РЕФАКТОР ИКС

ООО «Рефактор-ИКС»

Многоцелевой ситуационный центр «КОЛОС»

ТЕХНИЧЕСКАЯ АРХИТЕКТУРА

Документация с описанием технической архитектуры программного обеспечения

г. Москва,
2025

Оглавление

Список сокращений

Сокращение	Определение
БД	база данных
ИС	информационная система
ПО	программное обеспечение
ПОС	подсистема обмена событиями
СУБД	система управления базами данных
СЦ	ситуационный центр
API	Application Programming Interface — программный интерфейс взаимодействия между компонентами системы
C4	нотация визуального моделирования архитектуры (Context, Container, Component, Code)
CSV	Comma-Separated Values — текстовый формат представления табличных данных
HTML	HyperText Markup Language — язык разметки гипертекста
HTTP / HTTPS	сетевые протоколы передачи данных
JSON	JavaScript Object Notation — текстовый формат обмена структурированными данными
Linux	операционная система, основанная на ядре Linux
ORM	Object-Relational Mapping — технология сопоставления объектов с реляционной моделью данных
REST	архитектурный стиль взаимодействия компонентов через унифицированный API
SPA	Single Page Application — одностраничное веб-приложение
UI	User Interface — пользовательский интерфейс
XML	eXtensible Markup Language — расширяемый язык разметки

Термины и определения

Термин	Определение
Журнал событий	Перечень (список) событий, произошедших за текущую смену или перешедших в нее из предыдущей как незакрытые
Информационная система	Совокупность содержащейся в базах данных информации и обеспечивающих ее обработку информационных технологий и технических средств (ФЗ РФ №149-ФЗ от 27 июля 2006 г. «Об информации, информационных технологиях и защите информации»)
Информация	Сведения (сообщения, данные) независимо от формы их представления (ФЗ РФ №149-ФЗ от 27 июля 2006 г. «Об информации, информационных технологиях и защите информации»)
Карта происшествий	Карта, на которой отображены происшествия и события, произошедшие за смену и незакрытые с предыдущей смены
Компонент	Элемент внутри системы, имеющий дискретную структуру (например, компоновочный или программный модуль), рассматриваемый на конкретном уровне анализа
Лента происшествий	Происшествия за текущую смену
Объект информатизации	Совокупность информационных ресурсов, средств и систем обработки информации, используемых в соответствии с заданной информационной технологией, а также средств их обеспечения, помещений или объектов (зданий, сооружений, технических средств), в которых эти средства и системы установлены, или помещений и объектов, предназначенных для ведения конфиденциальных переговоров
Подсистема	Любая система, входящая в другую (большую) систему
Происшествие	Полученная из внешних источников информация о происшествии, противоправных действиях, вызовах специальных служб и т. п. В системе используется для информирования сотрудников, но не является для них руководством к действию. На основе происшествия может создаваться событие
Процесс	Совокупность взаимосвязанных и взаимодействующих видов деятельности, преобразующих входы в выходы
Сводка	Отчет о событиях и мероприятиях, произошедших за определенный период
Система	1. Сложная структура, состоящая из одного или нескольких процессов, технических и программных средств, устройств и персонала, удовлетворяющая установленным потребностям или целям. 2. Комбинация взаимодействующих элементов, организованных для достижения одной или нескольких поставленных целей.
Ситуация	Последовательность объектов в системе, связанных с одним событием — исходное событие, происшествия, поручения, комментарии и т. п.

Термин	Определение
Событие	Основной объект в системе, описывает что-либо требующее реакции пользователей системы — выполнения действий вне системы. К событию могут добавляться комментарии и уточнения, поручения и данные об их исполнении.
Элемент системы	Представитель совокупности элементов, образующих систему. Элемент системы является отдельной частью системы, которая может быть создана для выполнения заданных требований.

1 Введение

1.1 Назначение и цель документа

Данный документ описывает техническую архитектуру многоцелевого ситуационного центра «КОЛОС» (далее — СЦ «КОЛОС»), который обеспечивает мониторинг, оперативное управление и анализ данных из различных ведомственных и иных внешних источников.

Документ содержит информацию о ключевых подсистемах, используемых технологиях, структуре базы данных и логике взаимодействия компонентов.

1.2 Область применения

Настоящая документация адресована:

- Архитекторам и разработчикам, которым необходимо понимать внутреннее устройство СЦ «КОЛОС» и принципы взаимодействия его подсистем для дальнейшего сопровождения;
- Интеграторам и инженерам, которым важно знать общую структуру и способы взаимодействия СЦ «КОЛОС» с внешними системами;
- Руководителям и профильным специалистам, отвечающим за контроль ключевых архитектурно-технических параметров и корректное функционирование системы.

1.3 Ссылки и используемые источники

- ГОСТ Р 57100-2016 / ISO/IEC/IEEE 42010:2011 «Системная и программная инженерия. Описание архитектуры»: используется в качестве методической основы при описании архитектуры, включая определение архитектурных представлений (viewpoints), элементов архитектуры, а также их взаимосвязей.
- Внутренняя документация СЦ «КОЛОС». Включает эксплуатационные и технические руководства, описания пользовательских сценариев, интерфейсов и регламентов взаимодействия.

- Методология C4 https://c4model.com/ — используется для графического описания архитектуры на трёх уровнях абстракции: контекста, контейнеров и компонентов.

2 Логическая и компонентная архитектура Системы в нотации C4

2.1 Уровень 1: контекст архитектуры системы «КОЛОС»

Диаграмма контекста, которая описывает взаимодействие пользователя с системой «КОЛОС», представлена на рисунке ниже:

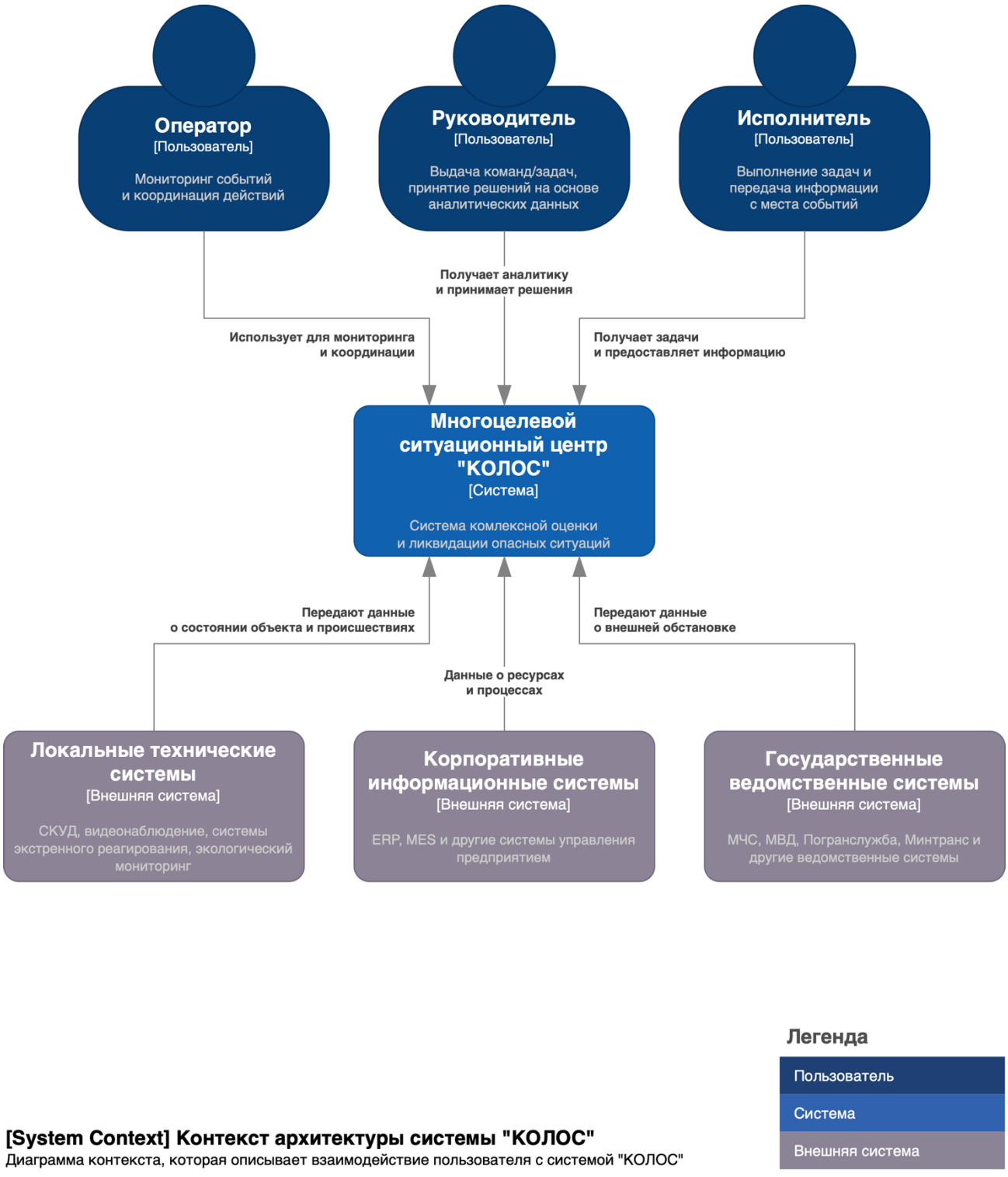


Рисунок 1 — уровень 1: контекст

На 1-м уровне представлена диаграмма контекста, которая описывает универсальное взаимодействие пользователей с многоцелевым Ситуационным центром КОЛОС и его интеграцию с логически сгруппированными внешними системами.

Пользователи системы:

- Оператор — сотрудник, осуществляющий мониторинг событий и происшествий, координирующий действия различных служб и подразделений. Операторы обрабатывают поступающую информацию, создают события, назначают задачи исполнителям.
- Руководитель — лицо, принимающее решения на основе аналитических данных и сводок, предоставляемых системой. Использует систему для получения комплексной картины происходящего и контроля эффективности реагирования.
- Исполнитель — сотрудник, выполняющий конкретные задачи «в поле», получающий поручения через систему и отчитывающийся о выполнении. Может передавать информацию с места событий, включая фото и видеоматериалы.

Центральная система:

- Ситуационный центр КОЛОС — многоцелевая информационная система для мониторинга обстановки, комплексной оценки и ликвидации опасных ситуаций. Система может быть развернута в различных контекстах: на промышленных объектах, спортивных сооружениях, в городской среде или на транспортных узлах.

Внешние системы:

- Локальные технические системы — комплекс инфраструктурных и технических средств, обеспечивающих безопасность объекта и мониторинг его состояния, включая:
  - Системы контроля доступа (СКУД, турникеты, проходные);
  - Системы видеонаблюдения (камеры и видеоаналитика);

  - Системы экстренного реагирования (пожарная сигнализация, системы оповещения);
  - Системы экологического мониторинга (датчики загрязнения, метеостанции).
- Корпоративные информационные системы — программные комплексы, используемые для управления бизнес-процессами и ресурсами организации:
  - ERP-системы (управление ресурсами предприятия);
  - MES-системы (управление производственными процессами);
  - Системы документооборота;
  - Другие бизнес-приложения организации.
- Государственные и ведомственные системы — информационные системы государственных органов и ведомств, предоставляющие актуальные данные о внешней обстановке:
  - Информационные системы МЧС (данные о ЧС);
  - Информационные системы МВД (данные о правопорядке);
  - Системы Погранслужбы (контроль пересечения границ);
  - Системы Минтранса (данные о транспорте);
  - Другие специализированные ведомственные системы.

Ситуационный центр КОЛОС построен на веб-технологиях с модульной архитектурой, что обеспечивает гибкий доступ к его ресурсам с любого компьютера без установки дополнительного ПО и легкую адаптацию под различные сценарии использования. Взаимодействие с внешними системами реализовано через универсальную подсистему обмена событиями, которая получает, обрабатывает и сохраняет данные в единой базе данных.

Масштабируемая распределённая архитектура системы обеспечивает возможность создания единого информационного пространства для нескольких региональных ситуационных центров с возможностью их автономной работы для решения локальных задач.

2.2 Уровень 2: контейнеры системы «КОЛОС»

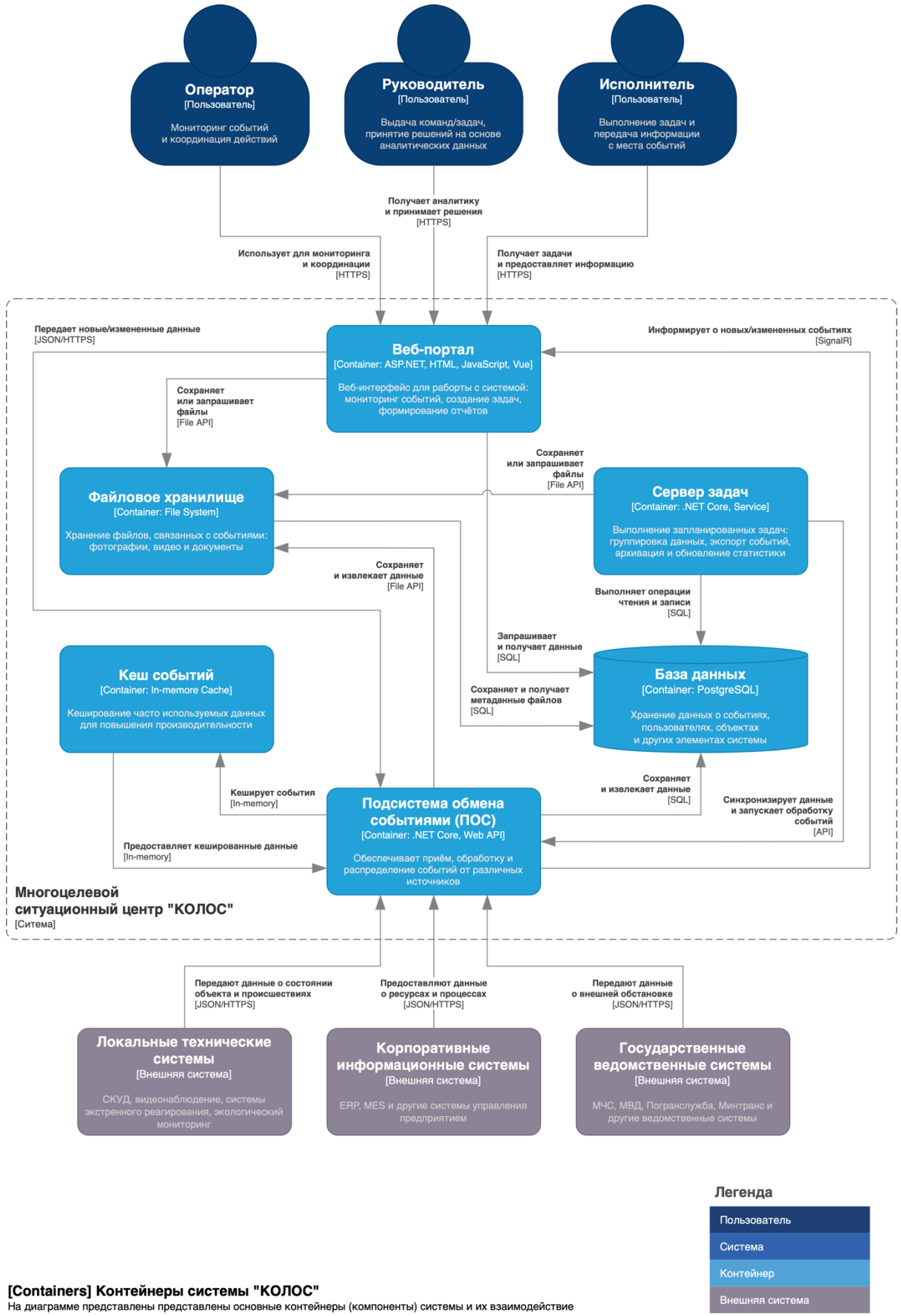


Рисунок 2 — уровень 2: контейнеры

На уровне 2 диаграммы C4 представлены основные контейнеры системы КОЛОС и их взаимодействие. Многоцелевой ситуационный центр КОЛОС состоит из следующих ключевых контейнеров:

Веб-портал

Центральный компонент взаимодействия с пользователями, реализованный на ASP.NET с использованием HTML, JavaScript и Vue.js. Обеспечивает доступ к функциям системы через веб-интерфейс:

- Мониторинг событий и происшествий
- Создание и назначение задач
- Формирование отчетов и аналитических материалов
- Запрос и сохранение данных непосредственно в базе данных
- Обмен файлами с Файловым хранилищем

Взаимодействует со всеми остальными компонентами системы и предоставляет информацию пользователям.

Подсистема обмена событиями (ПОС)

Ключевой компонент системы, реализованный на .NET Core с Web API. Обеспечивает:

- Прием данных от внешних систем и их обработку
- Распределение событий внутри системы
- Взаимодействие с кешем событий для повышения производительности
- Сохранение данных в базе данных и файловом хранилище
- Синхронизация данных с Сервером задач для дальнейшей обработки

ПОС служит интеграционным слоем, связывающим внешние системы с внутренними компонентами ситуационного центра.

Кеш событий

In-memory хранилище для кеширования часто используемых данных. Повышает производительность системы за счет быстрого доступа к актуальной информации без обращения к базе данных.

Сервер задач

Компонент, реализованный как сервис на базе .NET Core, выполняющий запланированные задачи:

- Группировка данных
- Экспорт событий
- Архивация устаревших данных
- Обновление статистики
- Синхронизация данных с ПОС и запуск обработки событий
- Работа с файлами через Файловое хранилище

Сервер задач работает в фоновом режиме, обеспечивая автоматизацию рутинных операций и поддержание системы в актуальном состоянии.

База данных

Централизованное хранилище данных на PostgreSQL, содержащее информацию о:

- Событиях и происшествиях
- Пользователях системы
- Объектах мониторинга
- Метаданных файлов из Файлового хранилища
- Других элементах системы

База данных имеет двунаправленное взаимодействие с Веб-порталом, ПОС и Сервером задач.

Файловое хранилище

Компонент для хранения файлов, связанных с событиями:

- Фотографии с мест происшествий
- Видеозаписи
- Документы и отчеты
- Сохраняет и получает метаданные файлов из Базы данных

Обеспечивает сохранение и предоставление мультимедийных данных по запросу других компонентов системы.

2.3 Уровень 3: компоненты системы «КОЛОС»

2.3.1 Компоненты контейнера «Веб-портал»

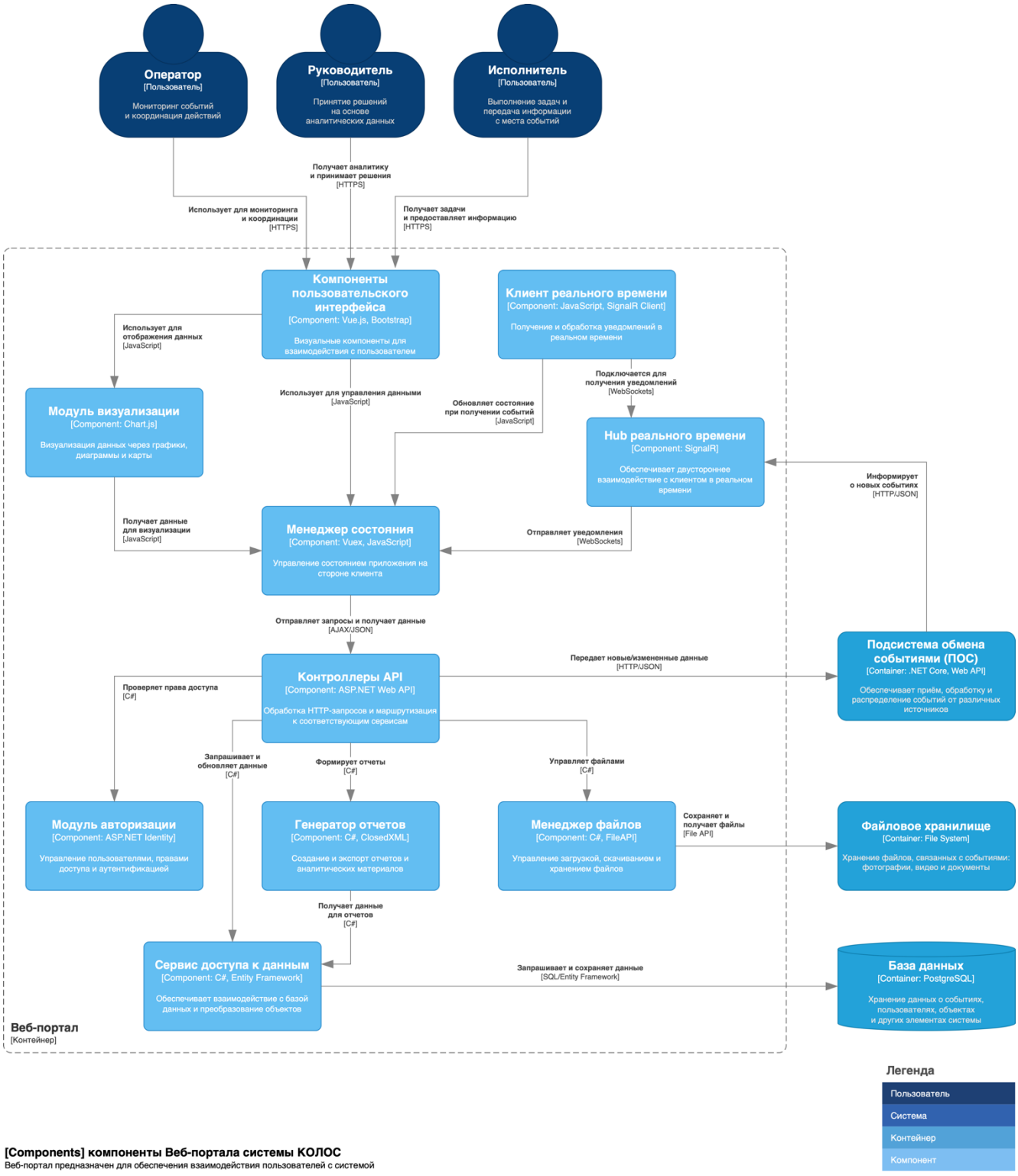


Рисунок 3 — уровень 3: «Веб-портал»

Веб-портал является ключевым контейнером системы КОЛОС, обеспечивающим взаимодействие пользователей с системой. На уровне 3

архитектуры C4 (компонентный уровень) Веб-портал состоит из следующих основных программных компонентов:

2.3.1.1 Клиентские компоненты (выполняются в браузере пользователя)

Компоненты пользовательского интерфейса

Набор компонентов, реализованных на Vue.js с использованием Bootstrap:

- Формы для ввода данных
- Таблицы для отображения информации
- Элементы навигации
- Интерактивные панели управления

Менеджер состояния

Компонент на базе Vuex и JavaScript, управляющий состоянием клиентского приложения:

- Хранение текущих данных приложения
- Обеспечение реактивного обновления интерфейса
- Управление запросами к серверу
- Обработка ошибок и логирование

Модуль визуализации

Компонент для визуализации данных, использующий Chart.js:

- Построение графиков и диаграмм
- Визуализация статистики
- Отображение данных на карте
- Интерактивные информационные панели

Клиент реального времени

JavaScript-компонент, взаимодействующий с SignalR Hub:

- Поддержка WebSocket-соединения с сервером
- Получение уведомлений о новых событиях
- Обработка и отображение оперативных уведомлений

- Обновление интерфейса при получении новых данных

2.3.1.2 Серверные компоненты (выполняются на сервере)

Контроллеры API

Компонент, реализованный на ASP.NET Web API, отвечает за обработку HTTP-запросов от клиентской части приложения. Контроллеры API маршрутизируют запросы к соответствующим сервисам и компонентам, обеспечивая:

- Обработку запросов на получение данных о событиях и происшествиях
- Создание и обновление задач
- Формирование отчетов
- Загрузку и скачивание файлов

Модуль авторизации

Компонент на базе ASP.NET Identity, обеспечивающий:

- Аутентификацию пользователей
- Управление ролями и правами доступа
- Контроль доступа к функциям системы в соответствии с ролевой моделью
- Безопасность взаимодействия с системой

Сервис доступа к данным

Компонент, реализованный на C# с использованием Entity Framework, обеспечивает взаимодействие с базой данных:

- Преобразование запросов из бизнес-логики в SQL-запросы
- Маппинг данных из базы в объекты домена
- Кэширование часто используемых данных
- Реализация бизнес-правил доступа к данным

Менеджер файлов

Компонент для управления файлами, реализованный на C# с использованием File API:

- Загрузка файлов от пользователей
- Сохранение файлов в файловое хранилище
- Скачивание файлов пользователями
- Управление метаданными файлов

Генератор отчетов

Компонент, созданный на C# с использованием библиотеки ClosedXML, отвечает за:

- Формирование аналитических отчетов
- Экспорт данных в различные форматы
- Создание сводок и статистических материалов
- Визуализацию данных для принятия решений

Hub реального времени

Компонент на базе SignalR, обеспечивающий двустороннее взаимодействие с клиентом в реальном времени:

- Мгновенное уведомление пользователей о новых событиях
- Обновление интерфейса без обновления страницы
- Получение информации от ПОС о новых событиях
- Поддержка постоянного соединения с клиентами

2.3.1.3 Взаимодействие с другими контейнерами

Взаимодействие с Подсистемой обмена событиями (ПОС):

a. Контроллеры API передают новые/измененные данные в ПОС.
b. ПОС информирует Hub реального времени о новых событиях.
c. Обмен данными происходит по протоколу HTTP с использованием JSON.

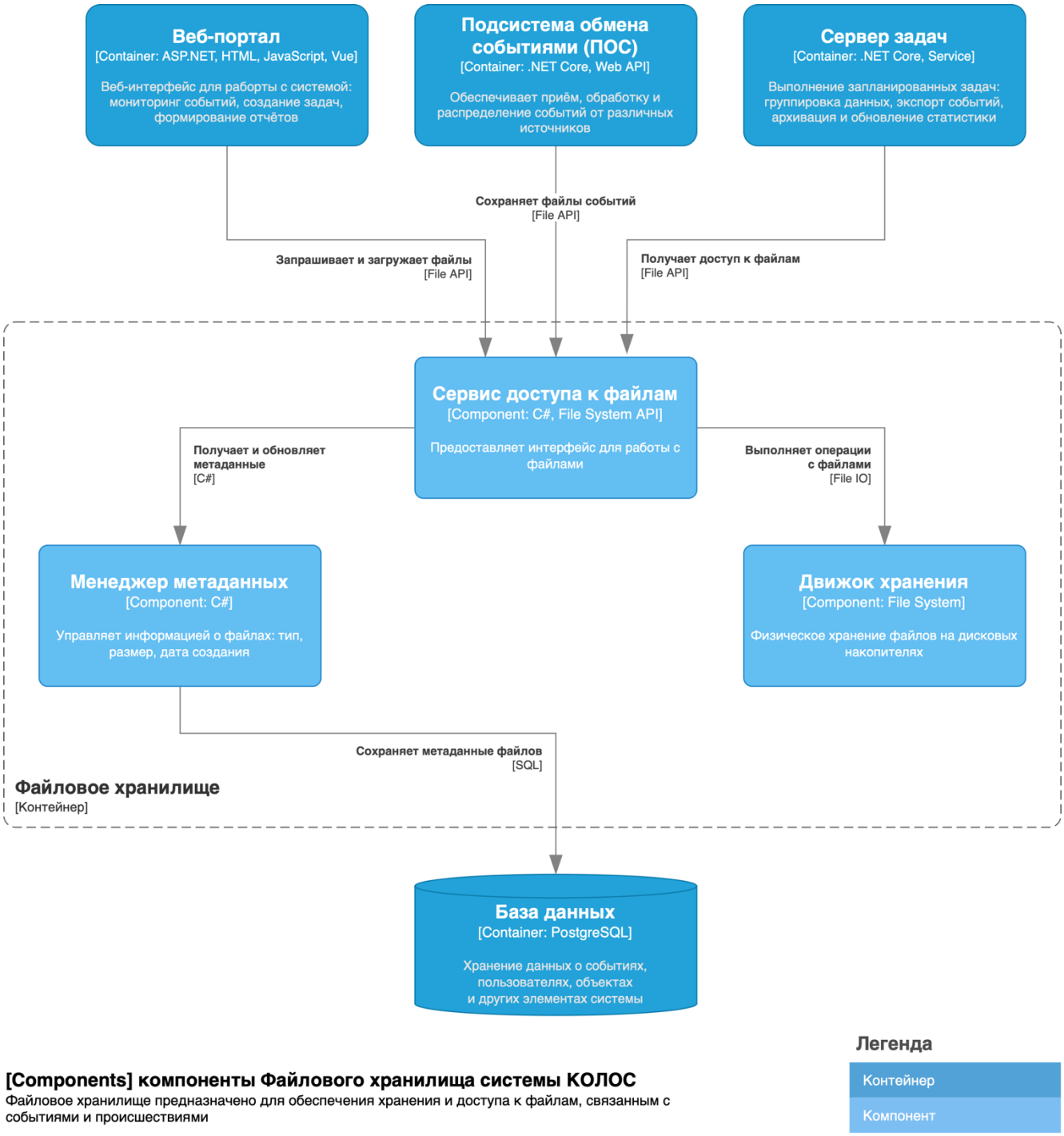
Взаимодействие с Базой данных:

a. Сервис доступа к данным взаимодействует с Базой данных через Entity Framework.
b. Выполняет запросы на чтение и запись данных с использованием SQL.
c. Обеспечивает согласованность и целостность данных.

Взаимодействие с Файловым хранилищем:

a. Менеджер файлов сохраняет и получает файлы из Файлового хранилища.
b. Использует File API для взаимодействия с файловой системой.
c. Управляет метаданными файлов в базе данных.

2.3.2 Компоненты контейнера «Файловое хранилище»



[Components] компоненты Файлового хранилища системы КОЛОС
Файловое хранилище предназначено для обеспечения хранения и доступа к файлам, связанным с событиями и происшествиями

Рисунок 4 — уровень 3: «Файловое хранилище»

Файловое хранилище является базовым контейнером системы «КОЛОС», обеспечивающим хранение и доступ к файлам, связанным с событиями и происшествиями. На уровне 3 архитектуры C4 (компонентный уровень).

2.3.2.1 Компонентный состав контейнера

Сервис доступа к файлам

Центральный компонент Файлового хранилища, реализованный на C# с использованием File System API:

- Предоставляет программный интерфейс для работы с файлами;
- Обрабатывает запросы на чтение, запись, обновление и удаление файлов;
- Обеспечивает взаимодействие с движком хранения и менеджером метаданных;
- Выполняет валидацию входящих запросов.

Менеджер метаданных

Компонент, отвечающий за управление информацией о файлах:

- Сохраняет метаданные файлов в базе данных (имя, тип, размер, дата создания);
- Связывает файлы с событиями и другими объектами системы через структуру базы данных;
- Обеспечивает получение метаданных при запросе файлов;
- Обновляет метаданные при изменении файлов.

Движок хранения

Низкоуровневый компонент, обеспечивающий физическое хранение файлов:

- Управляет физическим размещением файлов на дисковых накопителях;
- Выполняет чтение и запись файлов по запросу сервиса доступа;
- Обеспечивает основные операции с файловой системой;
- Поддерживает структуру каталогов для организации файлов.

2.3.2.2 Взаимодействие с другими контейнерами

Веб-портал

- Запрашивает файлы для отображения пользователям через Сервис доступа к файлам

- Загружает новые файлы, полученные от пользователей
- Получает метаданные файлов для отображения в интерфейсе

Подсистема обмена событиями (ПОС)

- Сохраняет файлы, связанные с новыми событиями через Сервис доступа к файлам
- Получает метаданные файлов для включения в информацию о событиях

Сервер задач

- Получает доступ к файлам для выполнения операций архивации
- Взаимодействует с Сервисом доступа к файлам для управления хранением
- Выполняет плановые задачи по обслуживанию файлового хранилища

База данных

- Хранит метаданные о файлах, предоставляемые Менеджером метаданных;
- Обеспечивает связь файлов с событиями и другими объектами системы.

2.3.2.3 Основные операции Файлового хранилища

Сохранение файла

a. Внешний контейнер передает файл в Сервис доступа к файлам.
b. Движок хранения сохраняет файл физически.
c. Менеджер метаданных сохраняет информацию о файле в базе данных.

Получение файла

a. Внешний контейнер запрашивает файл через Сервис доступа к файлам.
b. Менеджер метаданных предоставляет информацию о расположении файла.
c. Движок хранения считывает и предоставляет содержимое файла.

Файловое хранилище в системе КОЛОС реализовано как простая и надежная система для хранения документов, фотографий и видеоматериалов, связанных с событиями. Проверка прав доступа осуществляется на уровне Веб-портала через

общую ролевую модель, а операции архивации и управления жизненным циклом файлов реализованы в рамках Сервера задач.

2.3.3 Компоненты контейнера «Сервер задач»

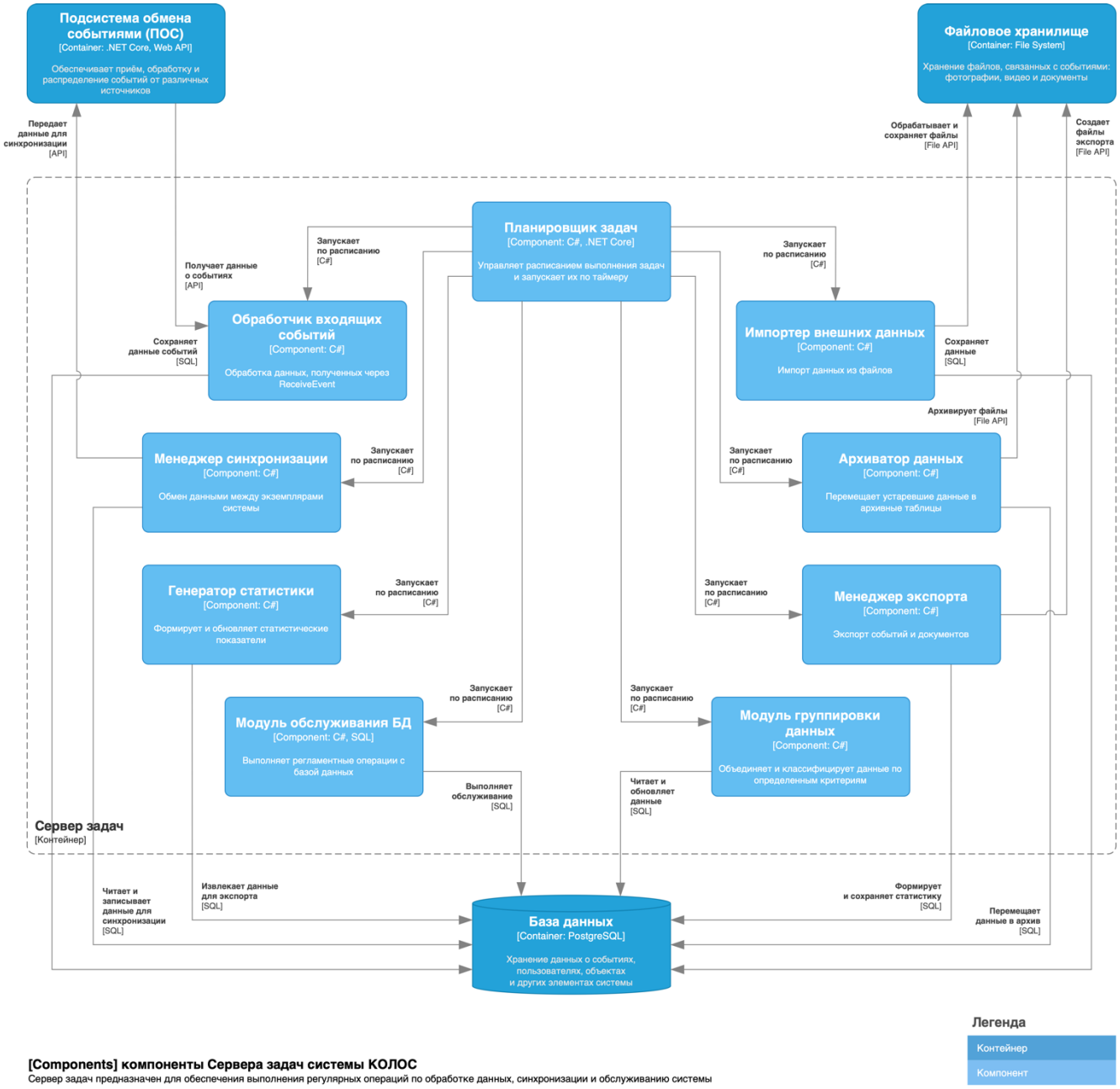


Рисунок 5 — уровень 3: «Сервер задач»

Сервер задач является ключевым контейнером системы «КОЛОС», обеспечивающим выполнение регулярных операций по обработке данных, синхронизации и обслуживанию системы. На уровне 3 архитектуры C4 (компонентный уровень) представлена детализация внутреннего устройства данного контейнера.

2.3.3.1 Компонентный состав контейнера

Планировщик задач

Центральный компонент Сервера задач, реализованный на C# и .NET Core:

- Управляет расписанием выполнения всех задач системы;
- Запускает задачи согласно настроенному расписанию;
- Контролирует выполнение задач и обрабатывает исключения;
- Обеспечивает последовательный запуск взаимозависимых задач.

Обработчик входящих событий

Компонент, отвечающий за обработку данных из внешних источников (ReceiveEventTask):

- Получает данные от ПОС о различных событиях (СКУД, погода, происшествия);
- Преобразует данные из внешних форматов в единый формат системы;
- Сохраняет обработанные события в базе данных;
- Осуществляет первичную валидацию входящих данных.

Импортер внешних данных

Компонент для импорта данных из файлов (ImportFilesDataTask, ImportFileListCheckTask):

- Импортирует данные из внешних источников в виде файлов;
- Проверяет целостность данных и наличие всех файлов в импортированных пакетах;
- Обновляет статусы обработанных пакетов;
- Сохраняет импортированные данные в базе данных и файловом хранилище.

Модуль группировки данных

Компонент для объединения и классификации данных (InitialDataGroupingTask, DuplicateIncidentTask):

- Объединяет данные из разных источников по одному объекту/субъекту;

- Выявляет связи между событиями и происшествиями;
- Объединяет разные сообщения об одном происшествии;
- Обеспечивает классификацию данных по определенным критериям.

Архиватор данных

Компонент для управления жизненным циклом данных (ArchiveEventsTask, DeleteLogsTask):

- Перемещает устаревшие данные из основных таблиц в архивные;
- Очищает старые записи в журнальных таблицах;
- Архивирует файлы, связанные с устаревшими событиями;
- Оптимизирует использование дискового пространства.

Генератор статистики

Компонент для формирования аналитических данных (StatisticsGenerationTask):

- Формирует и обновляет статистические показатели для различных типов событий;
- Заполняет аналитические кубы данных;
- Обеспечивает актуальность статистических данных;
- Подготавливает данные для аналитических отчетов.

Менеджер синхронизации

Компонент для обмена данными между экземплярами системы (EventsExchangeTask, EventsExchangeForwardTask):

- Подготавливает данные для отправки на другие экземпляры системы;
- Отправляет подготовленные пакеты данных;
- Обеспечивает целостность данных в распределенной инфраструктуре;
- Контролирует статусы отправленных пакетов.

Менеджер экспорта

Компонент для экспорта данных во внешние системы (OshExportTask):

- Подготавливает события и документы для экспорта;

- Формирует файлы экспорта в требуемых форматах;
- Создает файлы экспорта в файловом хранилище;
- Ведет учет экспортированных данных.

Модуль обслуживания БД

Компонент для обслуживания базы данных (UpdateDbStatistics):

- Выполняет обновление статистики базы данных;
- Осуществляет регламентные операции с базой данных;
- Оптимизирует работу базы данных;
- Повышает производительность запросов.

2.3.3.2 Взаимодействие с другими контейнерами

База данных

- Получает запросы на чтение и запись данных от всех компонентов Сервера задач;
- Предоставляет данные для анализа, группировки и формирования статистики;
- Хранит результаты обработки данных компонентами Сервера задач;
- Обеспечивает хранение метаданных о событиях, пользователях и объектах.

Файловое хранилище

- Принимает запросы на сохранение и получение файлов от Импортера внешних данных;
- Обеспечивает хранение файлов экспорта, создаваемых Менеджером экспорта;
- Взаимодействует с Архиватором данных для архивации файлов;
- Предоставляет доступ к файлам для всех компонентов Сервера задач.

Подсистема обмена событиями (ПОС)

- Предоставляет события Обработчику входящих событий;

- Получает данные для синхронизации от Менеджера синхронизации;
- Обеспечивает интеграцию с внешними системами;
- Служит транспортным механизмом для обмена событиями.

2.3.3.3 Основные операции Сервера задач

Обработка входящих событий

a. ПОС получает события из внешних источников.
b. Планировщик задач активирует Обработчик входящих событий.
c. Обработчик получает события от ПОС и преобразует их.
d. Преобразованные события сохраняются в базе данных.

Импорт данных из файлов

a. Планировщик задач запускает Импортер внешних данных.
b. Импортер обнаруживает новые файлы для импорта.
c. Данные из файлов преобразуются и сохраняются в базе данных.
d. Метаданные файлов регистрируются в системе.

Группировка и объединение данных

a. Планировщик задач запускает Модуль группировки данных.
b. Модуль выявляет связанные события и происшествия.
c. Происходит объединение дублирующих или уточняющих данных.
d. Обновленные данные сохраняются в базе данных.

Архивация данных

a. Планировщик задач запускает Архиватор данных.
b. Архиватор определяет устаревшие данные согласно настройкам.
c. Устаревшие данные перемещаются в архивные таблицы.
d. Журналы очищаются от старых записей.

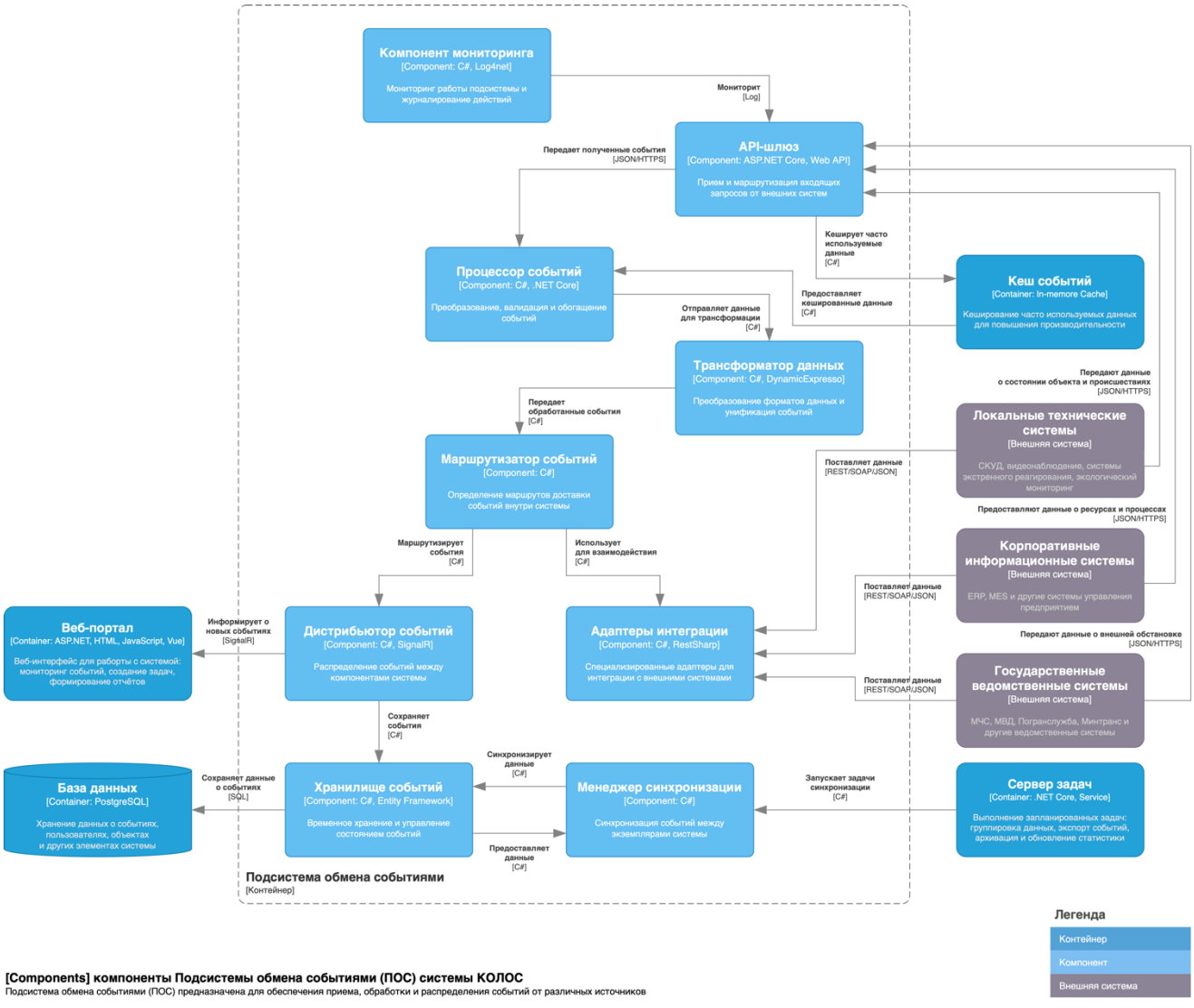
Формирование статистики

a. Планировщик задач запускает Генератор статистики.
b. Генератор собирает и анализирует данные из базы данных.
c. Формируются статистические показатели и заполняются кубы данных.

d. Результаты сохраняются для использования в отчетах.

Сервер задач в системе КОЛОС реализован как отказоустойчивый сервис с гибким расписанием выполнения задач. Он обеспечивает бесперебойную работу системы, своевременную обработку данных и эффективное управление информационными ресурсами. Все компоненты работают под управлением единого Планировщика задач, что позволяет координировать выполнение операций и оптимизировать использование системных ресурсов.

2.3.4 Компоненты контейнера «Подсистема обмена сообщениями (ПОС)»



[Components] компоненты Подсистемы обмена событиями (ПОС) системы КОЛОС
Подсистема обмена событиями (ПОС) предназначена для обеспечения приема, обработки и распределения событий от различных источников

Рисунок 6 — уровень 3: «Подсистема обмена событиями»

Подсистема обмена событиями (ПОС) является критически важным контейнером системы «КОЛОС», обеспечивающим прием, обработку и распределение событий от различных источников. На уровне 3 архитектуры C4

(компонентный уровень) представлена детализация внутреннего устройства данного контейнера.

2.3.4.1 Компонентный состав контейнера

API-шлюз

Компонент, реализующий внешний программный интерфейс подсистемы:

- Принимает входящие запросы от внешних систем через метод ReceiveEvent;
- Маршрутизирует запросы к нужным компонентам обработки;
- Обеспечивает аутентификацию и авторизацию запросов;
- Реализует REST API для взаимодействия с другими контейнерами.

Процессор событий

Центральный компонент обработки входящих событий:

- Осуществляет первичную валидацию полученных событий;
- Выполняет преобразование и обогащение событий дополнительными данными;
- Классифицирует события по типам и категориям;
- Определяет приоритеты обработки событий.

Маршрутизатор событий

Компонент, определяющий дальнейший маршрут обработки событий:

- Анализирует события и определяет их маршруты доставки;
- Направляет события соответствующим получателям внутри системы;
- Обеспечивает настраиваемую логику маршрутизации;
- Поддерживает фильтрацию событий.

Трансформатор данных

Компонент для унификации форматов данных:

- Преобразует входящие данные из различных форматов в единый формат системы;

- Использует библиотеку DynamicExpresso для обработки данных формата JSON;
- Обеспечивает совместимость данных между различными компонентами.

Адаптеры интеграции

Набор специализированных компонентов для интеграции с внешними системами:

- Реализуют специфические протоколы взаимодействия с внешними системами;
- Обеспечивают получение данных из различных источников (HTTP/S, файловые ресурсы, СУБД);
- Поддерживают возможность подключения к практически любой системе, имеющей API;
- Используют библиотеку RestSharp для взаимодействия с REST API.

Дистрибьютор событий

Компонент для распределения событий между потребителями:

- Обеспечивает доставку событий всем заинтересованным компонентам;
- Реализует механизмы асинхронного оповещения о событиях;
- Использует технологию SignalR для передачи событий в реальном времени;
- Поддерживает различные стратегии доставки событий.

Хранилище событий

Компонент для временного хранения и управления состоянием событий:

- Обеспечивает временное сохранение событий в процессе обработки;
- Управляет состоянием событий (новые, обрабатываемые, обработанные);
- Взаимодействует с базой данных для долговременного хранения;
- Использует Entity Framework для работы с данными.

Менеджер синхронизации

Компонент для синхронизации данных между экземплярами системы:

- Обеспечивает обмен данными между различными экземплярами ПОС;
- Подготавливает и передает пакеты данных для синхронизации;
- Обрабатывает входящие пакеты синхронизации;
- Поддерживает работу в распределенной топологии (многоуровневый граф без циклов).

Компонент мониторинга

Компонент для наблюдения за работой подсистемы:

- Отслеживает работоспособность всех компонентов ПОС;
- Ведет журналирование всех действий и ошибок;
- Собирает метрики производительности;
- Использует библиотеку Log4net для логирования.

2.3.4.2 2. Взаимодействие с другими контейнерами

Веб-портал

- Получает от ПОС информацию о новых событиях в реальном времени;
- Отправляет запросы на получение данных через API-шлюз;
- Использует SignalR для получения уведомлений о событиях;
- Отображает события, полученные от ПОС, пользователям системы.

Сервер задач

- Запускает регулярные задачи по синхронизации данных;
- Обрабатывает данные, полученные через метод ReceiveEvent;
- Взаимодействует с Менеджером синхронизации для обмена данными;
- Выполняет группировку и анализ событий, полученных от ПОС.

База данных

- Хранит данные о событиях, поступающих через ПОС;
- Обеспечивает долговременное хранение событий;
- Предоставляет данные для обработки и анализа;
- Поддерживает историю изменений событий.

Кеш событий

- Кеширует часто используемые данные для повышения производительности;
- Обеспечивает быстрый доступ к актуальным данным;
- Синхронизируется с Хранилищем событий;
- Использует in-memory кеширование для оптимизации доступа.

Внешние системы

- Отправляют данные в ПОС через API-шлюз в формате JSON;
- Получают данные через специализированные адаптеры интеграции;
- Являются источниками событий для системы КОЛОС;
- Включают СКУД, системы погоды, МЧС, МВД и другие источники.

2.3.4.3 Основные операции Подсистемы обмена событиями

Получение события от внешней системы

a. Внешняя система отправляет данные через API-шлюз.
b. API-шлюз проверяет аутентификацию и авторизацию запроса.
c. Процессор событий валидирует и обрабатывает событие.
d. Трансформатор данных преобразует событие в единый формат системы.
e. Маршрутизатор событий определяет получателей события.
f. Дистрибьютор событий доставляет событие всем заинтересованным компонентам.
g. Хранилище событий сохраняет событие для последующей обработки.

Синхронизация данных между экземплярами

a. Сервер задач запускает задачу синхронизации.
b. Менеджер синхронизации запрашивает данные из Хранилища событий.
c. Формируются пакеты данных для синхронизации.
d. Пакеты передаются на другие экземпляры ПОС.
e. На принимающей стороне Менеджер синхронизации обрабатывает входящие пакеты.

f. Данные сохраняются в локальном Хранилище событий.
g. Обновленные данные становятся доступны всем компонентам системы.

Оповещение о новом событии

a. В ПОС поступает новое событие.
b. Событие обрабатывается и сохраняется в Хранилище событий.
c. Дистрибьютор событий определяет, что о событии необходимо уведомить Веб-портал.
d. Через SignalR отправляется уведомление о новом событии.
e. Веб-портал получает уведомление и обновляет интерфейс пользователя.
f. Пользователь видит новое событие в реальном времени.

Подсистема обмена событиями в системе КОЛОС построена на основе современных технологий и обеспечивает надежную обработку и распределение событий с высокой производительностью. Система способна обрабатывать до 8 млн событий в сутки с пиковой нагрузкой до 1,3 млн событий в час, что отвечает требованиям работы в условиях высоких нагрузок при проведении массовых мероприятий и в промышленных условиях.

2.3.5 Компоненты контейнера «Кеш событий»

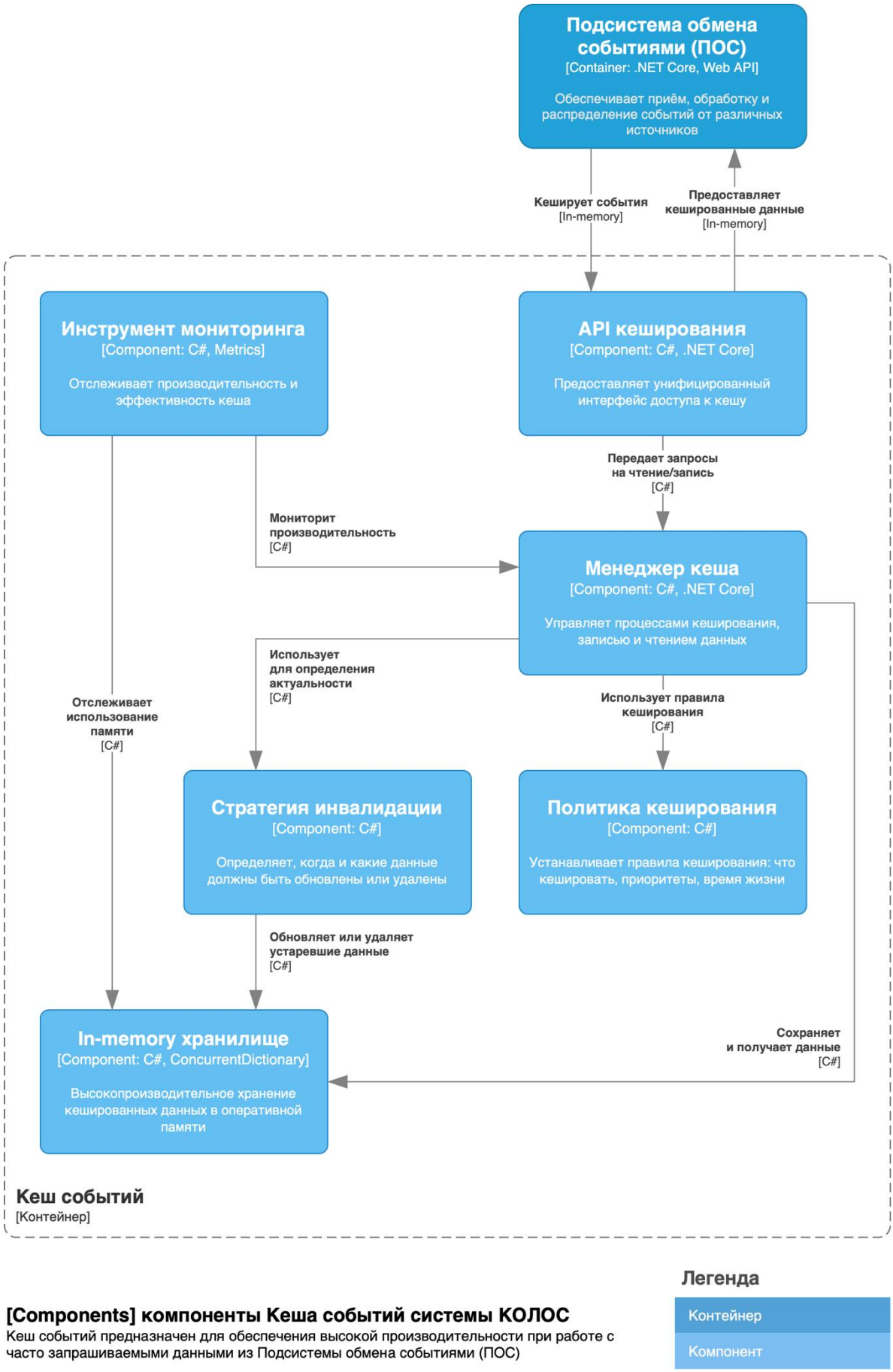


Рисунок 7 — уровень 3: «Кеш событий»

Кеш событий является важным контейнером системы «КОЛОС», обеспечивающим высокую производительность при работе с часто запрашиваемыми данными. На уровне 3 архитектуры C4 (компонентный уровень) представлена детализация внутреннего устройства данного контейнера в строгом соответствии с диаграммой контейнеров (уровень L2).

2.3.5.1 Компонентный состав контейнера

Менеджер кеша

Центральный компонент, координирующий все процессы кеширования:

- Управляет операциями записи и чтения кешированных данных;
- Принимает решения о кешировании на основе политик и стратегий;
- Координирует работу других компонентов кеша;
- Обеспечивает целостность и актуальность кешированных данных.

In-memory хранилище

Компонент, реализующий физическое хранение кешированных данных:

- Использует структуры данных в оперативной памяти для максимальной скорости доступа;
- Реализован на базе ConcurrentDictionary для потокобезопасного доступа;
- Оптимизирован для быстрого поиска и извлечения данных;
- Обеспечивает эффективное использование оперативной памяти.

Стратегия инвалидации

Компонент, отвечающий за поддержание актуальности кешированных данных:

- Определяет, когда данные в кеше становятся устаревшими;
- Реализует различные алгоритмы инвалидации (TTL, LRU, LFU);
- Инициирует обновление или удаление неактуальных данных;
- Обеспечивает баланс между актуальностью и производительностью.

Политика кеширования

Компонент, определяющий правила кеширования данных:

- Устанавливает критерии того, какие данные следует кешировать;
- Определяет время жизни различных типов данных в кеше;
- Устанавливает приоритеты при ограниченных ресурсах памяти;
- Адаптируется к изменениям паттернов доступа к данным.

Инструмент мониторинга

Компонент для наблюдения за работой кеша:

- Отслеживает ключевые метрики производительности кеша (hit ratio, latency);
- Мониторит использование оперативной памяти;
- Собирает статистику использования кеша для оптимизации;
- Интегрируется с общей системой мониторинга КОЛОС.

API кеширования

Компонент, предоставляющий унифицированный интерфейс к кешу:

- Реализует методы для записи, чтения и удаления данных из кеша;
- Абстрагирует клиентов от деталей реализации кеша;
- Обеспечивает типобезопасный доступ к кешированным данным;
- Поддерживает асинхронные операции для повышения эффективности.

2.3.5.2 Взаимодействие с другими контейнерами

Подсистема обмена событиями (ПОС)

- Единственный контейнер, взаимодействующий с Кешем событий согласно диаграмме контейнеров (L2);
- Кеширует часто используемые события для ускорения их обработки;
- Получает кешированные данные для повышения производительности обработки событий;
- Использует кеш как высокоскоростную промежуточную память для оптимизации работы.

2.3.5.3 Основные операции Кеша событий

Чтение данных из кеша

a. ПОС обращается к API кеширования с запросом данных.
b. API кеширования передает запрос Менеджеру кеша.
c. Менеджер кеша проверяет наличие данных в In-memory хранилище.
d. Если данные найдены и актуальны (проверяется Стратегией инвалидации), они возвращаются ПОС.
e. Если данные отсутствуют или устарели, ПОС получает информацию из основного источника данных.

Запись данных в кеш

a. ПОС передает данные для кеширования через API кеширования.
b. API кеширования направляет запрос Менеджеру кеша.
c. Менеджер кеша, используя Политику кеширования, определяет, следует ли кешировать эти данные.
d. Если кеширование необходимо, данные сохраняются в In-memory хранилище.
e. Инструмент мониторинга регистрирует операцию кеширования для сбора статистики.

Инвалидация кеша

a. Стратегия инвалидации периодически проверяет актуальность данных в кеше.
b. При обнаружении устаревших данных они помечаются для удаления.
c. Менеджер кеша очищает неактуальные данные из In-memory хранилища.
d. ПОС получает сообщение об инвалидации кеша (опционально).
e. Инструмент мониторинга регистрирует события инвалидации для анализа эффективности кеширования.

Кеш событий в системе КОЛОС спроектирован как высокопроизводительное in-memory хранилище, обеспечивающее значительное повышение скорости обработки событий для Подсистемы обмена событиями. Использование кеша

особенно критично в ситуациях с высокой нагрузкой, например, при проведении массовых мероприятий или при одновременной обработке большого количества событий из различных источников.

2.4 Архитектура данных системы

Верхнеуровневая ER-диаграмма представлена на рисунке ниже:

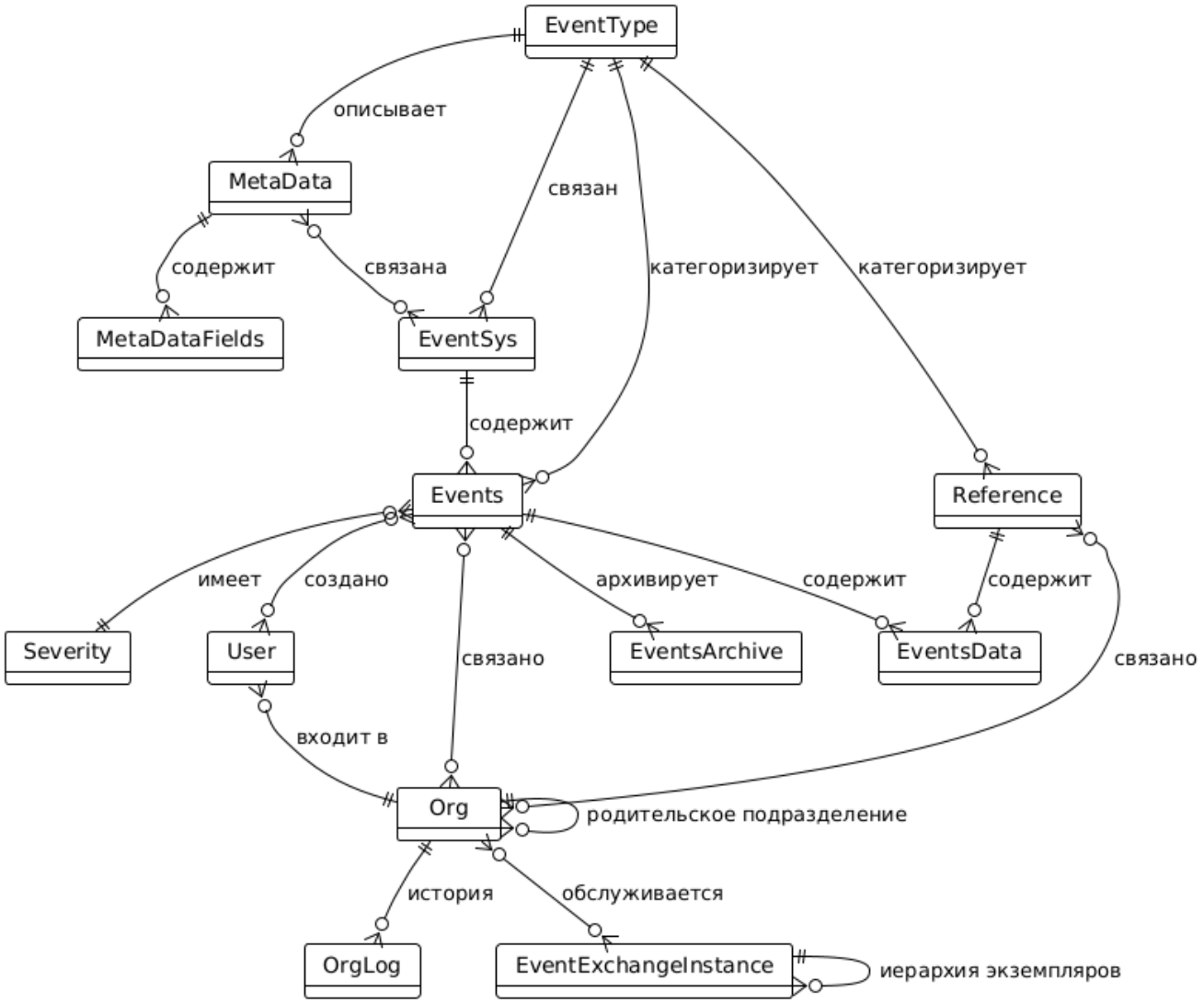


Рисунок 8 — верхнеуровневая ER-диаграмма

Описание данных, представленных на ER-диаграмме:

a. EventType — слои данных: данные о происшествиях, нормативно-справочная информация и т.п.
b. MetaData — метаданные. Описывают где и как данные по каждому слою хранятся.

c. MetaDataFields — поля метаданных. Содержат информацию по хранению атрибутов и выполнению запросов к отдельным атрибутам каждого слоя данных.
d. Events — хранит данные об «устаревающих» событиях: происшествия, прогноз погоды, события, лог удаления записей.
e. Reference — хранит неустаревающие и относительно редко обновляемые данные: сводки, документы, объекты, мероприятия, типы событий, внешние данные и т.п.
f. EventsData — файлы, привязанные к событиям. При этом файлы также могут хранится в папке общего доступа.
g. EventsArchive — архив событий. События, «срок жизни» которых прошел, перемещаются сюда, чтобы уменьшить размер таблицы Events.
h. Severity — критичность события (критичное, важное, информационное).
i. User — набор таблиц, описывающих пользователя системы (реализовано на основе технологии ASP.NET Membership). Необязательная связь с References\Events указывает автора события (кто внес в систему).
j. Org — подразделение, в котором работает\служит пользователь. Связь (необязательная) с References\Events введена для ускорения некоторых запросов (в том числе по разграничению доступа – кому какие данные показывать). Org образует иерархию (связь на себя – это родительское подразделение).
k. OrgLog — история изменения данных о подразделении.
l. EventsExchangeInstance — экземпляры ПОС (лицензируемое ПО «Событикус»). Связь с организацией показывает, каким экземпляром какое подразделение обслуживается (где хранятся данные, получаемые от сотрудников подразделения).

В Системе используется небольшое количество таблиц, основная часть которых хранит только справочную информацию и информацию для настройки системы.

Все объекты Системы (документы, сводки, события, происшествия и т. п.) хранятся в трех таблицах БД: «EventsData», «Events» и «Reference». Однотипные данные (координаты, сведения об изменениях и внешне ссылки) сохраняются в отдельных полях таблицы, а основные и отличные атрибуты объектов хранятся в едином поле в формате JSON.

Данная реализация была выбрана для решения одних из ключевых задач проекта: обеспечения масштабируемости системы — возможности подключения новых типов данных без внесения изменений в базу данных (потребуется только указать новый тип данных в MetaData).

2.4.1 Потоки данных

Система «КОЛОС» организует сложное взаимодействие между различными компонентами для обеспечения эффективного обмена информацией. В центре этого взаимодействия находится Подсистема обмена событиями (ПОС).

Диаграмма компонентов (в нотации UML), на которой отражены потоки данных, представлена на рисунке ниже:

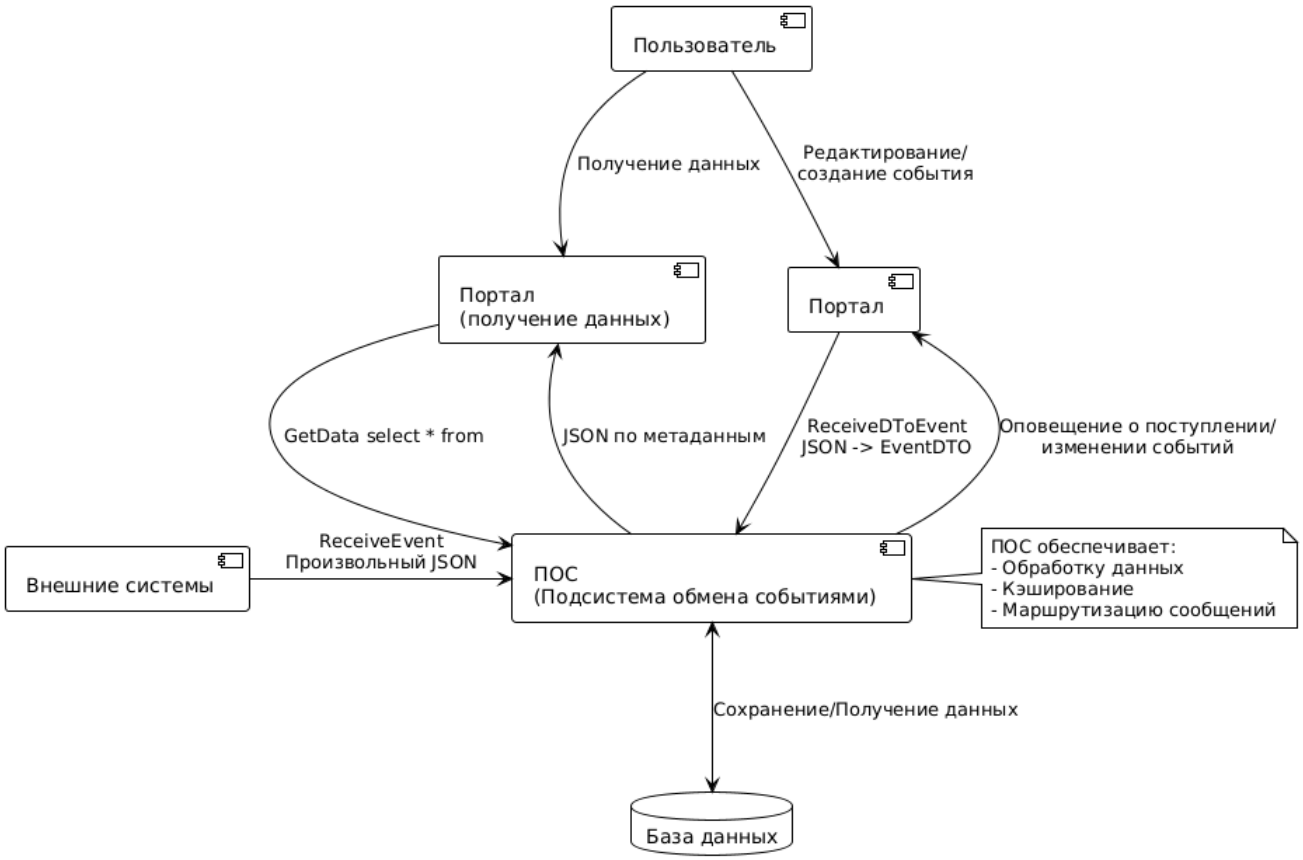


Рисунок 9 — диаграмма компонентов с потоками данных

Основные потоки данных в системе осуществляются следующим образом:

a. Ввод и редактирование данных:
   - Пользователь взаимодействует с порталом — создает или редактирует события, загружает сводки, документы, изменяет карточки объектов.
   - Портал передает новые или измененные данные в ПОС с помощью метода ReceiveDToEvent, который преобразует данные из формата JSON в объекты EventDTO.
   - ПОС обрабатывает полученные данные и информирует портал (все узлы, подключенные в рамках кластера) о поступлении/изменении событий.
b. Получение данных:
   - При запросе пользователя, портал обращается к ПОС с запросом GetData select * from.
   - ПОС возвращает данные в формате JSON, структурированные согласно метаданным.
   - Если необходимые данные содержатся в кэше ПОС, они возвращаются без обращения к базе данных, что ускоряет работу системы.
c. Интеграция с внешними системами:
   - Внешние системы передают данные в ПОС с помощью метода ReceiveEvent в произвольном формате JSON.
   - ПОС обеспечивает стандартизацию и обработку этих данных для последующего использования в системе.
d. Взаимодействие с базой данных:
   - ПОС осуществляет двунаправленное взаимодействие с базой данных, сохраняя и получая необходимую информацию.
   - Сервер задач по расписанию выполняет различные операции с базой данных, включая группировку, архивацию и обновление статистики.

2.4.2 Загрузка данных из внешних систем

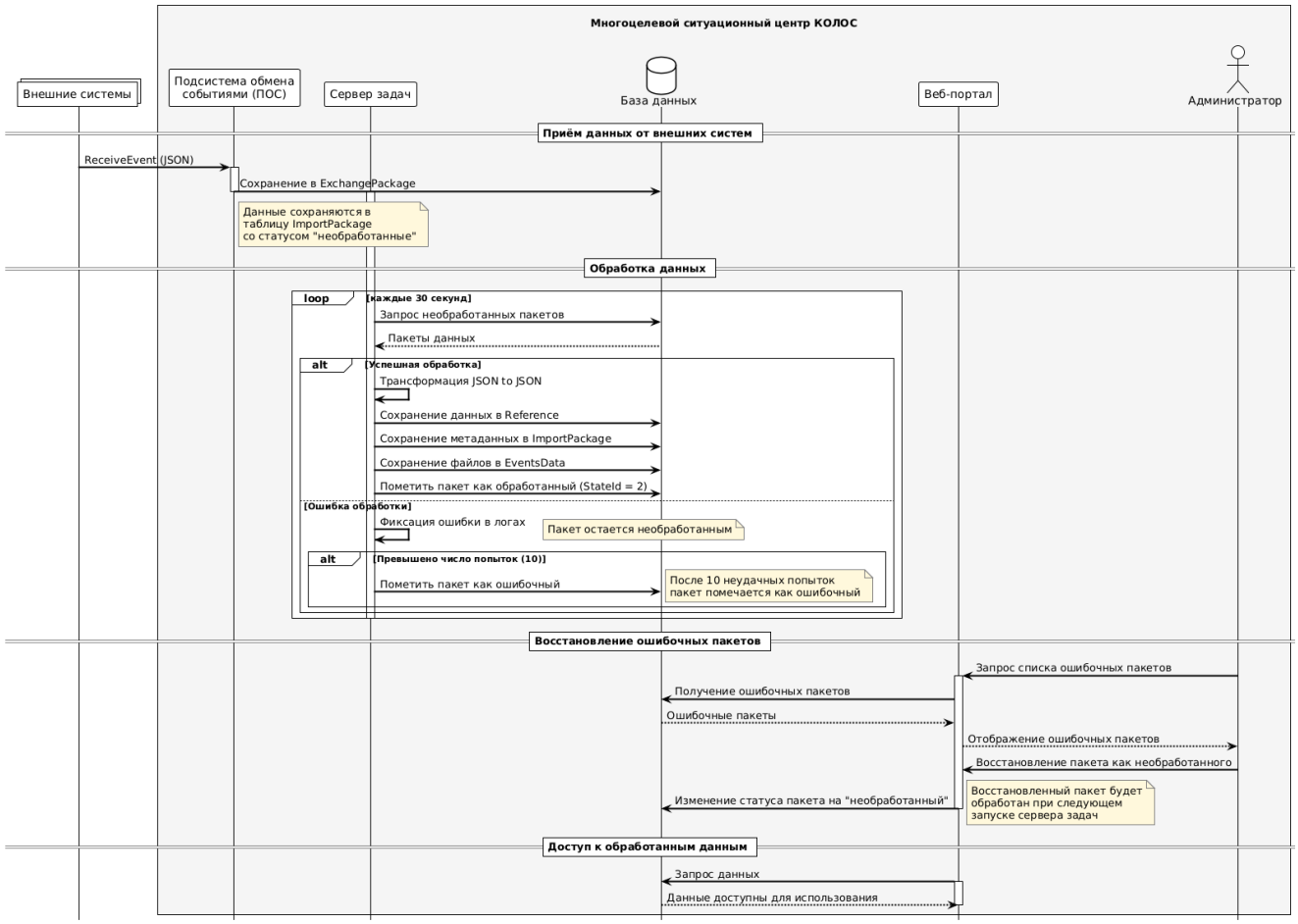


Рисунок 10 — схема загрузки данных из внешних систем

Загрузка данных из внешних систем осуществляется по следующему алгоритму:

a. Получение данных:
   - Внешние системы передают данные через метод ReceiveEvent.
   - Полученные данные преобразуются в формат ExchangePackage и записываются в таблицу ImportPackage.

b. Обработка данных:
   - Сервер задач каждые 30 секунд проверяет наличие необработанных пакетов в таблице ImportPackage.
   - Для каждого необработанного пакета выполняется трансформация JSON-данных.
   - Результаты трансформации параллельно сохраняются:

        - В таблицу Reference — основные данные в формате JSON BasePoint.
        - В таблицу ImportPackage — метаданные о группировке записей и идентификаторах.
        - В таблицу EventsData — файлы и прикрепленные документы.
    - После успешной обработки пакет помечается как обработанный (StateId = 2).
c. Обработка ошибок:
    - В случае ошибки при трансформации или записи данных информация фиксируется в логах.
    - Пакет не отмечается как обработанный и будет обработан при следующем запуске задачи.
    - Для каждого пакета предусмотрено до 10 попыток обработки (по одной на каждый запуск задачи).
    - По достижении лимита неудачных попыток пакет отмечается как ошибочный.
    - Пользователи с ролью «Администратор» имеют возможность восстановить ошибочные пакеты как необработанные через интерфейс системы.

2.4.3 Трансформация данных

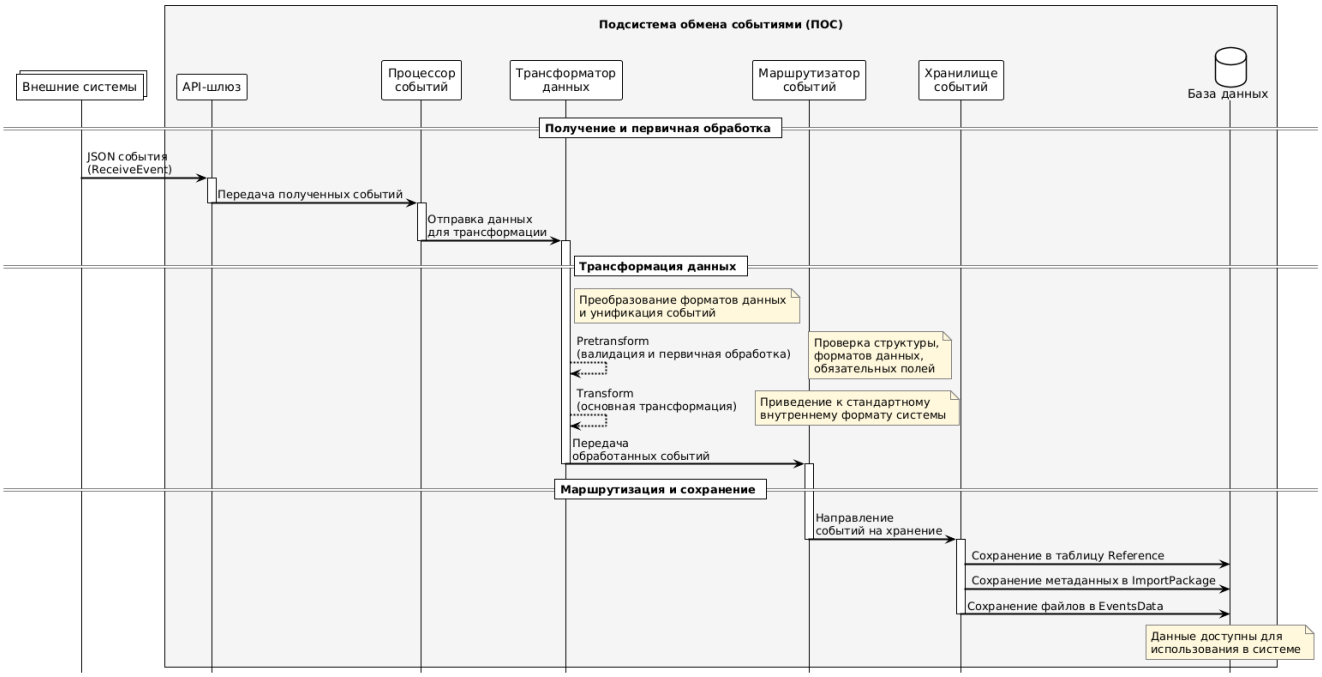


Рисунок 11 — схема трансформации данных

Система «КОЛОС» использует многоэтапный процесс трансформации данных для обеспечения их совместимости и использования в различных компонентах:

a. Предварительная обработка:
   - Входящие JSON-данные проходят этап предварительной трансформации (Pretransform), где осуществляется валидация и первичная обработка.
   - Затем выполняется основная трансформация (Transform), приводящая данные к стандартному формату.

b. Параллельная обработка:
   - Трансформированные данные направляются по двум основным путям:
     - Преобразование в формат JSON BasePoint и сохранение в таблице ImportPackage для последующего использования.
     - Сохранение в системе (Save) с дополнительной обработкой для статистики и сопоставления с запросами пользователей.

c. Конечное размещение:
   - После всех этапов обработки данные сохраняются в соответствующих таблицах базы данных, становясь доступными для использования в системе.

Эта многоуровневая схема трансформации обеспечивает гибкость системы и возможность работы с разнородными источниками данных при сохранении единого формата внутри системы.