событий, в зависимости от формата и балансировки :
  - слишком большое количества маленьких файлов (< 1 ГБ) могут вызвать проблемы с хранением данных
  - слишком большие файлы (> 10 ГБ) могут снизить эффективность передачи.

Датасеты и контейнеры

- Файлы группируются в датасеты (наборы данных)
  - Датасет как правило содержит события относящиеся к одному Run-у
    - Данные для длительных run-ов в процессе обработки как правило разбиваются на несколько датасетов
- Датасеты, в свою очередь могут быть сгруппированы в контейнеры
  - Контейнер может содержать датасеты относящиеся к одному Run-у, или например, к целому году набора данных
  - Один и тот же датасет может быть отнесен к нескольким контейнерам
- Пользователи могут создавать свои датасеты для нужд конкретного анализа

Датасеты в Atlas

- В зависимости от стадии обработки данные могут имет различные форматы
  - Изначально данные детектора записываются в формате RAW
  - После реконструкции событий получаются AOD датасеты.
  - Derivation (DAOD) - наборы производных данных для анализов
- + на Grid генерируются (MC) аналоги реальных событий
  - EVNT датасеты содержат информацию о смоделированных частицах
  - AOD и DAOD датасеты — аналогично как для реальных событий
- Различные версии датасетов происходящих от одних и тех же событий, полученных с детектора или смоделированных
  - Получены с различными настройками реконструкции и версиями ПО
  - Могут создаваться регулярно либо по мере обнаружения ошибок

EventIndex

- EventIndex – единый каталог всех событий, полученных, обработанных и смоделированных в эксперименте ATLAS
- В рамках проекта была создана инфраструктура, которая
  - предоставляет систему индексирования данных, хранящую указатели на события ATLAS в миллионах файлов разбросанных по сотням серверов глобальной распределенной вычислительной сети LHC GRID, а также основную информацию о событиях.
  - позволяет собирать и сохранять информацию о событиях, автоматически обнаруживать и индексировать новые данные.
  - предоставляет набор инструментов для доступа к этой информации из командной строки, графического интерфейса либо через RESTful API.
  - позволяет быстро и эффективно отбирать искомые события из миллиардов полученных в результате эксперимента и смоделированных событий, используя (в том числе) триггерную информацию.

Сценарии использования

- Отбор событий (Event Picking)
  - получить событие в нужном формате и обработанное конкретной версией программного обеспечения
- Подсчет и выборка события на основании триггерных решений
- Проверка целостности и согласованности данных
- Создание матриц перекрытия триггерных цепочек
- Создание матриц перекрытия датасетов
- Быстрый просмотр данных
  - Поиск датасетов, составление отчетов, проверка датасетов

Отбор событий / Event Picking

- Поиск событий среди сотен миллионов файлов разбросанных по серверам распределенной системы хранения данных
- Получение одного или нескольких событий
  - В нужном формате (AOE, DAOD_TOP2, RAW) и обработанное конкретной версией программного обеспечения
    - просмотреть на eventDisplay (и вставить в презентацию)
    - Исследовать свойства событий и проверить правильность процедуры восстановления
- Получение большого числа событий
  - Анализ набора событий отобранных согласно какому-то критерию
  - Получение случайного набора событий

Контроль согласованности данных

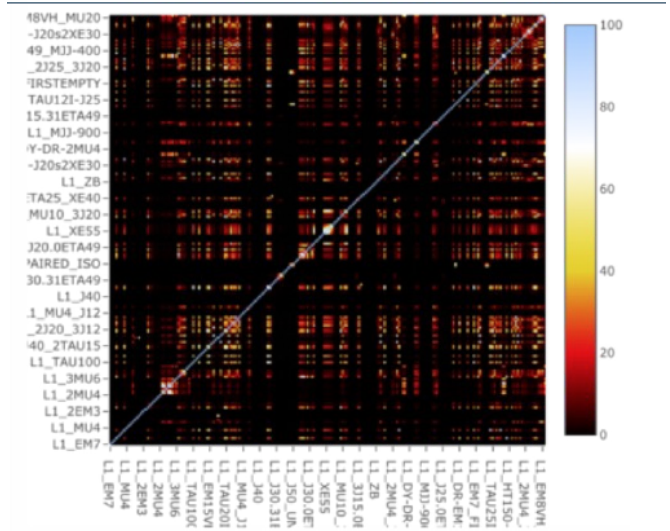
- В процессе обработки данных возможны сбои
  - Одни и те же события могут быть обработаны и/или записаны дважды с одними и тем же либо различными идентификаторами.
  - События может быть пропущены / потеряны
    - из-за ошибки ввода-вывода,
    - некорректной работы алгоритма
    - аварийного выхода программы
  - Особенно часто это происходит в первые годы эксперимента
- События могут повторяться/теряться при передаче данных
- Поломки оборудования также могут вызвать потерю данных
- На начальных этапах эксперимента доля «плохих» данных может быть значительной (десятки процентов)

Контроль согласованности данных

- Контроль согласованности данных при обработке
  - Проверка количества данных во входном и выходном датасете
  - Сравнение идентификаторов событий на входе и на выходе
- Индексирование новых «официальных» датасетов
  - Позволяет отлавливать ошибки доступа, каталогов ,поврежденные идентификаторы, etc… до начала физического анализа
  - Позволяет выявлять дублирующиеся данные
  - Позволяет выявлять проблемы в триггерной информации
- Контроль согласованности и целостности данных
  - НЕ позволяет контролировать физические параметры
  - НЕ гарантирует что обнаруженные ошибки будут исправлены

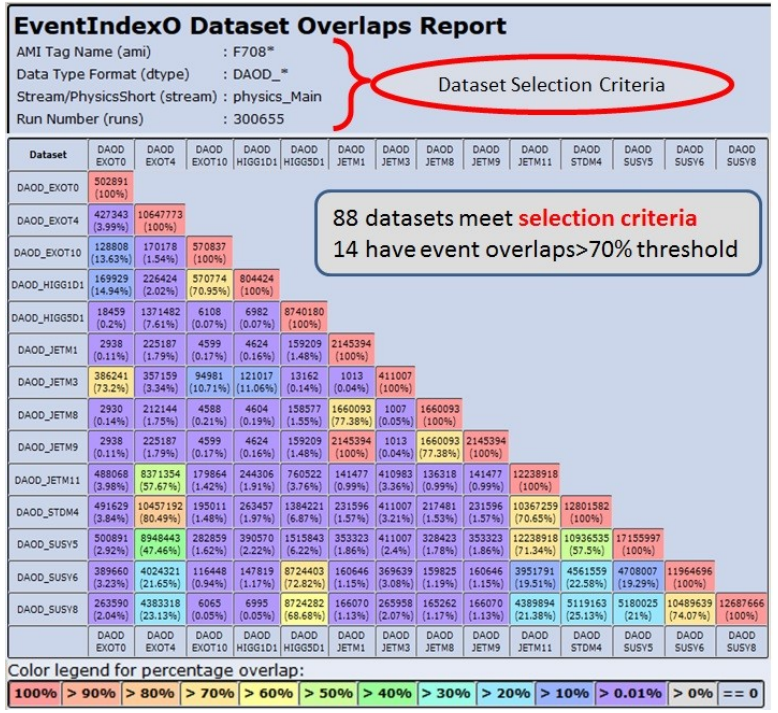
Триггерная информация

- Триггерная информация в EventIndex позволяет
  - Подсчитывать число событий удовлетворяющих определенным наборам триггерных условий (триггерным цепочкам)
  - Строить матрицы перекрытия триггерных цепочек (trigger overlaps)
    - Полезно для оптимизации триггерных таблиц и выбора триггерных условий для анализа.
  - Отбирать события на основании комбинации триггерной информации и других свойств (например мгновенной светимости)
    - Для подсчета частоты появления таких событий
    - Отобранные события можно использования в физических анализах



Перекрытия производных данных

- При создании производных ("derivation") датасетов используются все полностью реконструированные события из, например, AOD датасета, а на выход поступает только часть событий, пригодных для одного или нескольких анализов.
- EventIndex позволяет рассчитывать долю перекрытий наборов событий в DAOD датасетах и таким образом оптимизировать их распределение и использование дискового пространства.
  - Речь идет о петабайтах!



EventIndexO Dataset Overlaps Report

AMI Tag Name (ami) : F708*
Data Type Format (dtype) : DAOD_*
Stream/PhysicsShort (stream) : physics_Main
Run Number (runs) : 300655

Dataset Selection Criteria

88 datasets meet selection criteria
14 have event overlaps>70% threshold

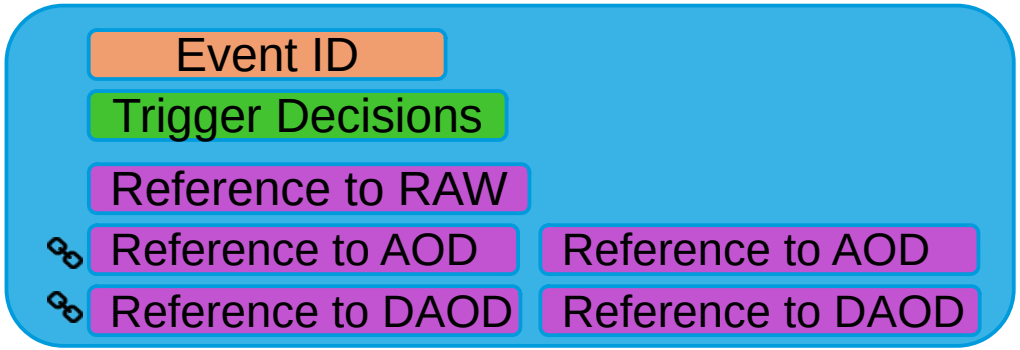
Dataset	DAOD EXOT0	DAOD EXOT4	DAOD EXOT10	DAOD HIGG1D1	DAOD HIGG5D1	DAOD JETM1	DAOD JETM3	DAOD JETM8	DAOD JETM9	DAOD JETM11	DAOD STDM4	DAOD SUSY5	DAOD SUSY6	DAOD SUSY8
DAOD_EXOT0	502891 (100%)													
DAOD_EXOT4	427343 (3.99%)	10647773 (100%)												
DAOD_EXOT10	128808 (13.63%)	170178 (1.54%)	570837 (100%)											
DAOD_HIGG1D1	169929 (14.94%)	226424 (2.02%)	570774 (70.95%)	804424 (100%)										
DAOD_HIGG5D1	18459 (0.2%)	1371482 (7.61%)	6108 (0.07%)	6982 (0.07%)	8740180 (100%)									
DAOD_JETM1	2938 (0.11%)	225187 (1.79%)	4599 (0.17%)	4624 (0.16%)	159209 (1.48%)	2145394 (100%)								
DAOD_JETM3	386241 (73.2%)	357159 (3.34%)	94981 (10.71%)	121017 (11.06%)	13162 (0.14%)	1013 (0.04%)	411007 (100%)							
DAOD_JETM8	2930 (0.14%)	212144 (1.75%)	4588 (0.21%)	4604 (0.19%)	158577 (1.55%)	1660093 (77.38%)	1007 (0.05%)	1660093 (100%)						
DAOD_JETM9	2938 (0.11%)	225187 (1.79%)	4599 (0.17%)	4624 (0.16%)	159209 (1.48%)	2145394 (100%)	1013 (0.04%)	1660093 (77.38%)	2145394 (100%)					
DAOD_JETM11	488068 (3.98%)	8371354 (57.67%)	179864 (1.42%)	244306 (1.91%)	760522 (3.76%)	141477 (0.99%)	410983 (3.36%)	136318 (0.99%)	141477 (0.99%)	12238918 (100%)				
DAOD_STDM4	491629 (3.84%)	10457192 (80.49%)	195011 (1.48%)	263457 (1.97%)	1384221 (6.87%)	231596 (1.57%)	411007 (3.21%)	217481 (1.53%)	231596 (1.57%)	10367259 (70.65%)	12801582 (100%)			
DAOD_SUSY5	500891 (2.92%)	8948443 (47.46%)	282859 (1.62%)	390570 (2.22%)	1515843 (6.22%)	353323 (1.86%)	411007 (2.4%)	328423 (1.78%)	353323 (1.86%)	12238918 (71.34%)	10936535 (57.5%)	17155997 (100%)		
DAOD_SUSY6	389660 (3.23%)	4024321 (21.65%)	116448 (0.94%)	147819 (1.17%)	8724403 (72.82%)	160646 (1.15%)	369639 (3.08%)	159825 (1.19%)	160646 (1.15%)	3951791 (19.51%)	4561559 (22.58%)	4708007 (19.29%)	11964696 (100%)	
DAOD_SUSY8	263590 (2.04%)	4383318 (23.13%)	6065 (0.05%)	6995 (0.05%)	8724282 (68.68%)	166070 (1.13%)	265958 (2.07%)	165262 (1.17%)	166070 (1.13%)	4389894 (21.38%)	5119163 (25.13%)	5180025 (21%)	10489639 (74.07%)	12687666 (100%)
	DAOD EXOT0	DAOD EXOT4	DAOD EXOT10	DAOD HIGG1D1	DAOD HIGG5D1	DAOD JETM1	DAOD JETM3	DAOD JETM8	DAOD JETM9	DAOD JETM11	DAOD STDM4	DAOD SUSY5	DAOD SUSY6	DAOD SUSY8

Color legend for percentage overlap:

100%	> 90%	> 80%	> 70%	> 60%	> 50%	> 40%	> 30%	> 20%	> 10%	> 0.01%	> 0%	== 0

Событие в EventIndex

- Запись о каждом событии содержит следующие поля:
  - Идентификаторы события
    - Run and event number
    - Trigger stream
    - Дополнительная информация
      - Luminosity block
      - Bunch Crossing ID (BCID)



  - Информация о триггерных решениях
    - Триггерные маски и декодированные триггерные цепочки
  - Указатели на файлы, в которых содержится информация о событии, в различных форматах и версиях обработки
    - Информация о файлах добавляется по мере накопления данных

Архитектура EventIndex

Индексирование первичных данных

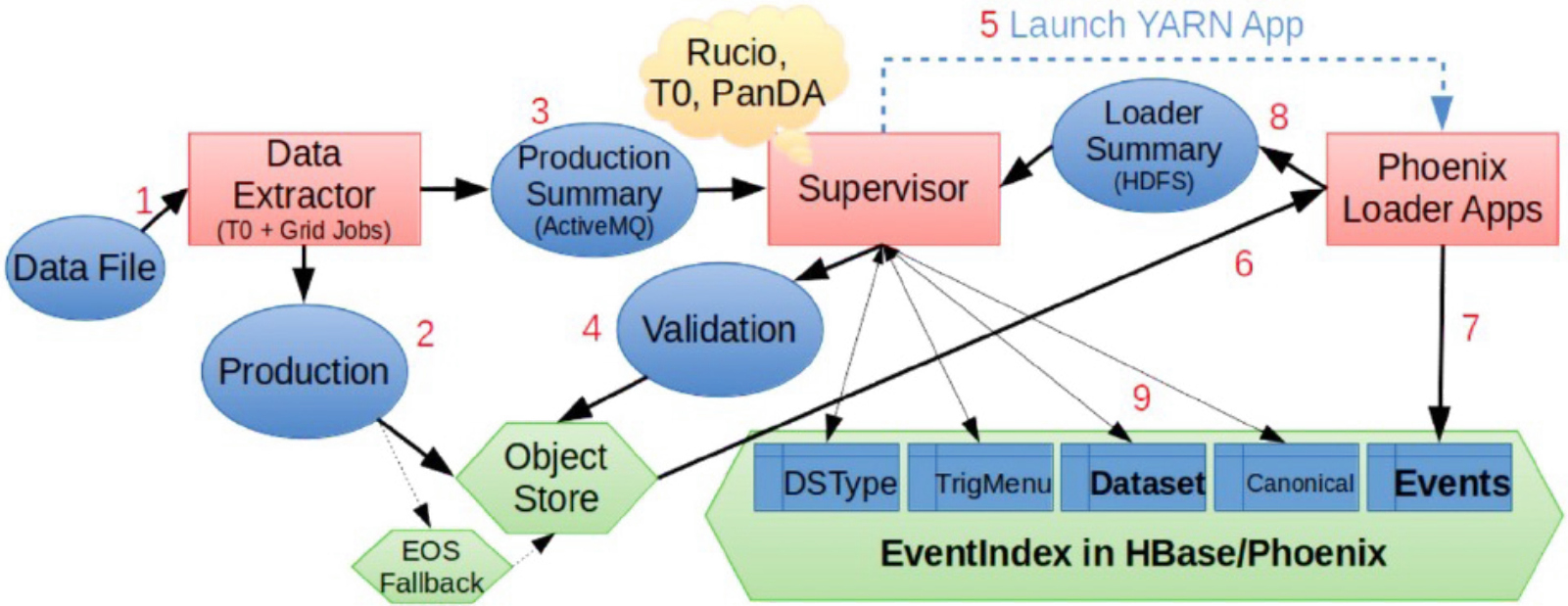
- Первичная обработка данных происходит на кластере Tier-0 в CERN. Реконструированные события записываются в AOD
- По готовности файлов, запускаются задачи (*Producer Transformation*) которые извлекают из них информацию для EventIndex.
  - Информация о RAW фаилах берется из дочерних AOD
- Producer сохраняет полученную информацию в виде файла
  - проверяется наличие дублирующихся событий
  - выходной файл отправляется во временное хранилище Object Store
  - Сообщение с отчетом о выполненной задаче и ссылкой на файл отправляется программе-диспетчеру EventIndex Supervisor

Индексирование данных на Grid

- EventIndex также собирает информацию о датасетах которые создаются в результате обработки и моделирования данных на серверах распределенной вычислительной системы Grid
- В качестве источника данных о появлении и характеристиках новых наборов данных используются информационные системы AMI и Rucio.
  - Ежедневно проводится опрос системы AMI на предмет появления новых датасетов, их характеристики уточняются с помощью Rucio.
- Полученных списки проверяются с отсевом: dataset может быть поврежден, неполон или быть вовсе пустым
- Отфильтрованные списки передаются в Supervisor

EventIndex Supervisor

- EI Data Collection Supervisor – программа которая контролирует получение данных EventIndex от producer-ов и организует их передачу в постоянное хранилище



Загрузка данных в хранилище

- Supervisor получает сообщения которые посылают задачи извлекающий информацию для EventIndex
  - Имя датасета, идентификаторы задачи, местоположение объекта с файлами eventIndex, id обработанных фаилов и число событий
- Проверяется соответствие этих данных с данными Rucio
- Supervisor запускает программу Loader, которая
  - считывает данные из Object Store ,
  - записывает их в таблицы событий EventIndex
  - Записывает отчет о выполнении в HDFS для Supervisor
- Supervisor записывает данные в таблицу датасетов
- Supervisor можно контролировать через web интерфейс

Хранилище данных EventIndex

- EventIndex разрабатывался в 2012-2013 гг для нужд Run 2 в качестве замены системы на основе Oracle, не справлявшейся с возросшими объемами данных.
- Для ядра системы было решено использовать нереляционные (NoSQL) базы данных из семейства Hadoop.
- При этом часть информации копировалась в таблицы Oracle для реалтзации быстрых запросов EventPicking
- Использование самых передовых на тот момент технологий обработки и хранения данных позволило обеспечить удовлетворительную производительность в течении всего периода работы начиная с запуска в середине 2015 года.

Хранилище данных EventIndex

- Однако ожидаемое существенное увеличение потока и темпа поступления данных в Run 3 и Run 4 потребовало соответствующей модернизации системы EventIndex с использованием новой платформы хранения данных
- Создана новая версия использующая Apache HBase для хранения данных и Apache Phoenix для организации запросов и записи данных
- Это позволяет объединить функции хранилища больших данных в HBase с возможностью выполнения SQL-запросов через интерфейс Phoenix

Apache HBase

- Apache HBase принадлежит к семейству HADOOP.
  - Нереляционная (NoSQL) база данных с открытым кодом
  - Распределенное и масштабируемое хранилище данных
- HBase организует данные в виде таблиц
  - Ряды таблицы имеют уникальный ключ-идентификатор
  - Каждый ряд может иметь свою схему
  - Данные в каждом ряду могут быть сгруппированы в заранее заданное семейства колонок
  - Доступ к значениям возможен по ключу и имени колонки

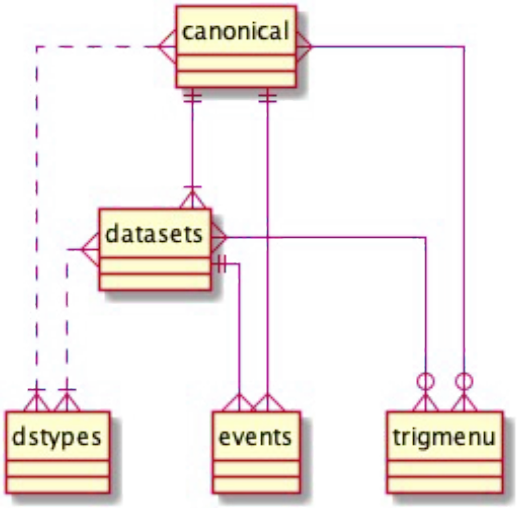
Apache HBase

- Свойства такой организации (RowKey):
  - Относительно небольшой размер
  - Позволяет уникально идентифицировать события
  - Позволяет производить поиск по интервалам
  - При росте таблиц используется гомогенное пространство RowKey
- Для тривиальной модели достаточно простой организации данных в виде пар ключ:значение, характерных для NoSQL
- Добавление функциональности SQL к HBase дает дополнительные преимущества:
  - Структурированные данные проще понимать и поддерживать
  - Стандартная логика запросов, пригодная для сложных запросов

Apache Phoenix

- Транзакционная система и операционная аналитика для HBase
  - Принимает SQL запросы
  - Преобразует их в серии сканирований в HBase
  - Напрямую использует API HBase, а также сопроцессоры и специальные фильтры
  - Выдает результаты запросов в стандарте JDBC
- Дизайн RowKey адаптируется к типам и размерам Phoenix
  - за счет небольшого уменьшения производительности
- Phoenix позволяет использовать поля RowKey в запросах, при этом они сохраняются как один объект в HBase

Схемы Phoenix для EventIndex



dstypes
• type : tinyint • name : varchar(20)
id : smallint

trigmenu
• smk : integer • type : tinyint • name : varchar(200)
id : smallint

events
• dspid : integer • dstypeid : smallint • eventno : bigint • seq : smallint
a.tid : integer a.sr : binary(32) a.mcc : integer a.mcw : float
b.pv : binary(34) ARRAY[]
c.lb : integer c.bcid : integer c.lpsk : integer c.etime : timestamp c.id : integer c.tbp : smallint[] c.tap : smallint[] c.tav : smallint[]
d.lb1 : integer d.bcid1 : integer d.hpsk : integer d.lph : smallint[] d.ph : smallint[]

datasets
• runno : integer • project : varchar(200) • datatype : varchar(200) • streamname : varchar(200) • prodstep : varchar(200) • version : varchar(200) • tid : integer • dspid : integer • dstypeid : smallint
smk : integer events_rucio : bigint files : integer events : bigint events_uniq : bigint events_dup : bigint files_dup : integer rank : smallint status : varchar(200) rucio_at : timestamp updated_at : timestamp dups_at : timestamp trigger_at : timestamp is_open : boolean is_deleted : boolean has_raw : boolean has_trigger : boolean prov_seen : smallint ARRAY[] sr_cnt : varchar(200) sr_clid : varchar(200) sr_tech : varchar(200)

canonical
• runno : integer • project : varchar(200) • datatype : varchar(200) • streamname : varchar(200) • prodstep : varchar(200) • version : varchar(200) • dspid : integer • dstypeid : smallint
smk : integer events_rucio : bigint files : integer events : bigint events_uniq : bigint events_dup : bigint files_dup : bigint rank : smallint status : varchar(200) rucio_at : timestamp updated_at : timestamp dups_at : timestamp trigger_at : timestamp is_open : boolean is_deleted : boolean has_raw : boolean has_trigger : boolean prov_seen : smallint ARRAY[] is_rucio_eq : boolean sr_cnt : varchar(200) sr_clid : varchar(200) sr_tech : varchar(200)

Структуры данных в HBase/Phoenix

- Ключи строк (Row keys). Они должны :
  - Содержать самую необходимую информацию
  - Иметь минимальновозможный размер
- В EI выбран формат: `dspid.dstypeid.eventno.seq`

ID датасета

тип данных

номер события

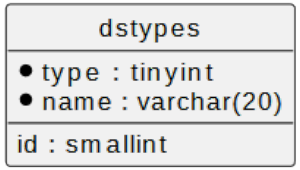
номер дубля события

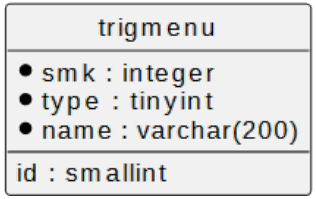
- Семества данных:
  - A: расположение события (и MC info)
  - B: происхождение события
  - C: информация триггера 1 уровня (L1).
  - D: данные триггера высокого уровня
    - (EF of HLT) и L2 для Run 1

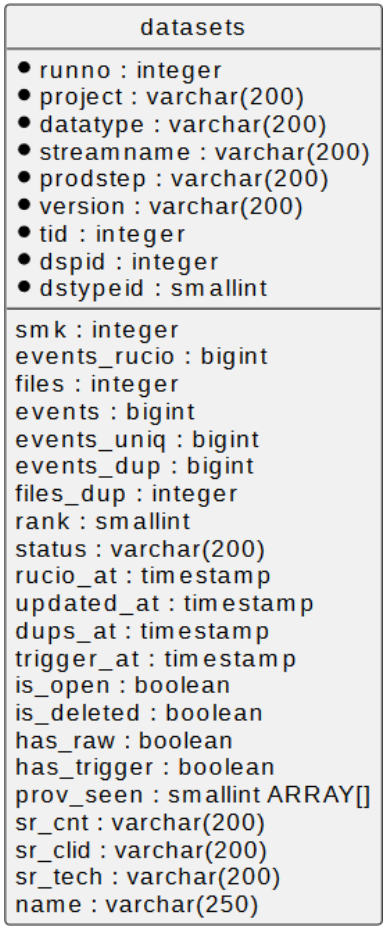
```
dspid           integer             NOT NULL ,
dstypeid        smallint            NOT NULL ,
eventno         bigint              NOT NULL ,
seq             smallint            NOT NULL ,

a.tid           integer                      ,
a.sr            binary(24)                   ,
a.mcc           integer,
a.mcw           float,

b.pv            binary(26) array             ,

c.lb            integer                      ,
c.bcid          integer                      ,
c.lpsk          integer                      ,
c.etime         timestamp                    ,
c.id            bigint                       ,
c.tbp           smallint array               ,
c.tap           smallint array               ,
c.tav           smallint array               ,

d.lb1           integer,
d.bcid1         integer,
d.hpsk          integer,
d.lph           smallint array,
d.lpt           smallint array,
d.lrs           smallint array,
d.ph            smallint array,
d.pt            smallint array,
d.rs            smallint array,

CONSTRAINT events_pk PRIMARY KEY
(dspid, dstypeid, eventno, seq)
```

Структуры данных в HBase/Phoenix

- Вспомогательные таблицы для хранения идентификаторов датасетов и данных учета
- Таблицы датасетов:
  - Расположение датасетов
  - Сгенерированные идентификаторы (dspid)
  - Статус импорта
  - Некоторая информация о метаданных: количество всех событий, уникальные события, дублирование
- Таблица типов данных:
  - типы данных
    - (RAW, EVENT, AOD, DAOD)
  - подтипы для производных







Event Lookup

- Клиент для поиска событий Atlas EventIndex (event lookup):
  - Пользователь запрашивает службу поиска событий, она отправляет запрос в EventIndex и возвращает htpekmnfn пользователю
- Пользователь предоставляет список событий для поиска
- Результат может быть представлен в различных форматах:
  - простой список
  - подробный список с дополнительными параметрами
  - JSON (полезно для вызова из программ, например из panDA)
- Пример:

```
$ event-lookup -c name 348895:3613825 -d AOD
348895 3613825 0d1c27ff-c5b2-6b46-a0fa-edbdc51c6b77 AOD data18_13TeV.00348895.physics_Main.merge.AOD.f937_m1972
348895 3613825 78c6837f-f448-e811-ae3c-44a8420a7621 RAW data18_13TeV.00348895.physics_Main.merge.AOD.f937_m1972
```

Web-интерфейс: поиск событий

Event Lookup

List of 'runnumber evtnumber' (-e)	00278880 558085589, 00278880 210318172
AMI tag (-p) (substring match)	
Stream name (-s)	Choose a stream name
Data type (-d)	Choose a data type
GUID type (-t)	○ AOD ○ ESD ○ RAW ◉ all
-api	○ simple ○ rich ◉ indexer ○ mc (indexer)
-details	☐ event ☑ type ☐ id ☐ dataset ☐ rich ouput
-email	(implies asynchronous execution)

Search Reset

-e 00278880 558085589, 00278880 210318172
-details typenullnullnullnull
-api indexer

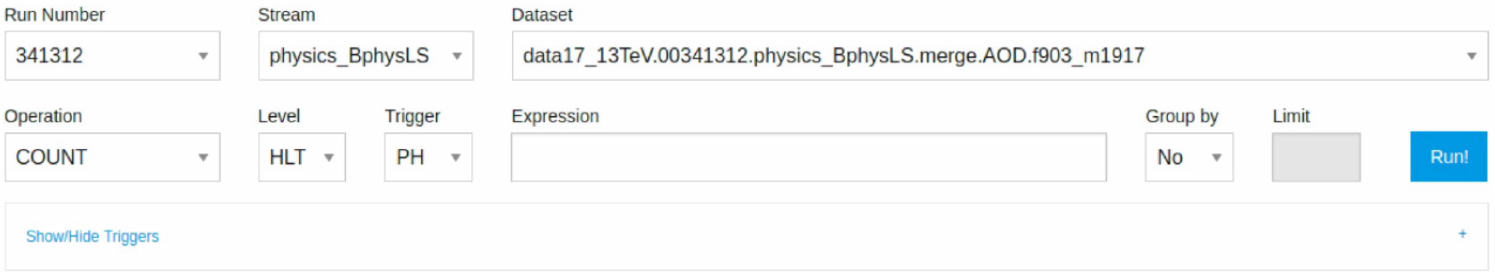
```
6 guids found for 1 runs with 2 events, 0 guids missing, 0s spend
148C774B-B5BB-BF41-9346-F82F213EC4A6 StreamDAOD_HIGG1D1
6C2A1CB8-8F8F-4B42-9FAF-AC55B0957CFD StreamAOD
886C7D3A-3E56-E511-9D00-44A8420A5EB7 StreamRAW
99ACB5FB-11FB-464F-B60F-E496213CABF1 StreamDAOD_HIGG1D1
EBC919C0-BAE8-9F42-BA63-21D782C0FAB7 StreamAOD
FC28C1E6-1D56-E511-AEDE-44A8420A8576 StreamRAW
```

Trigger Tool

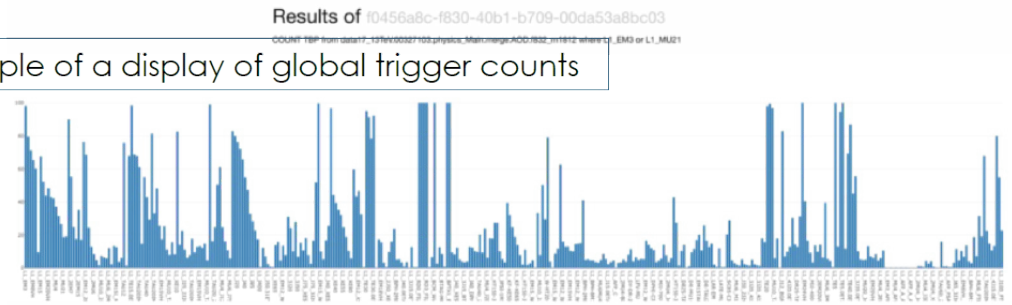
- Trigger Tool из предыдущего EventIndex:
  - Пользователь может искать события, которые удовлетворяют некоторым условиям запуска (прошли некоторые цепочки триггеров)
    - Пользователь может указать список цепочек триггеров для поиска
    - Также может быть предоставлен список вето
- Выполнение занимало существенное время, так как EventIndex должен был выполнять Map/reduce Job для огромного объема данных
  - Результатом является список событий
- Использовался для расчета матриц перекрытий триггеров
  - Выполнялся автоматически на новых наборах данных

Web-интерфейс: счетчик триггеров

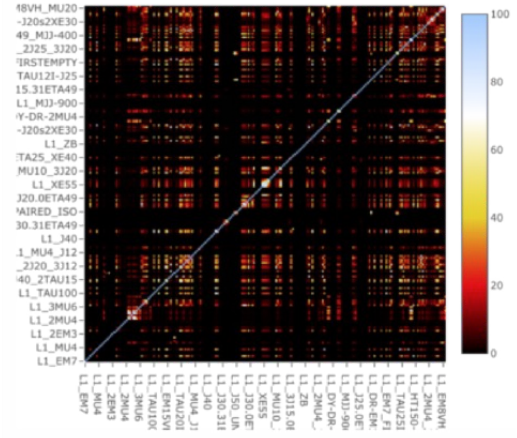
The web form to use the Trigger Counter



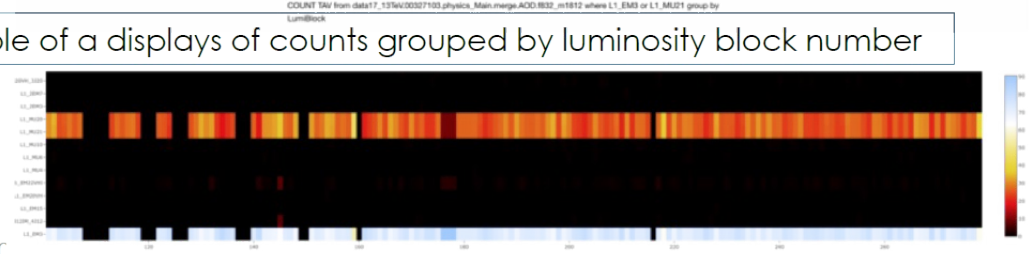
Example of a display of global trigger counts



The heat map of trigger overlaps within a given dataset



Example of a displays of counts grouped by luminosity block number

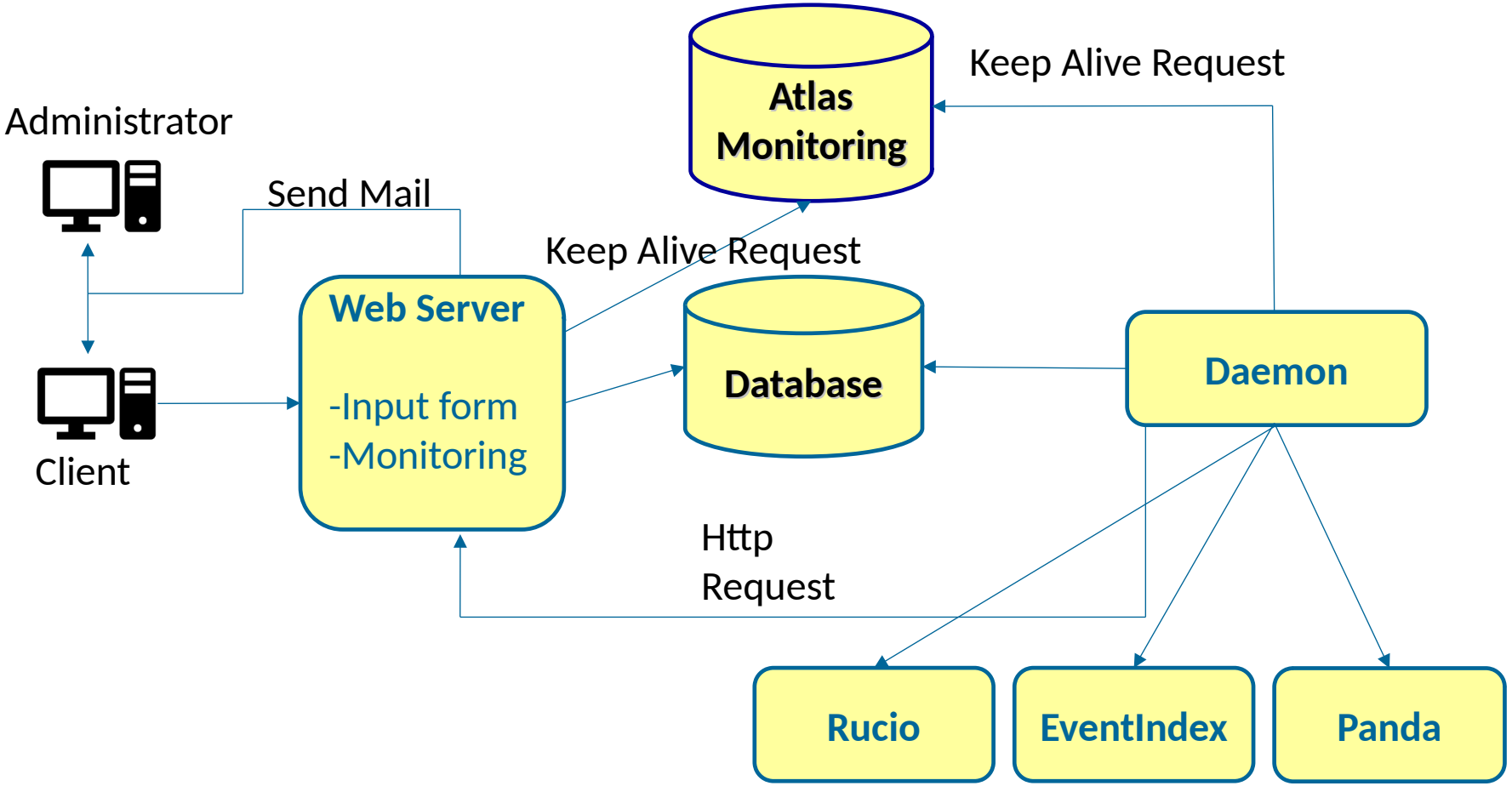


5 July 20

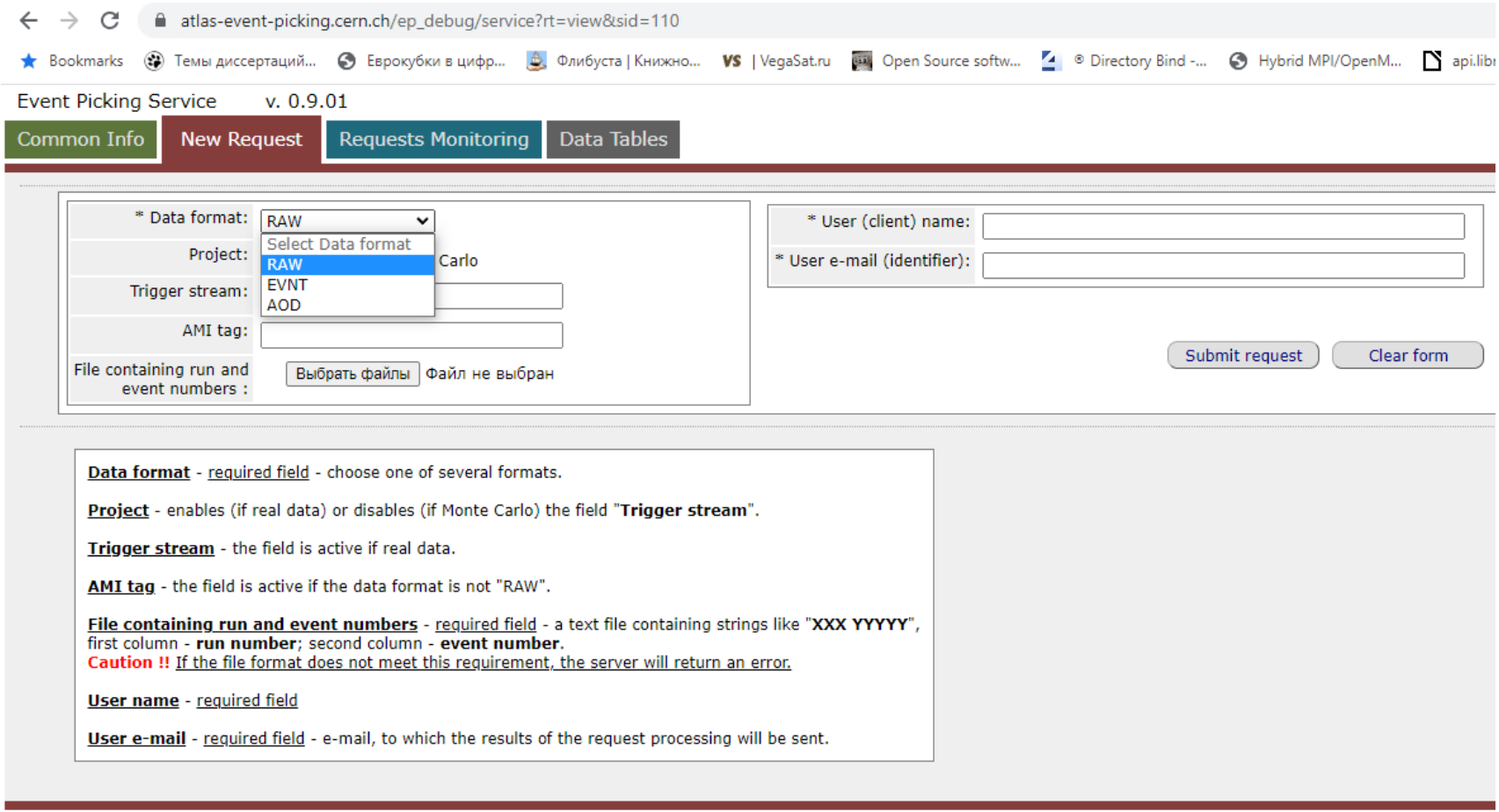
Event Picking Service

- Основная цель системы — автоматизировать операции, необходимые для массового выбора событий (event picking):
  - Графический интерфейс (GUI) получает от пользователя список событий и дополнительные параметры (тип данных для поиска, тип данных для извлечения, поток, тег AMI и т. д.
  - EPS разбивает список (списки) по run number, запрашивает интерфейс командной строки (EventIndex Query CLI), отправляет задания PanDA по сбору событий, повторяет их, если время ожидания истекло (например если события находятся на ленте), проверяет выходные данные, информирует пользователя о завершении.
  - Пользователь может следить за прогрессом с помощью графического интерфейса.

Архитектура Event Picking Service

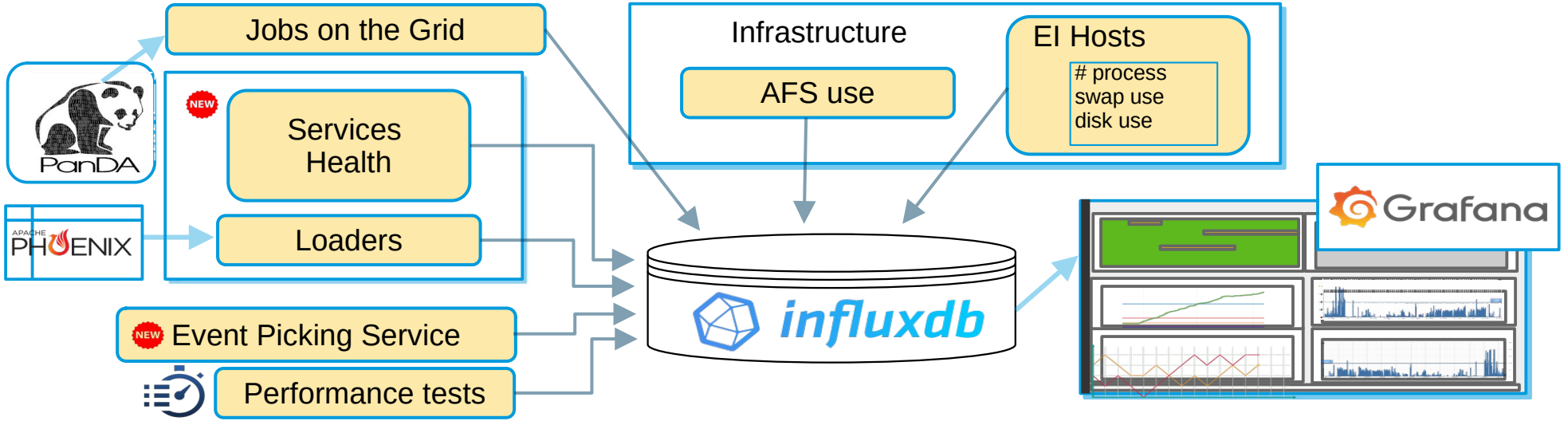


Архитектура Event Picking Service

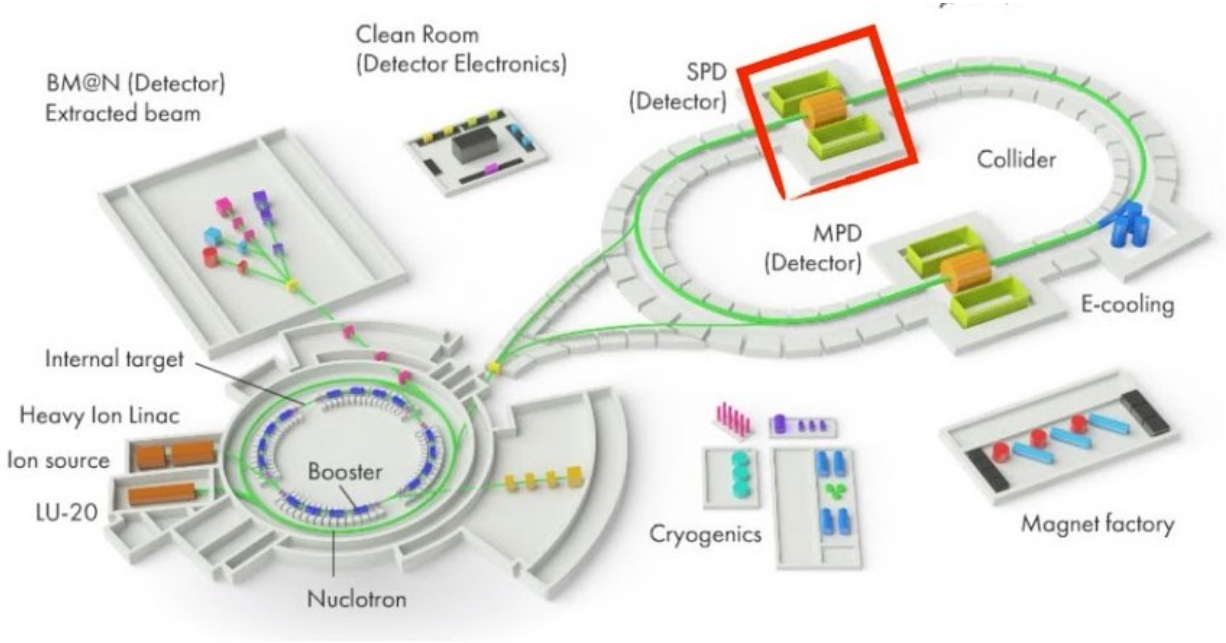


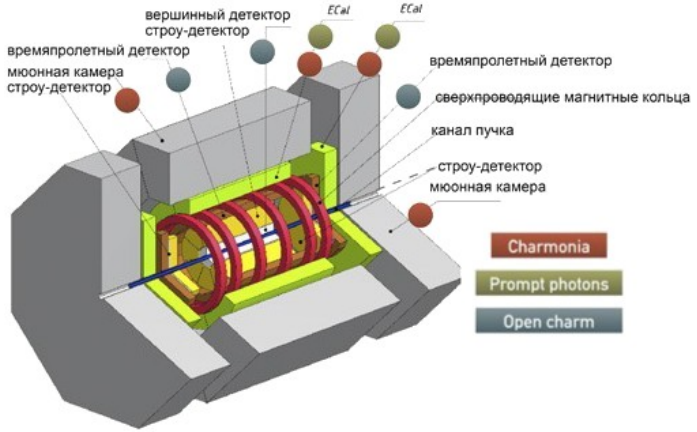
Мониторинг системы

- Данные о состоянии и производительности системы(~15k значений в день) собираются разными способами (сканирование журналов и HDFS, REST запросы и анализ WEB-страниц)
- Сбор и обработка информации управляется планировщиком
- Результаты загружаются в InfluxDB / Grafana через REST



Spin Physics Detector





SPD: количество событий

- Предполагаемые объемы данных с SPD достаточно велики
  - 20 ГБ/с => ~ или 200 ПБ/год (необработанные (RAW) данные)
  - ~30 миллиардов событий в год
  - это величины того же порядка или больше, чем в ATLAS
- После реконструкции и обработки для использования в физическом анализе добавятся еще несколько экземпляров каждого события
- Дополнительно ожидается сравнимый объем данных моделирования (MC)
- Данные о событиях в различных форматах записываются в файлы, организованные в датасеты различных типов

Форматы данных SPD: Real Data

- Данные временных слайсов, записанные в формате RAW
  - Временные данные, нет нужды индексировать
- После онлайн-фильтра → формат filtered RAW:
  - Результаты быстрой реконструкции событий,
  - распакованные RAW хиты, блоки RAW данных
- После оффлайн реконструкции → датасеты AOD и ESD:
  - AOD: Физические объекты: частицы, треки, кластеры …
  - ESD: + необработанные хиты
- Деривации для использования в конкретных анализах (DAOD)
  - Десятки типов
  - Как правило с меньшим числом событий и размером события

Форматы данных SPD: MC Data

- Генератор событий создает датасеты EVGEN
  - Содержат "истинные" параметры частиц (MC truth)
- После моделирования → RAW MC события
  - MC truth + распакованные RAW хиты
- После оффлайн реконструкции → датасеты AOD и ESD
  - То же, что и для реальных данных + MC truth
- Как и для реальных данных, деривации (DAOD)
- Данные моделирования появятся задолго до запуска установки, что позволит провести тестирование системы.

SPD needs EventIndex

- Регулярно будет производиться поворотная обработка данных, каждый раз добавляя новые экземпляры датасетов
- Данные будут распределенный по вычислительным серверам, их число будет меньше чем на LHC
- Для эффективного доступа к событиям, необходима база данных база данных, содержащая ссылки на экземпляры событий в распределенном хранилище данных
- То есть, **EventIndex**
- При разработке EventIndex предполагается опираться на опыт работающей на Atlas системы, адаптируя его к реалиям SPD

Сценарии использования

- Отбор событий (Event Picking)
  - получить событие в нужном формате и обработанное конкретной версией программного обеспечения
- Подсчет и выборка события
  - на основании параметров событий
  - на основании данных онлайн триггера
- Проверка целостности и согласованности данных
- Создание матриц перекрытия датасетов
- Быстрый просмотр данных
  - Поиск датасетов, составление отчетов, проверка датасетов
- Создание виртуальных датасетов 

Запись в SPD EventIndex (Event Record)

- Идентификаторы события
  - Run number
  - Event number
  - Frame ID
  - Bunch Crossing ID (BCID)
- Информация онлайн триггера
- Указатели на постоянные файлы, содержащие информация о событии, в различных форматах и версиях обработки
  - Информация о файлах добавляется по мере накопления данных
- Параметры события (например поляризация)

Пример записи EventIndex

- Первичные поля (при создании записи)
  - ‘RunNumber’ integer YYMMDDhh[0..9] 280308060
  - ‘EventNumber’ integer [-2147483648 .. +2147483647] 474 836
  - ‘FrameID’ integer [-2147483648 .. +2147483647] 5 000
  - ‘OFStream’ varchar(128) ’Main’
  - OPTIONAL ‘OFData’ JSONB …
  - ‘GUIDRaw’ UUID ’78c6837f-f448-e811-ae3c-44a8420a7621’
- Дополнительные (по мере обработки и создания датасетов)
  - ‘GUID_AOD’ UUID ’0d1c27ff-c5b2-6b46-a0fa-edbdc51c6b77’
  - OPTIONAL ‘ImportantValue’ real 3.14
  - ‘GUID_DAOD_PHYS’ UUID ’aa321c54-ca5e-b748-845d-02f30074f8de’
  - OPTIONAL ‘ImportantNumber’ smallint 42

Выбор платформы

- Разработку EventIndex естественно начать с выбора платформы для хранения и управления данными.

- Нужна Open Source СУБД позволяющая быстро и надежно оперировать терабайтными объемами данных
- Рассматриваются несколько вариантов
  - ClickHouse от Яндекс
    - столбцовая СУБД для онлайн обработки аналитических запросов (OLAP)
  - Apache HBase
    - распределенная, колоночно-ориентированная, мультиверсионная база типа «ключ-значение».
  - PostgreSQL
    - объектно-реляционная система управления базами данных (СУБД)

Особенности ClickHouse

- Столбцовое хранение данных — данные считываются только из нужных колонок, и однотипная информация эффективно сжимается
- Поддержка приближённых вычислений на части выборки — снижается число обращений к жёсткому диску, что ещё больше повышает скорость обработки данных;
- распараллеливание операций как в пределах одного сервера на несколько процессорных ядер, так и в рамках распределённых вычислений на кластере за счёт механизма шардирования;
- линейная масштабируемость — есть возможность построить кластер очень большого размера

Недостатки ClickHouse

- Разреженный индекс делает ClickHouse плохо пригодным для точечных чтений одиночных строк по своим ключам.
- Возможность изменять или удалять ранее записанные данные с низкими задержками и высокой частотой запросов не предоставляется. Есть массовое удаление и изменение данных для очистки более не нужного
- Отсутствие уникальности первичных ключей, что вместе с предыдущим пунктом в нашем случае осложняет добавление новых полей в EventRecord

Apache HBase

- Данные организованы в таблицы, проиндексированные первичным ключом, который в Hbase называется RowKey.
- Для каждого RowKey ключа может храниться неограниченны набор атрибутов (или колонок).
- Колонки организованны в группы колонок, называемые Column Family.
  - Как правило в одну Column Family объединяют колонки, для которых одинаковы паттерн использования и хранения.
- Для каждого атрибута может храниться несколько различных версий. Разные версии имеют разный timestamp.

Apache HBase

- Записи физически хранятся в отсортированном по RowKey порядке. При этом данные соответствующие разным Column Family хранятся отдельно, что позволяет при необходимости читать данные только из нужного семейства колонок.
- Атрибуты, принадлежащие одной группе колонок и соответствующие одному ключу физически хранятся как отсортированный список. Любой атрибут может отсутствовать или присутствовать для каждого ключа, при этом если атрибут отсутствует — это не вызывает накладных расходов на хранение пустых значений.
- Список и названия групп колонок фиксирован и имеет четкую схему.

PostgreSQL

- PostgreSQL это а объектно-реляционная СУБД с открытым исходным кодом.
  - Поддержка многочисленных типов данных
    - Численные, булевые, символьные, составные, сетевые типы данных, перечисление, типы «дата/время», массивы, etc.
  - Поддержка JSON, что позволяет использовать schema-less данные
    - Встроенные специализированные JSON операторы и функции
  - Поддержка пользовательских объектов и их поведения, включая типы данных, функции, операции и индексы.
  - Индексирование: частичное, функциональное, GiST, GIN, BRIN
  - Функции виртуальных таблиц, Материализованные представления
  - Возможность добавления/изменения столбцов

PostgreSQL

- Обеспечивает высокую надежность и производительность:
  - соответствие принципам ACID (атомарность, изолированность, непротиворечивость, сохранность данных)
  - Многоверсионный контроль конкурентных транзакций и изоляция транзакций
    - возможность изменения баз данных одновременно разными пользователями.
    - минимизирует блокировки данных и позволяет увеличить производительность
  - Журналы опережающей записи (**W**rite **A**head **L**ogging)
    - фиксирующая все изменения до их фактического применения.
  - Резервное копирование и восстановление
  - Возможность восстановления базы данных Point in Time Recovery
    - Откат состояние базы данных к предыдущему стоянию используя WAL.

- Основные ограничения

Максимальный размер БД	Неограничен
Максимальный размер таблицы	32 TB
Максимальный размер строки	1.6 TB
Максимальный размер поля	1 GB
Максимальное количество строк в таблице	Неограниченно
Максимальное количество столбцов в таблице	250-1600
Максимальное количество индексов в таблице	Неограничено