МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ «МИФИ»
(НИЯУ МИФИ)
ИНСТИТУТ ЯДЕРНОЙ ФИЗИКИ И ТЕХНОЛОГИЙ
КАФЕДРА №40 «ФИЗИКА ЭЛЕМЕНТАРНЫХ ЧАСТИЦ»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА ДЛЯ УПРАВЛЕНИЯ ПРОЦЕССАМИ ОБРАБОТКИ ПЕРВИЧНЫХ ДАННЫХ ЭКСПЕРИМЕНТА SPD

Студент _____ А. В. Плотников

Научный консультант _____ Д. А. Олейник

Москва 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время практически любой эксперимент в физике высоких энергий предполагает проведения большого количества измерений, порождающего, в результате, гигантский объём данных, требующих различной обработки.

Современные ускорители позволяют осуществлять миллионы взаимодействий пучков в секунду. Многие взаимодействия порождают исследуемые события. Так, ожидаемая частота взаимодействий для эксперимента SPD на коллайдере NICA составит ~13 МГц, при этом частота событий оценивается в 3 МГц. На данный момент детекторные системы позволяют регистрировать подобные события с достаточно высокой точностью, однако это приводит к возникновению большого потока данных, требующих последующей обработки.

Учитывая рост затрат на хранение и последующую обработку данных при увеличении их количества, существует необходимость отсеивать ненужные, в том числе и фоновые, события. Чаще всего в таких случаях применяют аппаратные триггерные системы, но в некоторых ситуациях подобные конструкции невозможны из-за особенностей эксперимента. В этом случае, необходима специальная предварительная обработка данных, с целью сокращения их объема для долговременного хранения.

В данной работе будет рассмотрена подобная система для разрабатываемого эксперимента SPD на коллайдере NICA (Дубна, Россия) [1]. А также будет описан этап разработки одной из подсистем – системы управления процессами обработки.

Цель и задачи работы

Цель работы: разработка системы управления процессами обработки (workflow management system - WfMS) для декларирования формализованного описания процессов обработки, осуществления управления и контроля за выполнением параллельных процессов обработки.

Задачи:

1. ознакомиться с проектом системы Workflow management system (WfMS): определить основные сервисы WfMS, выявить их основной функционал и принцип взаимодействия с другими системами SPD Online Filter;
2. изучить предлагаемый для использования в реализации стек программных платформ. При необходимости дополнить/переработать;
3. разработать API для работы с базой данных;
4. разработать сервисы WfMS с использованием выбранного стека технологий:
    1. создать сервис для взаимодействия с оператором обработки:
        - авторизация пользователей;
        - обработка CWL-шаблонов;
        - вывод информации о заданиях и шаблонах;
        - работа со статусами шаблонов.
    2. реализовать сервис генерации заданий:
        - получение датасетов из DMS;
        - создание заданий по шаблонам для последовательностей.
    3. разработать сервис для опроса DMS:
        - опрос DMS о готовности входных датасетов для заданий;
        - создание выходных датасетов и датасетов логов в DMS;
        - отправление заданий на обработку.

1. SPD Online Filter

1.1 Принцип работы SPD Online Filter

SPD Online Filter — это высокопроизводительная вычислительная система для высокопропускной первичной обработки данных эксперимента SPD [2].

Эта вычислительная система должна реализовывать первичную обработку экспериментальных данных, в соответствии с предварительно заданными процессами обработки, например, отфильтровать события, оставляя только интересные с точки зрения эксперимента; согласовывать выходные данные, объединять события в файлы и файлы в наборы данных для будущей обработки.

Так как SPD Online Filter предназначен для обработки большого потока данных. Данные объединяются в наборы файлов в зависимости от стадии их обработки. Каждый файл в наборе может быть обработан независимо, однако наборы обрабатываются в некоторой заданной последовательности. Каждый файл можно обработать за некоторое разумное время на ограниченном вычислительном ресурсе (вычислительном узле).

Разбиение данных на более мелкие части позволяет управлять объемом обработки и минимизировать последствия при возникновении ошибок при обработке больших объемов данных. Работа с отдельными частями позволяет изолировать ошибки, которые могут возникнуть в процессе обработки, и снизить их воздействие на всю систему.

Такой подход также обеспечивает масштабируемость системы: добавление дополнительных вычислительных ресурсов позволяет обрабатывать еще большие объемы данных без значительного изменения общей архитектуры системы. Это особенно полезно для систем, работающих с высокочастотными данными или при необходимости быстрой обработки информации в режиме реального времени.

По результатам анализа первичных требований к системе были выделены основные компоненты:

1. Workflow management system

2. Data management system

3. Workload management system & pilot

Архитектура системы SPD Online Filter приведена на рис. 1.

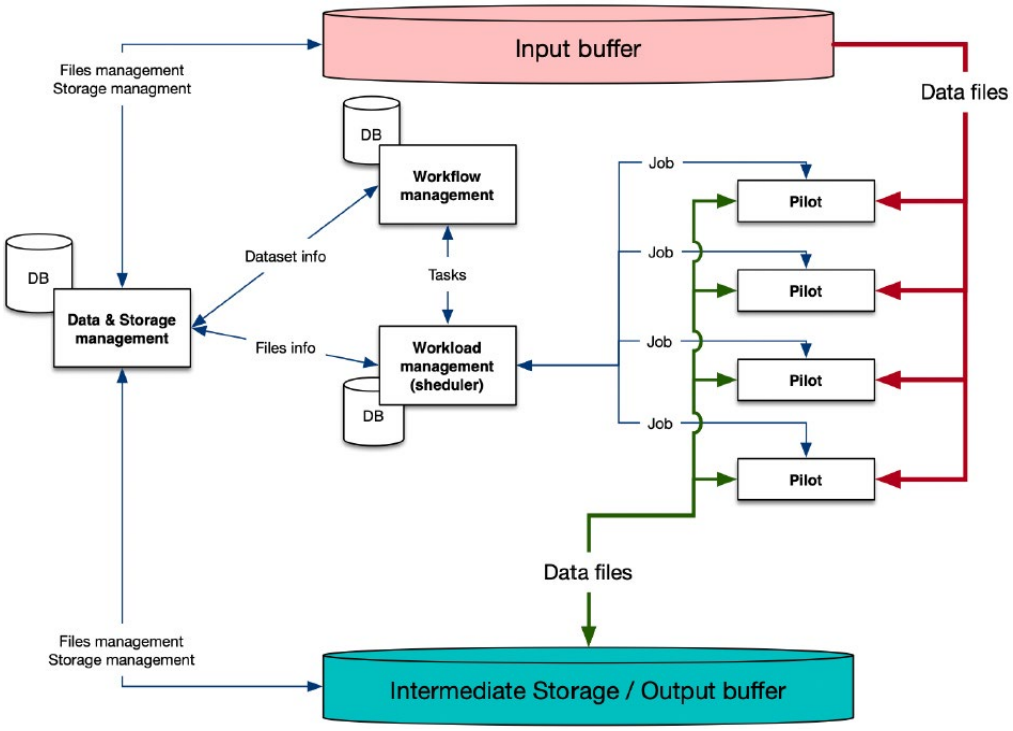


Рис. 1 Архитектура SPD Online Filter [3]

1.2 Входные данные

Входные данные, приходят из системы сбора данных (DAQ) во входной буфер.

● Период набора – ассоциирован с физической задачей. Состоит из набора ранов. Имеет дату начала, дату окончания.

● Ран – интервал, когда условия эксперимента неизменны (калибровки, например). Измеряется часами. Состоит из фреймов.

● Фрейм – нумерованный блок данных с детектора. Может содержать информацию о множестве событий. Продолжительность – секунды.

● Датасет – логическое объединение файлов в наборы для обработки. Датасет является единицей обработки для одного задания.

● Файл – количество данных для обработки одной задачей.

2. Система управления процессами обработки (WfMS)

После попадания в SPD Online Filter, данные регистрируются в системе управления данными (Data Management System - DMS). Затем данные подготавливаются к последующей обработке при помощи системы управления процессами обработки (Workflow Management System - WfMS).

WfMS принимает эти зарегистрированные данные и сопоставляет их с определенным шаблоном цепочки обработки, который предоставлен оператором обработки данных. Каждый этап этой цепочки обработки генерирует соответствующие задания для выполнения. Эти задания затем направляются в систему управления нагрузкой (Workload Management System - WMS), где они распределяются на вычислительные ресурсы для выполнения. После этого WfMS контролирует процесс выполнения задания путем периодического опроса WMS об их статусах и принимает решение об изменении приоритетов заданий или об их отмене.

Этот процесс позволяет эффективно организовать и контролировать обработку данных по заданной цепочке этапов. Задания каждого этапа обработки создаются автоматически и передаются в систему управления нагрузкой для выполнения на доступных вычислительных ресурсах. Такой подход обеспечивает систему планирования и контроля рабочих процессов, что улучшает эффективность обработки данных и ускоряет процесс анализа.

Система управления процессами обработки (WfMS) имеет ряд функциональных требований, чтобы эффективно организовать и контролировать последовательности обработки данных:

1. Организация последовательностей обработки данных:

● Последовательность: состоит из логически связанных шагов обработки данных, где результат одного шага является входными данными для следующего.

● Шаг обработки: представляет собой действие, которое выполняется над набором данных. Эти шаги могут быть типа "map" (однотипная обработка

для каждого элемента данных) или "merge" (объединение гомогенных данных).

● Описание шаблона: выражается с помощью CWL (Common Workflow Language).

2. Формирование запроса на обработку данных:

● Для каждого шага обработки определяются необходимые параметры для формирования задания.

3. Выполнение запроса на обработку данных:

● Создание заданий для каждого шага обработки данных на основе сформированных параметров.

● Отправка созданных заданий в систему управления нагрузкой для выполнения.

● Осуществление контроля выполнения обработки данных путем опроса системы управления нагрузкой.

Эти функциональные требования позволят системе WfMS эффективно управлять процессами обработки, обеспечивая оптимизацию обработки данных, контроль ошибок и достижение заданных целей обработки.

Пример возможной обработки данных:

1. Декодирование данных;

2. Выявление событий и реструктуризация данных;

4. Фильтрация выявленных событий;

5. Верификация данных;

6. Объединение файлов (уменьшение количества файлов, увеличение размера файлов);

Такой набор последовательных шагов обработки будет называться цепочкой обработки. Каждый из шагов в такой цепочке может быть определен шаблоном.

Во время работы системы предполагается создание промежуточных данных, которые будут удаляться после получения выходного набора данных.

В результате проектирования были определены основные сервисы системы WfMS:

● Сервис для взаимодействия с оператором обработки данных:

1) авторизация пользователей;

2) обработка CWL-шаблонов;

3) вывод информации о заданиях и шаблонах;

4) работа со статусами шаблонов;

● Сервис для опроса DMS:

1) получение информации о готовности входных датасетов;

2) формирование выходных датасетов и датасетов логов;

3) отправление заданий на обработку;

● Сервис генерации заданий:

1) определение последовательностей обработки данных;

2) создание заданий по шаблонам для последовательностей.

● Сервис для работы с WMS:

1) периодический опрос для отслеживания статуса;

2) изменение приоритета заданий;

3) отмена заданий.

Архитектура WfMS приведена на рис. 2.

В данной системе применяется микросервисная архитектура. В рамках этой архитектуры приложение разбивается на отдельные сервисы, которые могут быть развернуты и масштабированы независимо друг от друга. Сервисы взаимодействуют между собой через API-интерфейсы, что позволяет каждому из них функционировать автономно. В отличие от монолитной архитектуры, микросервисы позволяют быстрее внедрять новые функции и вносить изменения, не требуя переработки большого объема существующего кода.

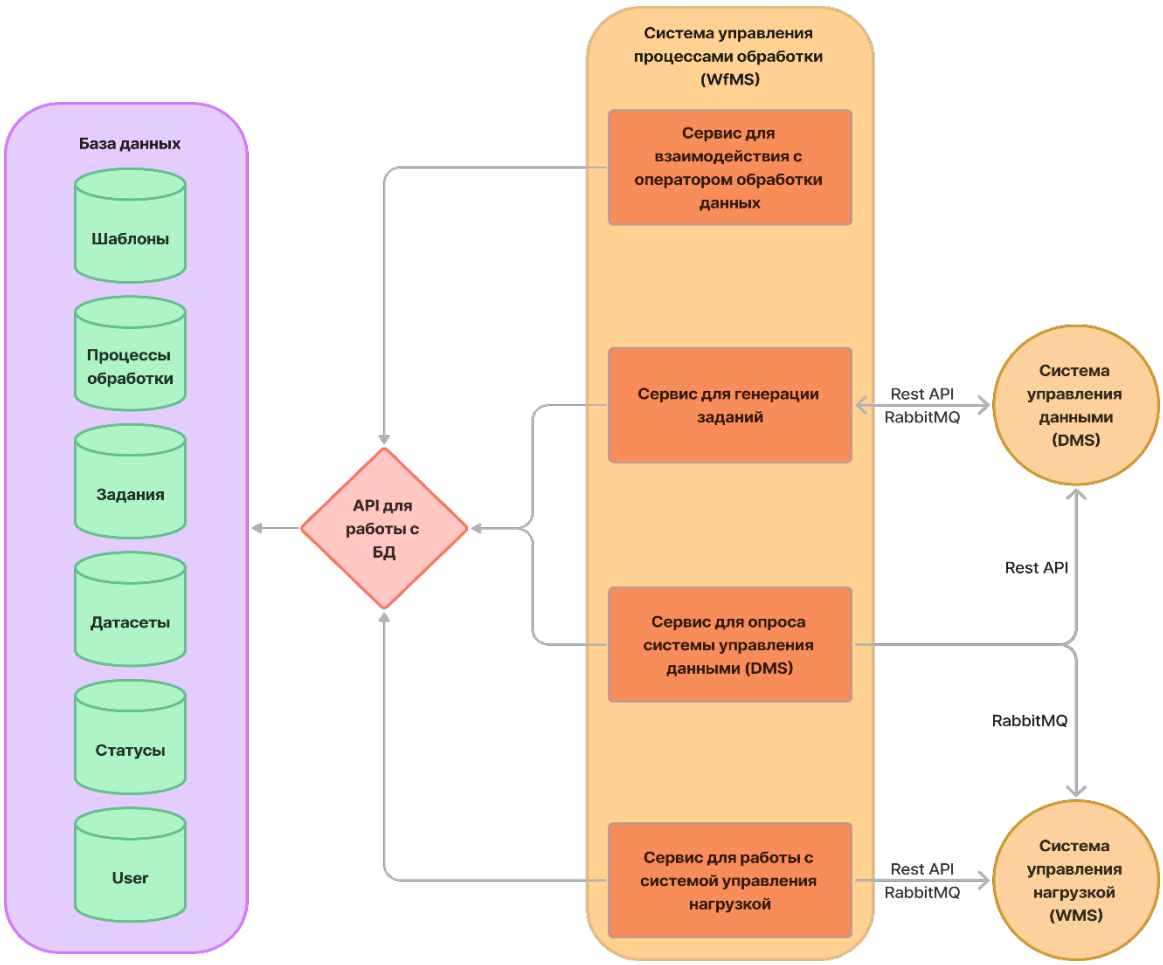


Рис. 2 Архитектура Workflow Management System

Основные преимущества микросервисной архитектуры включают:

● Повышенную доступность и отказоустойчивость. Сбой одного из сервисов не приводит к отказу всей системы.

● Масштабируемость. При достижении предельной нагрузки на микросервис можно развернуть дополнительные экземпляры этой службы и распределить нагрузку, поддерживая таким образом работоспособность большого количества экземпляров.

3. База данных и API к ней

Для хранения данных в системе WfMS используется объектно-реляционная база данных. В качестве СУБД используется PostgreSQL, развернутая на отдельной виртуальной машине.

Структура базы данных представлена на рис. 3.

Рис. 3 Структура базы данных

Для создания и работы с базой данных используется библиотека SQLAlchemy (ORM). Миграции проводятся с помощью библиотеки Alembic. Для достижения высокой скорости взаимодействия и асинхронного взаимодействия используются асинхронные сессии и asyncpg.

Взаимодействие каждого из микросервисов с базой данных представлено на рис. 4.

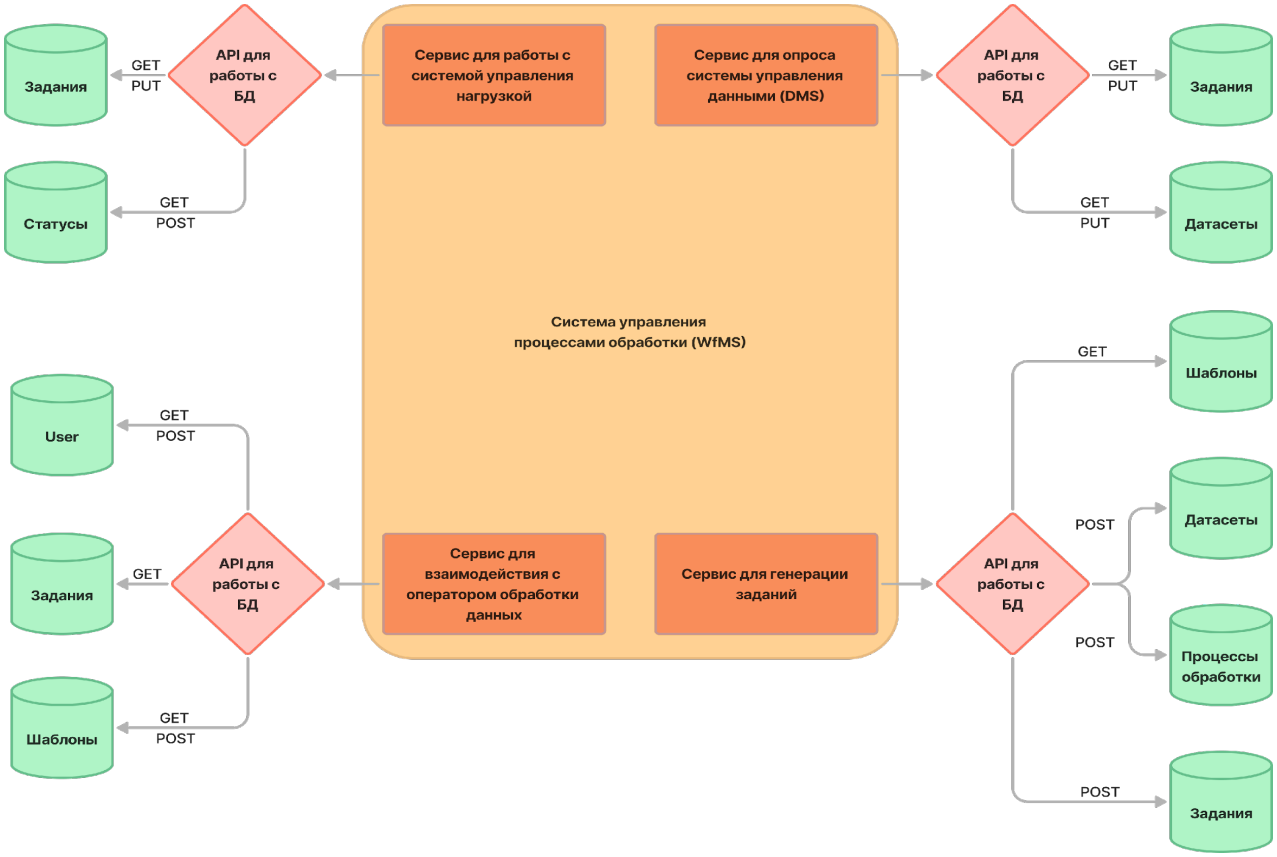


Рис. 4 Схема взаимодействия микросервисов с БД

Всё взаимодействие с базой данных осуществляется через REST API. Для этого используется фреймворк FastAPI [4]. Вся документация и проверка API производится во встроенном инструменте FastAPI — Swagger (рис. 5).

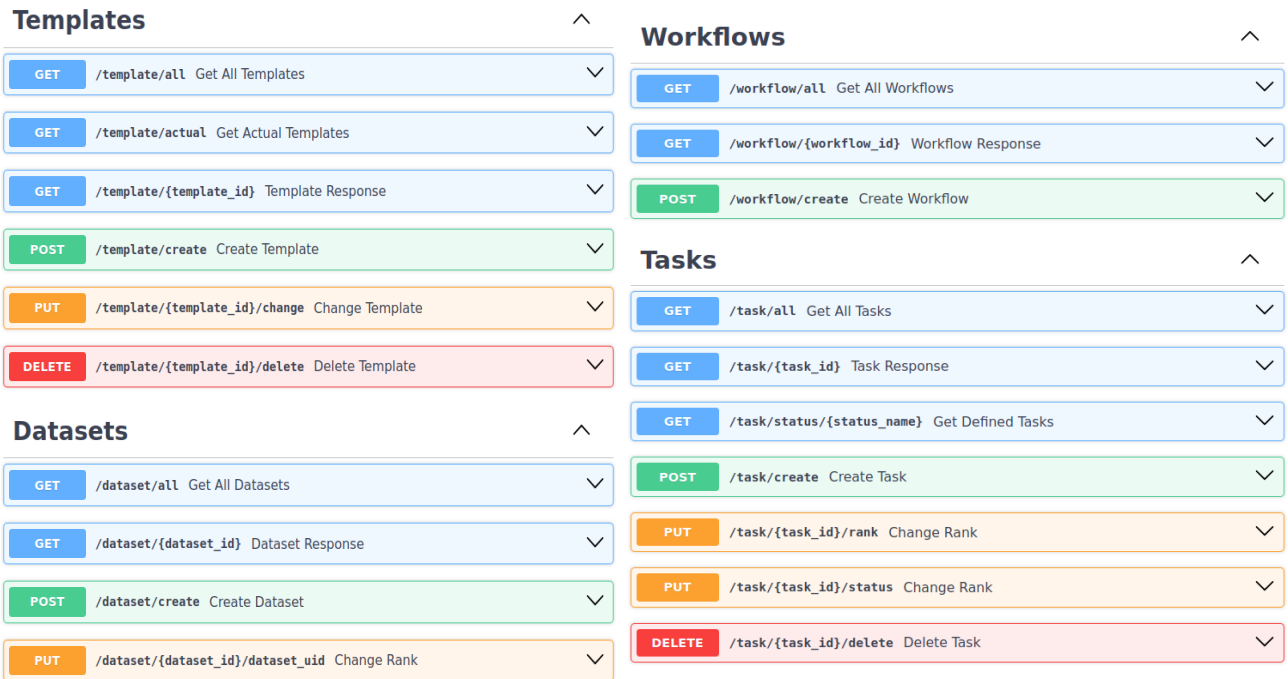


Рис. 5 DB API Swagger

4. Сервис для взаимодействия с оператором обработки данных

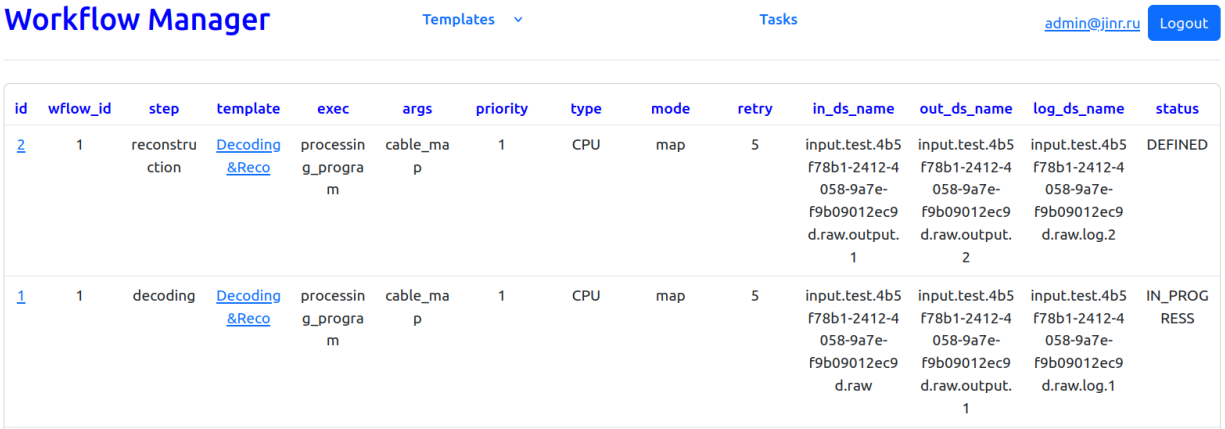
Основной функционал сервиса:

1) Регистрация и авторизация пользователей с разными правами;
2) Вывод CWL-шаблонов;
3) Вывод заданий;
4) Создание CWL-шаблонов суперпользователями;
5) Предварительная валидация и сохранение CWL-шаблонов;
6) Изменение статусов шаблонов суперпользователями;

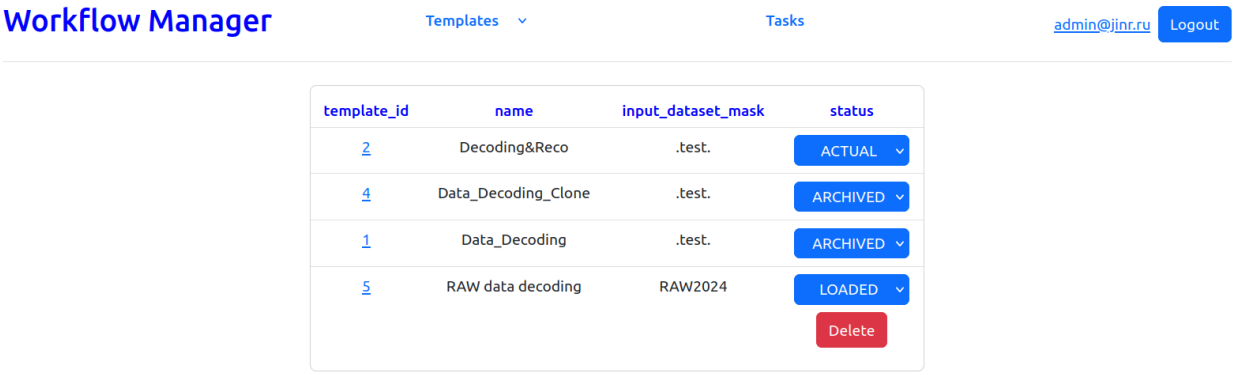
Сервис для взаимодействия с оператором обработки данных обеспечен пользовательским интерфейсом. Backend сервиса написан на FastAPI, для frontend использовался bootstrap (HTML + CSS + JS), а объединяет их воедино шаблонизатор Jinja2.

Для начала использования сервиса необходимо пройти процесс регистрации, а затем авторизацию и аутентификацию. Данная часть сервиса была реализована с помощью библиотеки FastAPI Users на сохранении JWT-токенов в cookie браузера. В дальнейшем было решено отказаться от внутренней регистрации и аутентификации пользователей, в угоду интеграции с новой системой SPD-IAM, объединяющей все приложения коллаборации SPD.

Суперпользователям, помимо просмотров шаблонов и заданий (рис. 6), так же доступно создание шаблонов, в том числе и с помощью клонирования уже существующих шаблонов, и смена их статусов: LOADED — загруженные шаблоны, которые можно удалить; ACTUAL — шаблоны, используемые при создании заданий; ARCHIVED — неактивные шаблоны. Вернуть шаблон в статус LOADED невозможно.



а)



б)

Рис. 6 Web-страницы Workflow Manager

а) Страница с заданиями;

б) Страница с шаблонами

Отдельно можно посмотреть и сам шаблон, суперпользователю же доступна возможность его клонирования, для создания на его основе нового экземпляра. Показано на рис. 7.

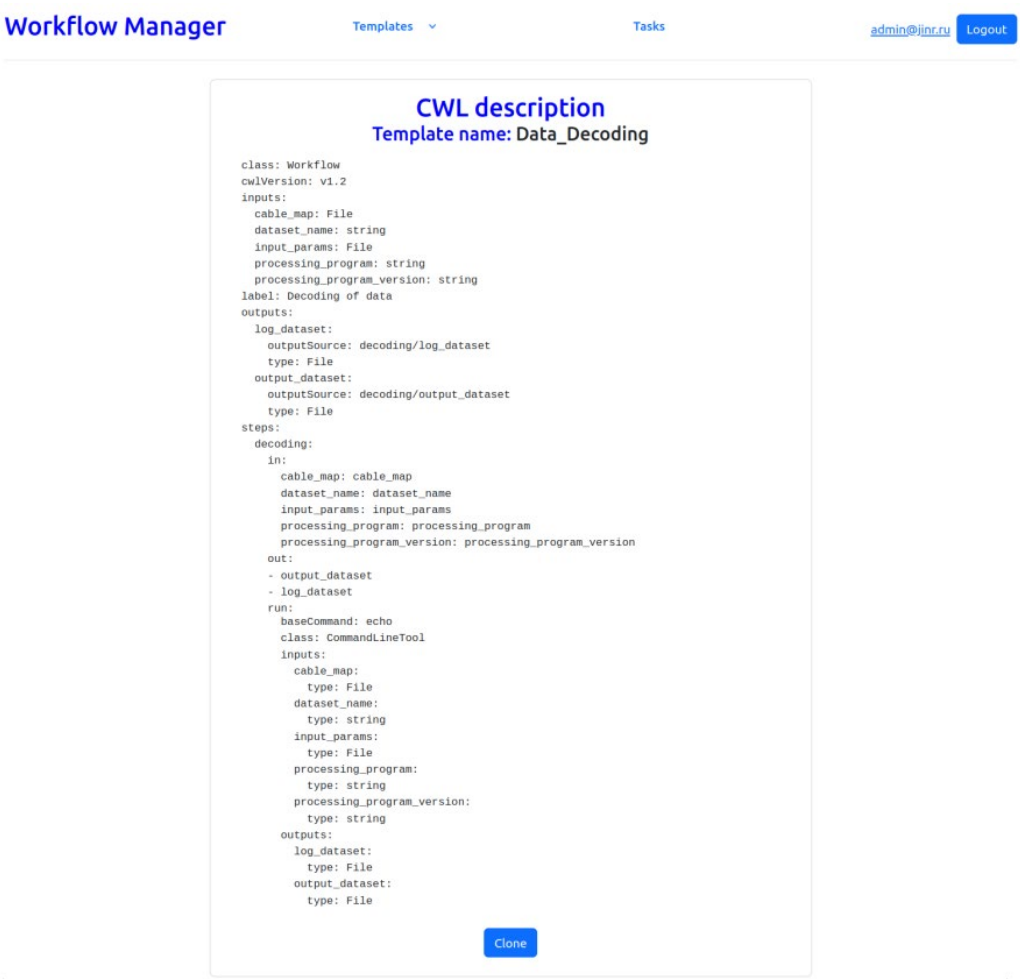


Рис. 7 Страница с отдельным шаблоном

Шаблон задается в формате Common Workflow Language (CWL). Перед сохранением в базу данных шаблон проходит процесс валидации с помощью инструментов cwltools. Страница создания шаблона представлена на рис. 8.

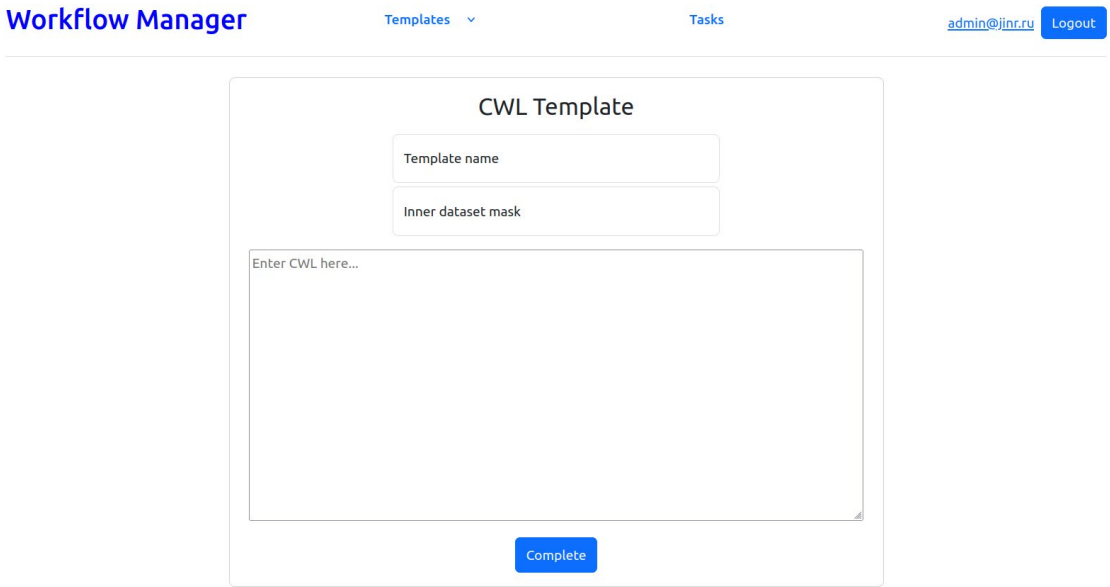


Рис. 8 Страница создания CWL-шаблона

5. Сервис для генерации заданий

Сервис для генерации заданий обладает следующим функционалом:

1) Получение зарегистрированных датасетов от DMS из брокера сообщений RabbitMQ [5];
2) Сопоставление датасета по маске имени с нужным шаблоном;
3) Регистрация входного датасета в системе;
4) Создание процесса обработки по шаблону;
5) Создание выходного датасета и датасета логов в системе;
6) Создание заданий.

После того, как датасет был сформирован, DMS отправляет информацию о нем (name, dataset_uid) в брокер сообщений RabbitMQ. Информация о датасетах извлекается по порядку из очереди и сопоставляется по маске имени с шаблонами, написанными оператором обработки в соответствующем сервисе. Используются только те шаблоны, которые находятся в статусе ACTUAL.

Входные датасеты регистрируются в системе, после чего создается процесс обработки по найденному шаблону. Генерируются записи о выходных датасетах и датасетах логов в базе данных для каждого из этапов цепочки обработки с целью дальнейшего их создания в DMS перед отправкой конкретного задания на обработку. Данная реализация позволяет не создавать лишних датасетов в системе в случаях, когда задание отменено.

Далее формируется само задание. Ему присваивается статус DEFINED. Данное задание отличается от того, что будет отправлено на дальнейшую обработку из-за особенностей хранения в базе данных. Таким образом, в данном сервисе формируется «шаблон» задания, который затем будет заполняться перед отправлением на обработку в WMS.

Все этапы генерации заданий осуществляются в асинхронном режиме.

Весь процесс наглядно можно видеть на рис. 9.

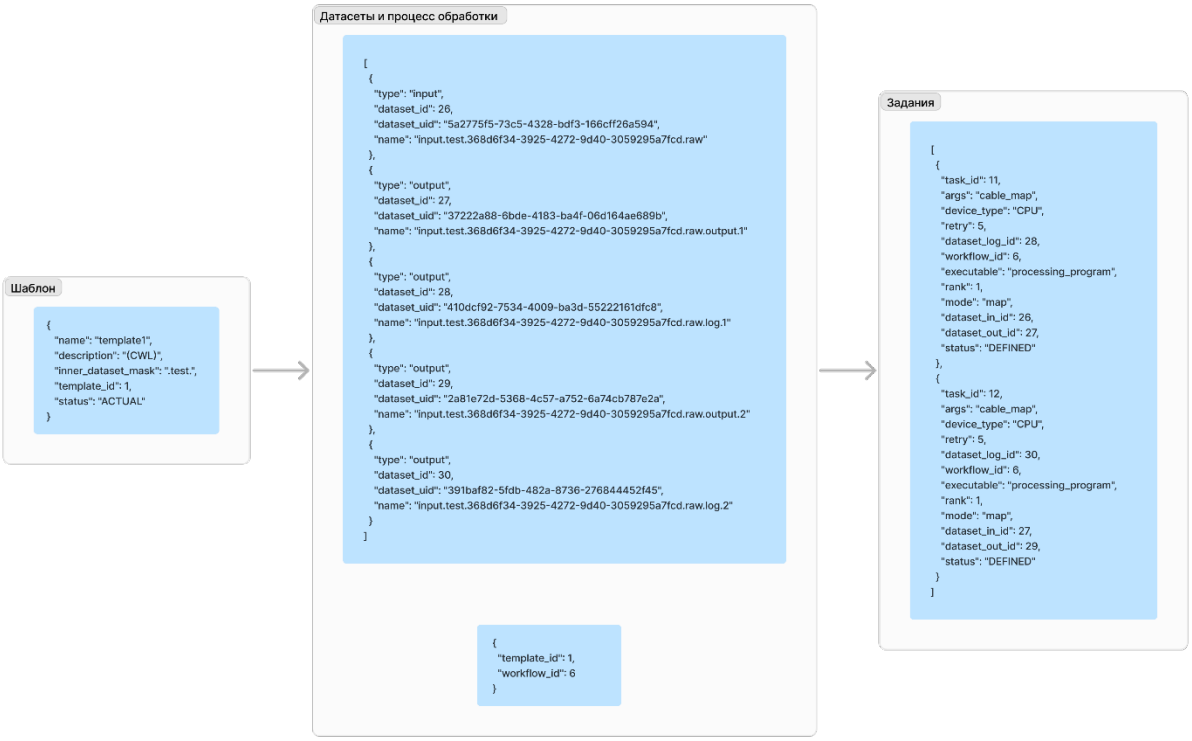


Рис. 9 Генерация задания по шаблону

6. Взаимодействие с Data Management System

Далее представлено подробное описание процесса взаимодействия с Data Management System (DMS). Основной задачей данного взаимодействия является получение новых данных, регистрация промежуточных данных и выгрузка выходных данных. Кроме основных функций, взаимодействие должно быть реализовано с учетом обеспечения высокой эффективности и максимальной скорости, исключения узких мест, а также минимизации возможных ошибок.

Взаимодействие с DMS включает следующие этапы:

1. Получение готового входного набора данных. После поступления данных в SPD Online Filter, DMS регистрирует их в своей системе. После этого WfMS может получить информацию об этом датасете через REST API.

2. Регистрация выходного набора данных. Перед любым этапом обработки данных WfMS должен зарегистрировать выходной набор данных, чтобы знать, где и как искать полученные данные. Для этого также используется REST API. Промежуточные и выходные данные сопровождаются логами, которые регистрируются аналогичным образом.

3. Обработка промежуточных данных. В ходе обработки данных появляются промежуточные результаты. Если в цепочке обработки имеется n этапов, то при сохранении всех промежуточных данных после каждого этапа объем используемой памяти значительно увеличивается. Поэтому принято решение сохранять только один набор входных данных и один набор промежуточных данных. Это значит, что после завершения этапа обработки и появления нового датасета, входные данные для этого этапа удаляются. Входные данные для первого этапа остаются до завершения всей цепочки обработки, чтобы можно было восстановить данные в случае сбоя системы. В данном взаимодействии используется асинхронный метод, позволяющий не останавливать работу WfMS и не ждать удаления данных. Логи для промежуточных и выходных данных также должны удаляться аналогичным способом.

6.1 Сервис для опроса DMS

После того, как задания были сгенерированы в соответствующем сервисе они получают статус DEFINED. Данный сервис извлекает такие задания из базы данных и итерируется по ним. С помощью REST API в DMS (dsm-manager) отправляется запрос на получение статуса входного датасета для конкретного задания. Данный датасет должен быть готов для обработки (находиться в статусе «CLOSED»).

Для входных датасетов в статусе «CLOSED» создаются выходные датасеты и датасеты логов в DMS через REST API (dsm-manager) при этом вся информация о них передаётся из внутренней базы данных WfMS, где они были созданы на этапе генерации заданий.

Задание формируется в необходимом для дальнейшей обработки виде и отправляется в очередь брокера сообщений RabbitMQ в формате JSON, после чего оно может быть извлечено WMS. Данная очередь создается в момент инициализации сервиса. Сохранность сообщений в очереди и их доставку до получателя обеспечивает RabbitMQ.

После того, как сообщение было отправлено, задание получает статус «IN_PROGRESS», что обеспечит возможность их отслеживания и управления ими в дальнейшем.

Принцип отправления заданий на обработку в WMS показан на рис. 10.

Рис. 10 Отправление заданий на обработку

7. Логирование, контейнеризация и оркестрация

Для получения необходимой информации о состоянии системы был разработан logger, записывающий необходимые для дальнейшего анализа данные о работе каждого сервиса. В частности, записываются название сервиса, уровень сообщения (INFO, WARNING, ERROR, CRITICAL), дата и время получения сообщения и описание сообщения.

Одно из сообщений logger можно видеть на рис. 11.

```
task_generator 2024-09-24 20:59:44,380 INFO Logger started
task_generator 2024-09-24 20:59:44,665 INFO Templates successfully initialised.
```

Рис. 11 Сообщение logger

Для быстрого развертывания приложений на любой системе и налаживания их совместного взаимодействия используются методы контейнеризации и оркестрации с помощью Docker и Docker-Compose [6]. Представление реализованных сервисов в docker compose изображено на рис. 12.

docker-compose

```
services:
  db_api:
    ...
  task_generator:
    ...
  task_manager:
    ...
  template_manager
    ...

networks:
  wfms:
```

Рис. 12 Сервисы WfMS в docker-compose

8. Результаты и дальнейшие планы

Результаты:

1) разработан API для работы с базой данных, открывающий доступ для сервисов WfMS к определенному набору функций;

2) создан сервис для работы с оператором обработки данных, позволяющий просматривать задания и шаблоны, а также предоставляющий возможность суперпользователям генерировать шаблоны и управлять их статусами;

3) разработан сервис для генерации заданий, сопоставляющий датасеты с шаблонами по маске имени и создающий цепочки обработки и задания;

4) создан сервис для опроса DMS, опрашивающий его о готовности входного датасета для каждого готового задания, создающий выходные датасеты для таких заданий и отправляющий задания на обработку в WMS;

5) произведена контейнеризация всех приложений и настроена их оркестрация с помощью docker-compose;

6) внедрен logger, сохраняющий информацию об актуальном состоянии системы в каждый момент времени.

Дальнейшие планы:

1) Интеграция с SPD-IAM (identity and access management service);
2) Добавление возможности загрузки шаблонов из файла;
3) Разработка сервиса для взаимодействия с WMS;
4) Переход к полной асинхронности;
5) Тестирование системы.

ЗАКЛЮЧЕНИЕ

Современные эксперименты в области физики высоких энергий, сталкиваются с необходимостью эффективной обработки огромных объемов данных, генерируемых детекторными системами. Эксперимент SPD на коллайдере NICA предъявляет высокие требования к системе фильтрации данных в реальном времени, учитывая интенсивность событий и необходимость оперативной обработки информации. В этом контексте проектирование системы управления процессами обработки данных становится не просто задачей, а ключевым фактором для достижения успешных результатов в исследованиях физики высоких энергий. Эффективное управление процессами обработки данных в вычислительных комплексах имеет ключевое значение для обеспечения производительности и надежности системы.

В ходе работы были рассмотрены основные аспекты функционирования системы SPD Online Filter эксперимента SPD и её подсистемы - системы управления процессами обработки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. International spin physics collaboration at the collider NICA [Электронный ресурс] // http://spd.jinr.ru/

2. Oleynik D. Data processing in HEP experiments. [Электронный ресурс] // https://lit.jinr.ru/sites/lit.jinr.ru/files/pdf/HEPNICA_computing_2022_OleynikD.pdf

3. Abazov V. M. [и др.] Technical Design Report of the Spin Physics Detector at NICA. 2024.arXiv:2404.08317v1 [hep-ex] https://doi.org/10.48550/arXiv.2404.08317

4. FastAPI [Электронный ресурс] // https://fastapi.tiangolo.com/

5. RabbitMQ [Электронный ресурс] // https://www.rabbitmq.com/

6. Docker-compose [Электронный ресурс] // https://docs.docker.com/compose/