

ТРЕБОВАНИЯ К АРХИТЕКТУРЕ СИСТЕМЫ УПРАВЛЕНИЯ НАУЧНЫМИ ДАННЫМИ И ВЫБОР РЕШЕНИЙ ДЛЯ РЕАЛИЗАЦИИ ЕЕ КОМПОНЕНТОВ

К. В. Гарев^[0009-0001-1961-3195]

НИЦ «Курчатовский институт» – НИИСИ, г. Москва, Россия

kv@garev.ru

Аннотация

Рассмотрена задача выбора технических решений для компонентов системы управления научными данными. Последние разделены на три категории: научные датасеты, научно-техническая и наукометрическая информации – различающиеся по структуре, объему, характеру обновления, профилю доступа и требованиям к метаданным, что усложняет эффективное управление ими в рамках единого универсального хранилища. На основе типологизации данных и сценариев использования для ключевых ролей (исследователь, аналитик, инженер данных, администратор) сформированы требования к компонентам системы. Был проведен сравнительный анализ альтернативных решений по каждому компоненту и предложена модульная архитектура, разграничивающая исходные данные и их производные представления. Определены согласованный набор технологий, совместимый с локальным развертыванием, а также сквозные механизмы идентификации, авторизации и управления ресурсами.

Ключевые слова: управление научными данными, научные датасеты, научно-техническая информация, наукометрическая информация, программная архитектура, требования к системе, распределенное хранение, семантический поиск.

ВВЕДЕНИЕ

Современная научно-технологическая деятельность генерирует разнородные информационные объекты, которые необходимо собирать, хранить, обрабатывать и представлять пользователям. Научные данные трактуются как широкий класс информации, формируемой в ходе исследований. В рамках настоящей

работы они подразделены на три категории: научные датасеты, научно-техническая информация (НТИ) и наукометрическая информация (НМИ). Каждая категория отличается форматами и сценариями применения [1], а также требованиями к метаданным, идентификации и условиям повторного использования [2]. НТИ объединяет отчеты, патенты, технические описания и требует средств текстового поиска и анализа; НМИ включает показатели публикационной активности и цитирования, необходимые для оценки результативности. Управление научными данными целесообразно рассматривать через единый жизненный цикл, однако на уровне архитектуры этот цикл реализуется различными технологическими средствами.

В российском контексте проблема состоит не только в отсутствии репозитория научных датасетов, но и в недостатке комплексных решений для всей совокупности научных данных. Существующие системы охватывают лишь отдельные аспекты и не имеют широкого междисциплинарного применения. Это обуславливает потребность в интегрированном решении, объединяющем работу с научными данными в едином информационном пространстве, что согласуется с принципами открытой науки [3]. Интеграция дает эффект сквозной связности. Возможность сопоставить датасеты с публикациями, патентами и показателями публикационной активности упрощает поиск и анализ данных и повышает точность мониторинга. Ранее сформулированные принципы проектирования системы обмена научными датасетами выявили ключевые недостатки экосистемы: отсутствие единых регламентов описания, разрозненность инфраструктуры, недостаточную межсистемную совместимость и сложность обеспечения воспроизводимости [4].

Цель настоящей работы – на основании исследования сценариев использования научных данных выработать требования к компонентам системы, провести сравнительный анализ альтернативных решений, предложить архитектуру и набор технологий, обеспечивающих масштабирование, контейнеризацию вычислений для исследователей, ключевые каналы доставки данных и инструментальную поддержку анализа с привлечением высокопроизводительных вычислительных систем [5].

1. ТИПЫ НАУЧНЫХ ДАННЫХ И ИХ ХАРАКТЕРИСТИКИ

Исходной предпосылкой проектирования системы является то, что научные данные не образуют однородного класса. Данные, как правило, различаются по структуре, объему и динамике формирования, требованиям к воспроизводимости и метаданным, а также по правовому режиму доступа и сценариям использования. Все эти различия трансформируются в архитектурные требования к хранению и обработке данных.

Приведенная ниже типологизация является необходимым шагом, связывающим свойства данных с требованиями к компонентам системы.

Научные датасеты – это структурированные и полуструктурированные массивы результатов экспериментов, наблюдений и моделирования. Характеризуются большим объемом и гетерогенностью, требуют стандартизации форматов, присвоения устойчивых идентификаторов (DOI) и долгосрочного распределенного хранения для обеспечения доступности и воспроизводимости экспериментов [2, 6]; хранение крупных табличных массивов требует специализированных методов представления и сжатия [7].

Научно-техническая информация – это корпус неструктурированных текстовых данных, например отчеты НИР/ОКР, статьи, патенты, документация. Их обработка требует методов автоматического извлечения сущностей и связей из научных текстов [8], а хранение – поддержки поиска по смысловой близости наряду с полнотекстовым поиском [9].

Наукометрическая информация – это структурированные данные о научной деятельности персоны или организационной сущности: связи, цитирования, индексы, патентная активность. Для поддержки управленческих решений требуется хранить эти данные в структурированном виде для последующего агрегирования показателей и визуализации.

Различия между категориями проявляются на уровне инженерных требований и зависят от степени структурированности, объема и темпа обновления, профиля доступа, требований к метаданным и контролю качества. Следовательно, система должна проектироваться как совокупность специализированных подсистем хранения и обработки, объединенных общими механизмами идентификации, интероперабельности и управления доступом. Сопоставление

категорий научных данных позволяет определить требования к компонентам и обосновать выбор технологий для их реализации.

В табл. 1 представлены различия типов данных между собой в разрезе их характеристик, что влияет на функциональные требования к системе.

Табл. 1. Сравнительные характеристики категорий научных данных.

Характеристика	Научные датасеты	Научно-техническая информация (НТИ)	Наукометрическая информация (НМИ)
Объект управления	Набор данных исследования	Документ или корпус документов (статья, отчет, патент) и их версии	Запись о научной активности и производные показатели
Степень структурированности	Структурированные и полуструктурированные массивы	Преимущественно неструктурированный полный текст и структурированные метаданные	Структурированные таблицы, отношения и графы взаимосвязей
Типичные масштабы объема	От МБ до ТБ и выше; возможны потоковые поступления	От КБ до десятков МБ на документ; до миллионов документов	От десятков МБ до ТБ на совокупность записей и связей
Темп обновления	Пакетный или потоковый	Нерегулярный (по факту публикации)	Регулярный (пересчет метрик)
Идентификация и цитирование	Устойчивые идентификаторы (DOI), цитирование набора данных по схеме DataCite	DOI и иные устойчивые идентификаторы документов	Идентификаторы сущностей как ключи связывания
Требования к метаданным	Машиночитаемые, пригодные для поиска и повторного использования	Метаданные и полный текст; классификация и рубрикация	Метаданные сущностей и правила расчета показателей

Доминирующий профиль доступа	Преимущественно чтение и работа с большими фрагментами; вычисления рядом с данными	Поиск: частые запросы к индексу, чтение фрагментов и документов	Агрегации и отчетность: группировки, срезы, детализация
Требования к хранению	Распределенное хранение, репликация, контроль целостности, версионирование	Полные тексты и структурированный слой извлечений	Реляционное аналитическое хранилище
Требования к поиску	По метаданным и по структуре и подмножествам	Полнотекстовый и семантический поиск	По атрибутам и связям, фильтрация и иерархии
Требования к вычислениям	Контейнеризованные вычисления рядом с данными; воспроизводимые окружения	Конвейеры обработки естественного языка (извлечение, классификация)	Конвейеры агрегации и нормализации, пересчет метрик
Контроль качества	Валидация формата и целостности, контроль версий, репликация	Дедупликация, контроль версий, качество извлечений	Валидация источников, воспроизводимость расчетов

2. ПОЛЬЗОВАТЕЛЬСКИЕ РОЛИ И ТРЕБОВАНИЯ

Система обслуживает несколько категорий пользователей, для каждой из которых определены типовые задачи, определяющие функциональные и технологические требования. В табл. 2 представлены ключевые роли и характерные сценарии взаимодействия пользователей с системой.

Табл. 2. Пользовательские роли и требования к системе управления научными данными.

Роль	Сценарии взаимодействия	Функциональные требования
Исследователь	Поиск, получение и публикация научных датасетов; вычисления над данными; доступ к связанной НТИ	Поиск датасетов в каталоге по метаданным и работа с их карточками; скачивание и программный доступ к данным; запуск изолированных вычислительных сессий и расчеты «рядом с данными»; сохранение и воспроизведение результатов; поиск и доступ к связанным документам НТИ, в том числе семантический
Аналитик	Анализ агрегированных наукометрических данных и содержимого коллекций документов; выявление трендов	SQL-доступ к витринам НМИ, построение визуальных представлений данных и их анализ; выгрузка данных в среды анализа; полнотекстовый и семантический поиск по НТИ; получение структурированных извлечений и сравнение тематик; формирование подборок документов со ссылками на источники
Инженер данных	Подключение и эксплуатация контуров поступления данных; автоматизированное обновление витрин и индексов	Интеграция внешних источников (API, файлы, веб-ресурсы) и инкрементальная загрузка; нормализация и контроль качества данных; сопровождение оркестрации ETL/ELT-пайплайнов; формирование витрин НМИ и структурированных извлечений НТИ, индексация поискового контура; обеспечение воспроизводимости и наблюдаемости процессов
Администратор	Поддержка инфраструктуры и данных; управление доступом, безопасностью и устойчивостью системы	Управление учетными записями, ролями и политиками доступа, аудит действий; конфигурирование федеративной аутентификации; мониторинг сервисов и управление ресурсами и квотами; резервное копирование и восстановление; управление жизненным циклом данных и обновлениями компонентов

Исследователь работает с датасетами и связанными материалами, ведет поиск в каталоге по метаданным, просматривает карточки датасетов, описание датасета, версия, состав файлов, условия доступа, связанные публикации и документы НТИ. Исследователь получает доступ к данным согласно политикам разграничения прав и загружает данные целиком или выборочно, в том числе с привлечением программных средств для их обработки. Ему необходимы персонализированные вычислительные среды с минимальной инфраструктурной задержкой доступа к данным («рядом с данными») и возможностью воспроизвести вычисления в дальнейшем, а также необходимо осуществлять поиск релевантной НТИ, включая семантический.

Аналитик проводит анализ агрегированной НМИ и содержимого коллекций документов. Он обращается к витринам наукометрических данных через SQL, строит дашборды и отчеты в BI¹-инструментах (агрегации по времени, организациям, направлениям), выполняет детализирующий анализ (drill-down) от агрегированных показателей к отдельным сущностям, а также выгружает подмножества данных в интерактивную среду для расширенной обработки с фиксацией даты среза и методики расчета. К задачам аналитика относится полнотекстовый и семантический поиск по НТИ с фильтрацией по метаданным и формирование структурированных извлечений для последующего анализа.

Инженер данных отвечает за процессы поступления данных и воспроизводимое обновление витрин и индексов: интеграцию внешних источников (API, выгрузки в JSON/CSV, веб-ресурсы и реестры), настройку и сопровождение конвейеров ETL/ELT, нормализацию и очистку, контроль качества загрузки, оркестрацию (автоматизированную обработку данных по расписанию и событию), формирование структурированных представлений НТИ и витрин НМИ, индексацию в поисковом контуре и поддержку наблюдаемости процессов с фиксацией версий кода и параметров расчета.

Администратор обеспечивает эксплуатацию инфраструктуры и данных: управляет учетными записями, ролями и политиками доступа, настраивает федеративную аутентификацию и интеграцию с институциональными провайде-

¹ Business intelligence (BI) — это комплекс методов и компьютерных инструментов для преобразования данных в наглядные отчёты и дашборды, помогающие принимать управленческие решения.

рами идентификации, ведет аудит действий, обеспечивает мониторинг сервисов, резервное копирование и восстановление (базы данных, объектное хранилище, поисковые индексы), управление вычислительными ресурсами и жизненным циклом данных, а также обновление компонентов без нарушения доступности.

3. КОМПОНЕНТЫ СИСТЕМЫ, ИХ ФУНКЦИИ И РЕШЕНИЯ

Для реализации разнообразных запросов на основе рассмотренных выше пользовательских ролей и сценариев взаимодействия с системой предлагается модульная архитектура системы, объединяющая следующие контуры: доступа и управления, хранения и каталогизации, обработки, вычислений и пользовательской аналитики, эксплуатации и устойчивости. Такое разделение отражает различия в природе управляемых объектов и позволяет масштабировать и развивать подсистемы независимо при сохранении единой модели идентификации и управления доступом. Для каждого компонента далее приводятся функция, ключевые требования, проанализированные альтернативы и обоснование выбора.

3.1. Контур доступа и управления

3.1.1. Пользовательский портал и прикладной слой. Обеспечивают управление жизненным циклом объектов и поддерживают единый интерфейс поиска и навигации по объектам всех типов в соответствии с правами пользователя. Из сценариев исследователей вытекают требования поиска и просмотра объектов всех типов, загрузки и публикации с фиксацией версий, формирования карточек и связей между объектами. К нефункциональным требованиям относятся ролевое разграничение доступа, воспроизводимость операций публикации, отказоустойчивость при росте числа пользователей и аудит ключевых действий.

При реализации портала были проанализированы два подхода. Первый состоит в адаптации готовой системы класса каталога данных, что обеспечивает быстрый старт, но требует существенной доработки для поддержки НТИ и НМИ. Второй подход предполагает разработку портала и сервисного API как самостоятельного слоя поверх специализированных подсистем, что обеспечивает более высокую гибкость системы при повышении стоимости разработки. Наиболее предпочтительным представляется гибридный вариант, в котором готовая система выступает как компонент интерфейса и каталога, а критичные функции

управления жизненным циклом, связностью, политиками доступа и аудитом реализуются в отдельном прикладном сервисе.

3.1.2. Сервисный API. Предоставляет программный доступ к операциям управления и потребления данных, включая загрузку и выгрузку объектов, доступ к метаданным, выполнение поисковых запросов и выборку структурированных данных, а также обеспечивает интеграцию с внешними системами и автоматизацию пользовательских сценариев.

3.1.3. Шлюз доступа. Обеспечивает пользователю единый вход в систему, осуществляет маршрутизацию запросов к portalу и API, транспортное шифрование, ограничение частоты и объема запросов, централизованный аудит. При этом требуются совместимость с подсистемой идентификации, а также отказоустойчивость и горизонтальное масштабирование. Был проведен сравнительный анализ решений NGINX², Traefik³ Proxy и Kong Gateway⁴, в результате которого определен NGINX как наиболее развитое и предсказуемое решение в условиях высокой нагрузки и с точки зрения требований к безопасности, представляющее собой обратный прокси с развитыми механизмами управления трафиком и устойчивой практикой эксплуатации, тогда как Traefik демонстрирует менее предсказуемое поведение под нагрузкой, а Kong избыточен при отсутствии потребности в расширенном управлении жизненным циклом API.

Основные риски, такие как единая точка отказа, деградация при крупной выдаче, обход контрольной точки, снижаются отказоустойчивой конфигурацией с балансировкой, разнесением контуров доставки и сетевой сегментацией.

3.1.4. Идентификация, аутентификация и авторизация (Authentication and authorization infrastructure – AAI). Реализует единое управление учетными записями и сквозными политиками доступа к portalу и API, объектам хранилищ, поисковым индексам и витринам с поддержкой федеративной аутентификации и ролевого разграничения. Требуются многоролевая модель, режимы доступа: открытый, ограниченный, по запросу и отсроченная публикация, аудит решений об авторизации и совместимость со стандартами OpenID Connect и SAML. Был

² <https://nginx.org/en/docs/>

³ <https://doc.traefik.io/traefik/>

⁴ <https://docs.konghq.com/gateway/>

проведен сравнительный анализ решений Keycloak⁵, WSO2 Identity Server⁶, Gluu Server⁷, Shibboleth⁸, SimpleSAMLphp⁹ и Dex¹⁰. В итоге анализа был выбран Keycloak, поскольку он сбалансированно покрывает требования за счет поддержки стандартных протоколов, функциональной модели ролей и групп, развитых средств администрирования и широкого применения в контейнерных средах, тогда как Shibboleth и SimpleSAMLphp требуют дополнительного слоя для сквозных ролевых политик, Dex ограничен в управлении пользователями, а WSO2 и Gluu избыточно сложны в сопровождении. Основные риски связаны с критичностью компонента для доступности, ошибками конфигурации политик и рассогласованием атрибутов при федерации; они снижаются отказоустойчивым развертыванием, формализацией ролевой модели с аудитом изменений и правилами сопоставления атрибутов.

3.2. Контур хранения и каталогизации

3.2.1. Распределенное объектное хранилище. Предназначено для приема, хранения и выдачи больших массивов экспериментальных и расчетных данных и оригиналов документов НТИ. Требуется горизонтальное масштабирование по объему и пропускной способности, репликация и контроль целостности, версионирование, управление жизненным циклом и стандартизированные интерфейсы массовой выгрузки и программного доступа, поддерживающие сценарии вычислений рядом с данными. Был проведен сравнительный анализ решений Ceph¹¹, MinIO¹² и Hadoop HDFS¹³, по результатам которого был выбран MinIO, поскольку объектная модель обеспечивает однозначную адресацию, удобство масштабирования и эффективную массовую выдачу через прикладной слой; по сравнению с Ceph он проще в эксплуатации, а по сравнению с Hadoop HDFS уни-

⁵ <https://www.keycloak.org/documentation>

⁶ <https://is.docs.wso2.com/>

⁷ <https://docs.gluu.org/>

⁸ <https://shibboleth.atlassian.net/wiki/spaces/IDP5/overview>

⁹ <https://simplesamlphp.org/docs/>

¹⁰ <https://dexidp.io/docs/>

¹¹ <https://docs.ceph.com/>

¹² <https://min.io/docs/minio/linux/index.html>

¹³ <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html>

версальнее в интеграции с порталом, программным интерфейсом и вычислительными окружениями и не требует промежуточных слоев для внешней выдачи. Основные риски связаны с деградацией при множестве мелких объектов, обеспечением отказоустойчивости и межплощадочной репликацией; они снижаются регламентами упаковки данных, версионированием, отказоустойчивой топологией и политиками жизненного цикла.

3.2.2. Реляционное хранилище метаданных и связей. Хранит метаданные объектов, их связи и этапы жизненного цикла, идентификаторы, авторство, версии, происхождение, правила доступа, ссылки на физическое размещение. Выступает системообразующим слоем, связывающим контуры хранения, индексирования и аналитики. Требуются строгие ограничения целостности, транзакционность, предсказуемое поведение при конкурентной записи, развитые механизмы обновления миграции схемы, экспорт метаданных по стандартным протоколам и устойчивая идентификация опубликованных объектов [6]. Был проведен сравнительный анализ решений PostgreSQL¹⁴, MariaDB¹⁵, MySQL¹⁶ и CockroachDB¹⁷. В результате анализа был выбран PostgreSQL, поскольку он в наибольшей степени покрывает требования к реляционному хранилищу и поддерживает гибкое хранение полуструктурированных атрибутов без отказа от реляционной модели, что важно для итеративно развивающейся схемы метаданных. CockroachDB избыточен на базовом этапе, когда распределенность достигается на уровне сервисов. Риски: критичность контура, рост сложности графа связей, неконтролируемый рост полуструктурированных полей – снижаются отказоустойчивым развертыванием, индексированием и выделенными представлениями, фиксацией обязательного набора полей и управляемыми миграциями.

3.2.3. Аналитическое хранилище структурированных представлений данных. Обеспечивает хранение структурированных слоев НТИ, включающих извлеченные сущности, атрибуты и связи, а также слоев НМИ, включающих таблицы фактов и измерений, метрики, агрегаты, параметры расчета и дату среза, для SQL-доступа и отчетности. Требуются стабильный SQL-доступ, регламентное

¹⁴ <https://www.postgresql.org/docs/>

¹⁵ <https://mariadb.com/kb/en/documentation/>

¹⁶ <https://dev.mysql.com/doc/>

¹⁷ <https://www.cockroachlabs.com/docs/>

обновление витрин, фиксация параметров расчета и предсказуемая производительность при большом числе параллельных агрегирующих чтений. Проведен сравнительный анализ решений PostgreSQL, ClickHouse¹⁸ и Greenplum¹⁹. С учетом ориентации на локальное развертывание на этапе моделирования системы был выбран PostgreSQL, поскольку он минимизирует число классов СУБД, упрощает администрирование и обеспечивает строгую схемную модель витрин. Колоночные и MPP-решения дают более высокую эффективность при росте объема и интенсивности запросов, однако повышают операционную сложность и оправданы на этапах масштабирования системы, что соответствует принципу специализации хранилищ под профиль нагрузки [10]. Основные риски связаны с обеспечением производительности при росте витрин, расхождением с транзакционным контуром и усложнением схемы; они снижаются денормализованными и материализованными представлениями, индексированием по измерениям, фиксацией даты среза и логическим разделением схем.

3.2.4. Индекс полнотекстового и семантического поиска. Обеспечивает полнотекстовый поиск по корпусу НТИ, метаданным и описаниям объектов с фильтрацией по атрибутам и выдачей фрагментов, а также хранение векторных представлений фрагментов для поиска по смысловой близости; в связке реализуется гибридное извлечение для семантического поиска и поисково-генеративных сценариев [11]. Требуются согласованная привязка результатов к структурированному слою через идентификаторы объектов, поддержка запросов ближайших соседей и гибридного ранжирования лексической [12] и семантической [9] релевантности, масштабируемость, предсказуемая латентность и резервное копирование индексов. Было выполнено сравнение решений OpenSearch²⁰, Apache Solr²¹ в сочетании с отдельным векторным индексом и Vespa²². В итоге был выбран OpenSearch, поскольку он реализует полнотекстовый и семантический режимы в единой индексной модели и общей инфраструктуре резервирования, мониторинга и масштабирования; Apache Solr требует отдельного векторного слоя с синхронизацией двух индексов, а Vespa концептуально подходит, но

¹⁸ <https://clickhouse.com/docs>

¹⁹ <https://greenplum.org/>

²⁰ <https://opensearch.org/docs/>

²¹ <https://solr.apache.org/guide/>

²² <https://docs.vespa.ai/>

сложнее во внедрении для базовой итерации. Основные риски связаны с деградацией при росте индекса, расхождением индекса и метаданных, восстановлением после сбоев и качеством семантического поиска; они снижаются проектированием индексной схемы и регламентами переиндексации, фиксацией единого идентификатора объекта, снимками индексов в объектное хранилище и формализацией конвейера подготовки фрагментов с версионированием эмбеддингов.

3.3. Контур обработки и обновления данных

Этот контур реализует воспроизводимое исполнение процедур загрузки, преобразования и публикации данных для всех доменов. Он включает оркестрацию конвейеров (ETL/ELT, расписания, зависимости, повторные попытки, журналирование), извлечение и нормализацию (подключение источников, сопоставление схем, дедупликация, фиксация происхождения), обработку НТИ (извлечение текста, разбиение на фрагменты, выделение сущностей и связей, тематическая классификация с направлением оригиналов в объектное хранилище, полного текста и фрагментов – в поисковый контур, структурированных извлечений – в аналитическое хранилище) и обработку НМИ (унификация и пересчет показателей, формирование таблиц фактов и измерений и витрин с фиксацией источников, даты среза и параметров расчета). Поскольку домены проходят схожие стадии жизненного цикла, целесообразен единый оркестратор; при этом доменно-специфическая логика реализуется как отдельные задачи, объединяемые в формальные конвейеры.

Из сценариев инженера данных, аналитика и администратора следуют требования управления зависимостями и регламентами обновления, запуска по расписанию и событию, повторных попыток, фиксации параметров и результатов, централизованной наблюдаемости и масштабирования ресурсоемких этапов. Проведен сравнительный анализ решений Apache Airflow²³, Argo Workflows²⁴ и Prefect²⁵. В результате был выделен Apache Airflow, поскольку раз-

²³ <https://airflow.apache.org/docs/>

²⁴ <https://argo-workflows.readthedocs.io/en/latest/>

²⁵ <https://docs.prefect.io/>

витая модель направленных ациклических графов и обширная экосистема операторов соответствуют доминированию регламентных процессов и большому числу зависимостей и расписаний; Argo Workflows ориентирован на контейнерные вычислительные цепочки, а приоритетом Prefect выступает удобство разработки. Основные риски связаны с критичностью оркестратора, каскадными сбоями из-за некорректных зависимостей и блокировкой регламентных процессов ресурсоемкими задачами; они снижаются отказоустойчивым развертыванием и мониторингом, стандартизацией шаблонов конвейеров и разделением доменных потоков, а также разделением очередей и квотированием ресурсов.

3.4. Контур вычислений и пользовательской аналитики

3.4.1. Интерактивные вычислительные рабочие места. Предоставляют изолированные вычислительные среды для анализа данных и воспроизводимых исследований с доступом к объектному хранилищу, витринам и поисковым индексам. Требуется выполнение кода с наследованием политик доступа через AAI, управление ресурсами, включая квоты на процессорное время, память и дисковое пространство, предсказуемость при одновременной работе множества пользователей, согласованность окружений и подключение к данным без обязательного полного копирования. Было выполнено сравнение решений JupyterHub²⁶, Kubeflow Notebooks²⁷ и Apache Zeppelin²⁸, а также варианта прямого доступа к вычислительным узлам. В итоге был выбран JupyterHub с развертыванием в контейнерной инфраструктуре, поскольку он обеспечивает многопользовательскую модель, изоляцию сессий, квотирование и единую аутентификацию; Kubeflow Notebooks ориентирован на контуры машинного обучения и повышает сложность, Apache Zeppelin менее универсален, а прямой доступ снижает воспроизводимость и контроль. Основные риски связаны с конкуренцией за ресурсы, разнообразием окружений и безопасностью доступа к данным; они снижаются разделением и квотированием ресурсов, стандартными профилями окружений, интеграцией с политиками доступа, сетевой сегментацией и аудитом.

²⁶ <https://jupyterhub.readthedocs.io/en/stable/>

²⁷ <https://www.kubeflow.org/docs/components/notebooks/>

²⁸ <https://zeppelin.apache.org/docs/>

3.4.2. SQL-слой доступа. Обеспечивает унифицированный SQL-доступ к витринам и, при необходимости, к данным объектного хранилища, позволяет использовать стандартные инструменты SQL/BI без прямого подключения к внутренним хранилищам. Требуется стабильный доступ к витринам НМИ и структурированному слою НТИ, поддержка параллельных запросов BI-нагрузки, управление доступом по ролевой модели и согласованность определений витрин. Был проведен сравнительный анализ решений Dremio²⁹, Trino³⁰ и PrestoDB³¹. Был выбран Dremio на базе открытого ядра dremio-oss, поскольку он дополняет SQL-движок слоем каталогизации и публикации представлений, что соответствует требованиям самообслуживания аналитиков; Trino и PrestoDB не задают прикладной модели публикации наборов данных. Фиксируется ограничение использования открытой кодовой базы; резервным сценарием замещения служит Trino. Основные риски связаны с различием открытой базы и условий эксплуатации дистрибутивов и с ограничением набора коннекторов; они снижаются фиксацией варианта поставки и предварительной валидацией требуемых источников.

3.4.3. BI-доступ и отчетность. Обеспечивает подключение BI-инструментов к витринам и реализует сценарии отчетности с управляемой публикацией дашбордов и режимом самообслуживания. В результате сравнительного анализа решений DataLens³², Apache Superset³³ и Metabase³⁴ был отобран DataLens в открытой реализации, поскольку он изначально ориентирован на построение дашбордов и работу с типовыми аналитическими источниками, включая PostgreSQL и ClickHouse. Риск ограниченности коннекторов и функциональности открытой версии снижается фиксацией перечня поддерживаемых источников и использованием SQL-слоя как стабильного интерфейса, позволяющего заменить BI-инструмент без изменения контуров хранения и обработки.

3.5. Контур эксплуатации и обеспечения устойчивости

²⁹ <https://docs.dremio.com/>

³⁰ <https://trino.io/docs/current/>

³¹ <https://prestodb.io/docs/current/>

³² <https://datalens.ru/opensource/docs>

³³ <https://superset.apache.org/docs/intro>

³⁴ <https://www.metabase.com/docs/>

3.5.1. Наблюдаемость. Для контроля состояния сервисов была определена связка Prometheus³⁵ и Grafana³⁶ для сбора и визуализации эксплуатационных метрик, дополненная OpenSearch для централизованного хранения и анализа логов, что охватывает как метрики, так и событийные данные и сохраняет единство технологического стека за счет повторного использования поискового компонента. Основные риски связаны с неполнотой телеметрии, ростом объема логов и отказом самого контура наблюдаемости; они снижаются стандартизацией форматов и корреляционных идентификаторов запросов, политиками хранения и ротации и отказоустойчивым развертыванием.

3.5.2. Контейнеризация и управление ресурсами. Обеспечивает изоляцию и масштабирование сервисов и вычислительных окружений с логическим разделением ресурсов, квотированием и сетевыми политиками. Требуются среда исполнения, горизонтальное масштабирование, квотирование, сегментация ресурсов между системными сервисами и пользовательскими вычислениями, сетевая изоляция и воспроизводимость окружений. Был проведен сравнительный анализ решений Kubernetes³⁷, OpenShift³⁸ и Nomad³⁹, в результате был выделен Kubernetes. Как фактический стандарт оркестрации он предоставляет модель планирования, отдельные пулы узлов для системных сервисов и пользовательских сессий, квоты и лимиты и сетевые политики [13]. При этом OpenShift увеличивает сложность, а Nomad не обеспечивает требуемого уровня поддержки сложных многосервисных систем данных. Основные риски связаны с конкуренцией за ресурсы, сложностью эксплуатации и ошибками политик безопасности, обходом контрольных точек и деградацией при обновлениях; они снижаются выделением пулов узлов, квотами, стандартизацией профилей развертывания, сетевой сегментацией и поэтапными обновлениями.

4. ИТОГОВЫЙ КОМПОНЕНТНЫЙ СОСТАВ

³⁵ <https://prometheus.io/docs/>

³⁶ <https://grafana.com/docs/>

³⁷ <https://kubernetes.io/docs/>

³⁸ <https://docs.openshift.com/>

³⁹ <https://developer.hashicorp.com/nomad/docs>

Предложенная архитектура системы объединяет хранение и обслуживание трех классов объектов: научных датасетов, НТИ и НМИ – по многоуровневому принципу, где каждый контур решает определенный класс задач, а согласованность достигается единой моделью идентификации, централизованным управлением доступом и воспроизводимыми процессами обработки.

Контур доступа и управления включает портал и сервисный API, шлюз доступа и подсистему на базе Keycloak, обеспечивающую единый вход и ролевую модель прав для всех сервисов.

Контур хранения и каталогизации объединяет объектное хранилище MinIO (датасеты и оригиналы), реляционное хранилище метаданных и связей на базе PostgreSQL как системообразующий слой, аналитические представления НМИ и структурированный слой НТИ в PostgreSQL с SQL-доступом, а также поисковый контур OpenSearch для полнотекстового и семантического извлечения. Архитектурные решения предусматривают хранение оригиналов, транзакционную модель метаданных, аналитические витрины и поисковые представления, связывая их едиными идентификаторами объектов и регламентами обновления. Контур обработки реализуется единым оркестратором Apache Airflow, управляющим обновлением витрин НМИ, формированием представлений НТИ и переиндексацией поиска с фиксацией статусов и параметров запусков.

Контур вычислений и аналитики включает решения JupyterHub (интерактивные рабочие места), SQL-слой Dremio (унифицированный доступ к витринам) и BI-инструмент DataLens (визуальная аналитика и отчетность), опирающиеся на единый SQL-контракт витрин. Контур эксплуатации обеспечивается наблюдаемостью на базе Prometheus и Grafana и анализом логов в OpenSearch, комбинированной стратегией резервного копирования источников истины и производных контуров, а также контейнеризацией и управлением ресурсами на базе Kubernetes с разделением системных и пользовательских нагрузок. В совокупности компоненты взаимодействуют через стандартизированные интерфейсы, что покрывает полный спектр сценариев работы с научными данными и допускает поэтапную эволюцию системы без пересмотра ключевых контрактов доступа и модели данных.

ЗАКЛЮЧЕНИЕ

Представлена интегрированная система управления научными данными, объединяющая научные датасеты, научно-техническую и наукометрическую информации. Предложены технологические решения ее базовых компонентов. Показано, что типы обрабатываемых данных различаются по структурированности, масштабам, профилю доступа и требованиям к метаданным и воспроизводимости, что исключает эффективное управление ими в рамках единого универсального хранилища или единого класса инструментов [1, 10]. Предложенное решение относится к классу полихранилищ, в которых запрос обслуживается несколькими специализированными подсистемами при едином формате доступа [14]. На основе сценариев использования ключевых ролей определены функциональные и нефункциональные требования к компонентам, включая масштабирование, безопасность, воспроизводимость обработки, поддержку интерактивных вычислений и предоставление данных через разные каналы потребления.

Определен состав модулей системы, объединенных архитектурой, поддерживающей разграничение типов данных и их производных представлений. Объектное хранилище используется для датасетов и оригиналов, реляционный контур метаданных – для целостности, версий и связности, аналитические представления – для SQL-доступа, поисковый контур – для полнотекстового и семантического извлечения; сквозными механизмами интеграции выступают единый прикладной слой управления жизненным циклом объектов, централизованная аутентификация и авторизация и единый оркестратор конвейеров обработки. На основе анализа альтернатив сформирован набор решений с возможностью локального развертывания, в частности MinIO, PostgreSQL, OpenSearch, Keycloak, Apache Airflow, JupyterHub, Dremio и DataLens, шлюз доступа NGINX, а также Prometheus, Grafana и Kubernetes для эксплуатации.

Предложенная архитектура представляет воспроизводимый шаблон проектирования системы управления научными данными, допускающий поэтапное развитие без нарушения режимов доступа к данным и управления объектами данных.

СПИСОК ЛИТЕРАТУРЫ

1. Borgman C.L. Big Data, Little Data, No Data: Scholarship in the Networked World. Cambridge, MA: MIT Press, 2015. 416 p. ISBN 978-0-262-02856-1
2. Wilkinson M.D., Dumontier M., Aalbersberg I.J. et al. The FAIR Guiding Principles for scientific data management and stewardship // Scientific Data. 2016. Vol. 3. Article 160018. <https://doi.org/10.1038/sdata.2016.18>
3. UNESCO Recommendation on Open Science. Paris: UNESCO, 2021. 36 p. <https://doi.org/10.5281/zenodo.5741832>
4. Гарев К.В. Определение принципов проектирования отечественной платформы обмена научными датасетами // Программные продукты и системы. 2025. Т. 38, № 1. С. 197–206. <https://doi.org/10.15827/0236-235X.149.197-206>
5. Савин Г.И., Шабанов Б.М., Баранов А.В. и др. Об использовании федеральной научной телекоммуникационной инфраструктуры для суперкомпьютерных вычислений // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2020. Т. 9, № 1. С. 20–35. <https://doi.org/10.14529/cmse200102>
6. DataCite Metadata Working Group. DataCite Metadata Schema Documentation for the Publication and Citation of Research Data and Other Research Outputs. Version 4.4. DataCite e.V., 2021. <https://doi.org/10.14454/3w3z-sa82>
7. Гарев К.В. Сравнительный анализ методов сжатия табличных данных // Программные продукты и системы. 2024. Т. 37, № 2. С. 170–177. <https://doi.org/10.15827/0236-235X.142.170-177>
8. Nasar Z., Jaffry S.W., Malik M.K. Information extraction from scientific articles: a survey // Scientometrics. 2018. Vol. 117, No. 3. P. 1931–1990. <https://doi.org/10.1007/s11192-018-2921-5>
9. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks // Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. P. 3982–3992. <https://doi.org/10.18653/v1/D19-1410>
10. Stonebraker M., Çetintemel U. «One Size Fits All»: An Idea Whose Time Has Come and Gone // Proceedings of the 21st International Conference on Data Engineering (ICDE'05). Tokyo: IEEE Computer Society, 2005. P. 2–11.

<https://doi.org/10.1109/ICDE.2005.1>

11. Lewis P., Perez E., Piktus A. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // *Advances in Neural Information Processing Systems*. 2020. Vol. 33. P. 9459–9474. <https://doi.org/10.48550/arXiv.2005.11401>

12. Robertson S., Zaragoza H. The Probabilistic Relevance Framework: BM25 and Beyond // *Foundations and Trends in Information Retrieval*. 2009. Vol. 3, No. 4. P. 333–389. <https://doi.org/10.1561/15000000019>

13. Burns B., Grant B., Oppenheimer D. et al. Borg, Omega, and Kubernetes // *Communications of the ACM*. 2016. Vol. 59, No. 5. P. 50–57. <https://doi.org/10.1145/2890784>

14. Duggan J., Elmore A.J., Stonebraker M. et al. The BigDAWG Polystore System // *ACM SIGMOD Record*. 2015. Vol. 44, No. 2. P. 11–16. <https://doi.org/10.1145/2814710.2814713>

REQUIREMENTS AND TECHNICAL SOLUTIONS FOR A SCIENTIFIC DATA MANAGEMENT SYSTEM ARCHITECTURE

K. V. Garev^[0009-0001-1961-3195]

National Research Centre "Kurchatov Institute", Moscow, Russia

kv@garev.ru

Abstract

This paper addresses the problem of defining requirements for the architecture of a scientific data management system and selecting technical solutions for implementing its components. Scientific data are divided into three categories – research datasets, scientific and technical information (STI), and scientometric information (SMI) – which differ in structure, volume, update patterns, access profile, and metadata requirements, complicating their effective management within a single general-purpose store. Based on a data typology and use cases for the key roles (researcher, analyst, data engineer, administrator), requirements for the components are formulated. A comparative analysis of alternative solutions is performed for each component, and a modular architecture is proposed that separates sources of truth from derived representations. A consistent technology stack compatible with on-

premises deployment is determined, along with cross-cutting mechanisms of identification, authorization, and resource management.

Keywords: *scientific data management, research datasets, scientific and technical information, scientometric information, software architecture, system requirements, distributed storage, semantic search.*

REFERENCES

1. Borgman C.L. Big Data, Little Data, No Data: Scholarship in the Networked World. Cambridge, MA: MIT Press, 2015. 416 p. ISBN 978-0-262-02856-1
2. Wilkinson M.D., Dumontier M., Aalbersberg I.J. et al. The FAIR Guiding Principles for scientific data management and stewardship // Scientific Data. 2016. Vol. 3. Article 160018. <https://doi.org/10.1038/sdata.2016.18>
3. UNESCO Recommendation on Open Science. Paris: UNESCO, 2021. 36 p. <https://doi.org/10.5281/zenodo.5741832>
4. Garev K.V. Comparative analysis of tabular data compression methods // Программные продукты и системы [Software & Systems]. 2024. Vol. 37, No. 2. P. 170–177. <https://doi.org/10.15827/0236-235X.142.170-177> (in Russian).
5. Savin G.I., Shabanov B.M., Baranov A.V. et al. On the use of the federal scientific telecommunication infrastructure for supercomputer computations // Vestnik YuUrGU. Ser. Vychislitel'naya matematika i informatika [Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering]. 2020. Vol. 9, No. 1. P. 20–35. <https://doi.org/10.14529/cmse200102> (in Russian)
6. DataCite Metadata Working Group. DataCite Metadata Schema Documentation for the Publication and Citation of Research Data and Other Research Outputs. Version 4.4. DataCite e.V., 2021. <https://doi.org/10.14454/3w3z-sa82>
7. Garev K.V. Comparative analysis of tabular data compression methods // Программные продукты и системы [Software & Systems]. 2024. Vol. 37, No. 2. P. 170–177. <https://doi.org/10.15827/0236-235X.142.170-177> (in Russian).
8. Nasar Z., Jaffry S.W., Malik M.K. Information extraction from scientific articles: a survey // Scientometrics. 2018. Vol. 117, No. 3. P. 1931–1990. <https://doi.org/10.1007/s11192-018-2921-5>

9. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks // Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Hong Kong, 2019. P. 3982–3992. <https://doi.org/10.18653/v1/D19-1410>

10. Stonebraker M., Çetintemel U. «One Size Fits All»: An Idea Whose Time Has Come and Gone // Proceedings of the 21st International Conference on Data Engineering (ICDE'05). Tokyo: IEEE Computer Society, 2005. P. 2–11. <https://doi.org/10.1109/ICDE.2005.1>

11. Lewis P., Perez E., Piktus A. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 9459–9474. <https://doi.org/10.48550/arXiv.2005.11401>

12. Robertson S., Zaragoza H. The Probabilistic Relevance Framework: BM25 and Beyond // Foundations and Trends in Information Retrieval. 2009. Vol. 3, No. 4. P. 333–389. <https://doi.org/10.1561/15000000019>

13. Burns B., Grant B., Oppenheimer D. et al. Borg, Omega, and Kubernetes // Communications of the ACM. 2016. Vol. 59, No. 5. P. 50–57. <https://doi.org/10.1145/2890784>

14. Duggan J., Elmore A.J., Stonebraker M. et al. The BigDAWG Polystore System // ACM SIGMOD Record. 2015. Vol. 44, No. 2. P. 11–16. <https://doi.org/10.1145/2814710.2814713>

СВЕДЕНИЯ ОБ АВТОРЕ



ГАРЕВ Кирилл Валерьевич – выпускник аспирантуры Федерального государственного бюджетного учреждения «Федеральный научный центр Научно-исследовательский институт системных исследований Национального исследовательского центра «Курчатовский институт» (НИЦ «Курчатовский институт» – НИИСИ).

Kirill Valerevich GAREV – graduate of the postgraduate program at the Scientific Research Institute for System Analysis, National Research Centre "Kurchatov Institute".

email: kv@garev.ru

0009-0001-1961-3195

Материал поступил в редакцию 27 июня 2026 года