

EVENTINDEX — КАТАЛОГ СОБЫТИЙ ЭКСПЕРИМЕНТА SPD

Ф. В. Прокошин^{1,*}, *И. В. Тваури*^{2,**}, *З. А. Будтуева*^{2,***},
Р. З. Гурицев^{2,****}, *А.-Б. В. Газзаев*^{2,*****}

¹ Объединенный институт ядерных исследований, Дубна

² Северо-Осетинский государственный университет, Владикавказ, Россия

В ходе реализации эксперимента SPD планируется набрать до триллиона событий (записей результатов столкновений), для хранения и обработки которых потребуются сотни петабайт данных. Эта информация будет распределена между несколькими хранилищами данных в компьютерных центрах. Для исключения потери данных и повышения производительности записи, относящиеся к одному событию, будут дублироваться. Для эффективного доступа ко всем экземплярам событий необходима информационная система, а именно разрабатываемый SPD EventIndex — каталог всех событий, полученных от детектора или смоделированных, постоянно хранящихся во всех форматах и версиях.

The SPD experiment has to collect a large amount of data: up to trillion events (records of collision results) have to be stored and analyzed, producing tens of petabytes of data. This information will be distributed among a number of computing sites on various storage locations, with duplication to avoid data loss and improve performance. An information system is necessary to efficiently access all instances of the events, and the SPD EventIndex is being developed for this purpose. It is a catalog of all events obtained from the detector or simulated, in all permanent instances of different formats and versions.

PACS: 07.05.Hd; 29.85.Ca

ВВЕДЕНИЕ

Эксперимент SPD [1] представляет собой исследовательский проект, нацеленный на изучение физики элементарных частиц и поиск новых фундаментальных знаний о взаимодействиях и структуре частиц. Главной целью данного проекта является анализ данных, полученных с детектора SPD, установленного на ускорителе частиц NICA, создаваем-

* E-mail: prof@jinr.ru

** E-mail: ateia@mail.ru

*** E-mail: zbudtueva@list.ru

**** E-mail: vilovnok@gmail.com

***** E-mail: gazzaev1999@yandex.ru

мом в Объединенном институте ядерных исследований, расположенном в г. Дубне Московской области.

Для хранения и обработки большого объема данных, полученных в ходе эксперимента SPD, потребуется создание сложных информационных систем. Одной из таких систем должен стать EventIndex — каталог физических событий, полученных с установки или смоделированных для анализа и обработки данных.

ИНФОРМАЦИОННАЯ ЭКОСИСТЕМА SPD

На данном этапе создания эксперимента SPD информационные и вычислительные системы находятся в начальной стадии разработки. Создание EventIndex планируется начать с частей, которые не зависят от других компонентов.

SPD EventIndex разрабатывается как комплексная информационная система, которая должна обеспечить (рис. 1):

- получение информации о событиях эксперимента и смоделированных данных путем индексирования файлов данных, содержащих информацию об этих событиях;
- передачу этой информации и запись в базы данных;
- доступ к информации для программ обработки и анализа данных через API и приложения;
- доступ к информации пользователям через интерактивные и асинхронные интерфейсы.

Выбор платформы для хранения и управления данными производился исходя из предполагаемых потоков и объемов информации, содержания записи и ожидаемых способов применения EventIndex.

Предполагаемый поток данных на выходе онлайн-фильтра составит от десятков тысяч событий в секунду на ранней стадии до 150 тыс. при максимальной производительности установки, что даст нам от сотен

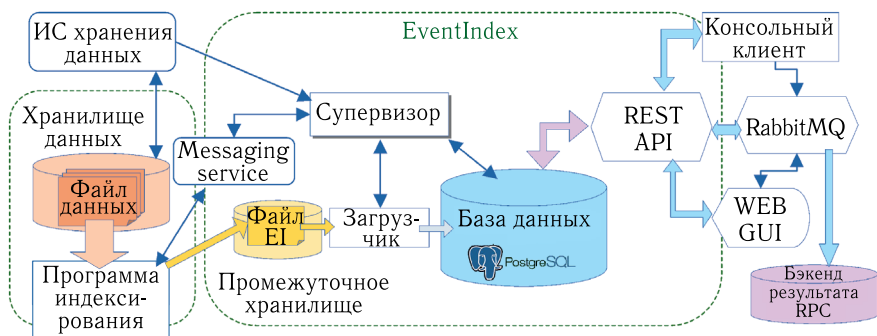


Рис. 1. Архитектура и схема потоков данных SPD EventIndex

миллиардов до нескольких триллионов событий в год. Также ожидается сопоставимый объем смоделированных данных.

Запись для события в EventIndex должна содержать следующие поля:

- идентификаторы события: номер Run-а (run_number) и номер события в Run (event_number);
- информация о решениях онлайн-фильтра в виде битовой маски (olf_result);
- уникальный идентификатор файла с сырыми (RAW) данными, в котором содержится данное событие (fileid_raw). С помощью файла UUID можно получить доступ к нему через распределенную систему хранения;
- идентификатор набора данных (dataset), в который входит этот файл (dsid_raw).

По мере обработки данных будут создаваться новые экземпляры восстановленных событий в формате, оптимизированном для физического анализа (AOD). Указатели на различные версии таких файлов будут добавляться в запись события. Также в запись могут быть добавлены важные параметры события, используемые для классификации и отбора.

Для надежного хранения и обработки структурированных данных была выбрана база данных PostgreSQL [2], особенностями которой являются многопоточность и способность обрабатывать большие объемы данных.

В рамках исследования был разработан и реализован удобный и эффективный программный интерфейс, осуществляющий обмен данными с помощью RESTful API [3]. Для обеспечения гибкости и надежности серверной части был выбран Flask [4] — микрофреймворк для языка Python. Фронтенд-часть клиентского интерфейса была разработана с использованием фреймворка Angular [5], предоставляющего инструменты

Request id	Client surname	Client e-mail	format	version	created	fileid
8	aleksand	aleksand@jinr.ru	RAW	r2345	2023-07-04 21:36:49	b6cc6e91-397d-4920-8748-85a446a5dcf8
9	aleksand	aleksand@jinr.ru	AOD	r7535	2023-07-04 21:52:17	b6cc6e91-397d-4920-8748-85a446a5dcf8
10	Semen Turchikhin	turchin@jinr.ru	AOD	r2345	2023-07-06 20:32:48	b6cc6e91-397d-4920-8748-85a446a5dcf8
11	fedor_123	prof@jinr.ru	RAW	r2345	2023-07-06 23:03:10	b6cc6e91-397d-4920-8748-85a446a5dcf8
12	fedor_123	prof@jinr.ru	AOD	r7535	2023-07-06 23:04:23	b6cc6e91-397d-4920-8748-85a446a5dcf8
13	val_ry	valery@gmail.com	RAW	r7535	2023-07-07 17:21:21	b6cc6e91-397d-4920-8748-85a446a5dcf8
14	val_ry	valery@gmail.com	RAW	r4345	2023-07-07 17:21:41	b6cc6e91-397d-4920-8748-85a446a5dcf8
15	val_ry	valery@gmail.com	AOD	r2345	2023-07-07 17:25:46	b6cc6e91-397d-4920-8748-85a446a5dcf8
16	val_ry	valery@gmail.com	RAW	r2345	2023-07-07 17:26:32	b6cc6e91-397d-4920-8748-85a446a5dcf8
17	fel_beg	felury	AOD	r4365	2023-07-07 17:28:16	b6cc6e91-397d-4920-8748-85a446a5dcf8
19	aleksand12321	kniflem@gmail.com	RAW	r2345	2023-07-07 23:44:02	b6cc6e91-397d-4920-8748-85a446a5dcf8
20	aleksand12321	kniflem@gmail.com	AOD	r4345	2023-07-07 23:44:16	b6cc6e91-397d-4920-8748-85a446a5dcf8
21	aleksand12321	kniflem@gmail.com	RAW	r2345	2023-07-07 23:47:27	b6cc6e91-397d-4920-8748-85a446a5dcf8
22	aleksand12321qw	dsntsfem@gmail.com	AOD	r2345	2023-07-08 13:06:49	b6cc6e91-397d-4920-8748-85a446a5dcf8
23	aleksand12321qw	dsntsfem@gmail.com	AOD	r2345	2023-07-08 13:09:08	b6cc6e91-397d-4920-8748-85a446a5dcf8
18	wilder	wer@gmail.com	AOD	r2345	2023-07-07 22:45:36	b6cc6e91-397d-4920-8748-85a446a5dcf8
24	davidmake123	davyd@gmail.com	AOD	r4345	2023-07-07 17:28:16	b6cc6e91-397d-4920-8748-85a446a5dcf8
25	johnsmith456	jhone2@gmail.com	RAW	r7535	2023-07-08 17:28:16	b6cc6e91-397d-4920-8748-85a446a5dcf8

Рис. 2. Web-интерфейс

для создания современных динамических пользовательских интерфейсов, а также обеспечивающего эффективное взаимодействие с сервером и манипуляцию данными (рис. 2).

RabbitMQ [6] является брокером сообщений, который позволяет отправлять, получать и маршрутизировать сообщения между компонентами приложений в асинхронном режиме. Celery [7] — это система, которая позволяет выполнять операции в фоновом режиме. Совместное использование RabbitMQ и Celery при создании системы для асинхронной обработки задач способствует улучшению ее производительности. Flask может сосредотачиваться на обработке HTTP-запросов, в то время как обработка задач передается RabbitMQ и Celery для обеспечения модульной и гибкой архитектуры приложения.

Интеграция всех этих технологий позволила создать мощный и гибкий интерфейс для обмена данными, что способствует более эффективной обработке информации в системе EventIndex.

ГЕНЕРАЦИЯ ДАННЫХ

Для тестирования прототипа системы генерируются наборы данных EventIndex. Формат этих наборов — JSON, и он не зависит от формата, в котором будут храниться данные с детектора. Для каждого события генерируются идентификаторы (`run_number` и `event_number`), случайный `olf_result`, а также `fluid_raw` и `dsid_raw`. В дальнейшем на основе этих наборов создаются псевдоданные для файлов AOD.

Эти данные затем записываются в таблицы в базе данных: записи событий — в таблицу «events», а информация о датасетах — в таблицу «datasets». ID датасета служит внешним ключом для таблицы событий.

ОПТИМИЗАЦИЯ СКОРОСТИ ЗАПИСИ ДАННЫХ

Применяемая в первой версии интерфейса процедура записи каждого события отдельной инструкцией PUT показала недостаточную скорость при тестировании даже на сравнительно небольших массивах данных. Например, 1000 событий в среднем записываются 1 мин 55 с. Необходимо система, способная справиться с потоком в десятки тысяч событий в секунду. Для решения данной задачи были исследованы различные методы оптимизации, позволяющие ускорить процесс записи данных в таблицы.

Наряду с использованием «стандартного» драйвера `psycopg2` для взаимодействия с PostgreSQL были протестированы следующие модули: `asynpcpg`, `pg8000`, `PyGreSQL`, `SQLAlchemy`.

Для оптимизации скорости записи данных использовался запрос COPY вместо запроса INSERT. Результаты тестирования представлены на рис. 3.

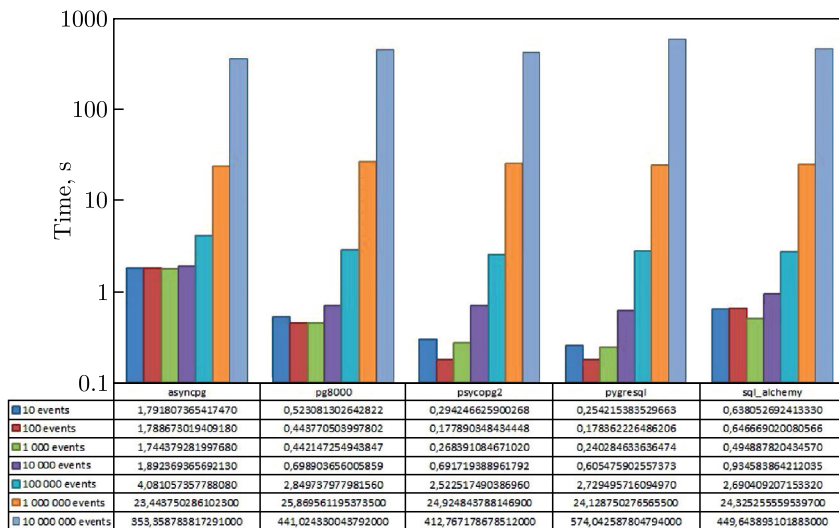


Рис. 3. Время записи наборов данных с использованием различных модулей

Использование COPY ускорило запись данных в таблицу примерно в два раза. Из графика следует, что при работе с малым количеством событий (от 10 до 100 000) эффективным является использование модуля PyGreSQL, однако для большого количества событий (от 1 000 000) следует использовать асинхронный модуль asynpcrg.

В настоящее время идет работа по оптимизации скорости записи данных с помощью изменения параметров PostgreSQL, исследуется влияние размеров блоков на скорость загрузки. Система пытается загрузить все данные сразу, но при высокой нагрузке она адаптируется, используя кластерный подход для эффективной обработки данных. Дополнительно рассматривается возможность параллельной загрузки данных для улучшения производительности.

ЗАКЛЮЧЕНИЕ

В ходе дальнейшего развития проекта EventIndex предполагается выполнение следующих задач.

1. Разработка механизмов авторизации и аутентификации пользователей с использованием технологии единого входа и групповых политик доступа.
2. Разработка API (REST, Python, C++, ...).
3. Оптимизация обработки запросов пользователей с синхронной или асинхронной выдачей результатов в зависимости от объема запрашиваемых данных.

4. Разработка механизмов передачи данных EventIndex, полученных при индексировании файлов, расположенных на удаленных узлах распределенной вычислительной сети.

5. Разработка диспетчера (supervisor) — программного обеспечения для управления, сбора и импорта данных в EventIndex.

6. Разработка системы мониторинга компонентов EventIndex с представлением данных в графическом виде на основе популярных платформ (Grafana и т. д.).

Реализация этих задач будет проводиться параллельно с разработкой других ИС установки.

СПИСОК ЛИТЕРАТУРЫ

1. *Abazov V. et al. (SPD Collab.). Conceptual Design of the Spin Physics Detector.* arXiv:2102.00442 [hep-ex].
2. Official PostgreSQL Website. <https://www.postgresql.org/> (accessed 13.09.2023).
3. *Richardson L., Amundsen M. RESTful Web APIs.* O'Reilly Media, 2013.
4. Flask's Documentation. <https://flask.palletsprojects.com/> (accessed 13.09.2023).
5. Introduction to Angular Concepts. <https://angular.io/guide/architecture> (accessed 13.09.2023).
6. RabbitMQ. <https://www.rabbitmq.com/> (accessed 13.09.2023).
7. Celery — Distributed Task Queue. <https://docs.celeryq.dev/> (accessed 13.09.2023).