ООО «Р2СПРО»

ОПИСАНИЕ

АРХИТЕКТУРЫ ПРОГРАММНОГО КОМПЛЕКСА «R2S PROWORK» («ПРОВОРК»)

г. Нижний Новгород

«12» августа 2025 г.

Введение

Настоящий документ описывает высокоуровневую архитектуру программного комплекса «R2S ProWork» («ПроВорк»). Решение основано на принципах микросервисной архитектуры и предназначено для построения масштабируемых, отказоустойчивых и легко поддерживаемых систем автоматизации бизнес-процессов предприятия.

Документ предназначен для технических специалистов: архитекторов, разработчиков, системных администраторов, а также менеджеров проекта.

1 Общая архитектурная диаграмма (Высокоуровневое представление)

Общая архитектурная диаграмма представлена на рисунке 1.1.

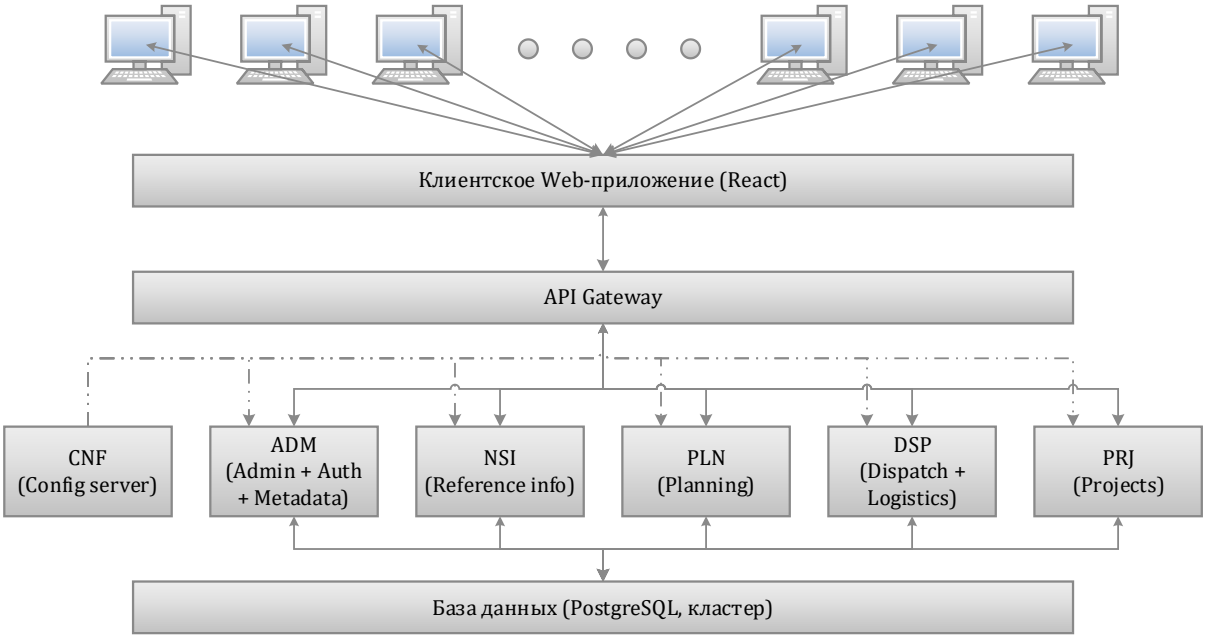

Рисунок 1.1. Общая архитектурная диаграмма

2 Принципы архитектуры

Микросервисная архитектура: система разбита на набор небольших, слабосвязанных сервисов, каждый из которых решает свою бизнес-задачу и может разрабатываться, развертываться и масштабироваться независимо.

RESTful API: сервисы взаимодействуют между собой и с клиентским приложением по протоколу HTTP, используя REST API.

Единая точка входа: запросы от клиентского приложения к бэкенду проходят через API Gateway, который отвечает за маршрутизацию, аутентификацию и балансировку нагрузки.

Централизованное управление конфигурацией: все сервисы получают настройки из единого Config Server (CNF).

Самостоятельность сервисов: каждый сервис владеет своими данными и инкапсулирует свою бизнес-логику.

3 Описание ключевых компонентов

3.1 Клиентский фронтенд (Web-интерфейс)

Технологии: React, Axios.

Назначение: предоставляет пользовательский интерфейс для работы со всеми модулями системы. Не содержит бизнес-логики, вся логика вынесена на бэкенд.

3.2 Серверные микросервисы (Backend)

Все сервисы реализованы на основе Spring Boot.

CNF (Config Server)

Назначение: централизованное хранение и управление конфигурациями всех сервисов. Поддерживает различные профили (dev, prod) и работу с секретами.

Технологии: Spring Cloud Config Server.

ADM (Сервис администрирования)

Назначение: управление пользователями, ролями, правами доступа. Содержит метаданные интерфейсов. Включает в себя сервис лицензирования.

Технологии: Spring Security, JWT.

NSI (Сервис нормативно-справочной информации)

Назначение: хранение и управление всей статической и условно-статической информацией предприятия (справочники, номенклатура, технологические операции, контрагенты и т.д.). Является основным источником данных для других сервисов.

Технологии: Spring Data JPA (Hibernate), Liquibase.

PLN (Сервис планирования)

Назначение: реализация алгоритмов планирования по методологиям MRP-II, APS, SCM. Формирует плановые задания на производство и закупку.

Технологии: Spring Data JPA (Hibernate), Liquibase.

DSP (Сервис диспетчеризации и логистики)

Назначение: оперативное управление производственными заказами, логистикой, складским хозяйством. Отслеживание исполнения планов.

Технологии: Spring Data JPA (Hibernate).

PRJ (Сервис управления проектами)

Назначение: управление проектной деятельностью организации.

Технологии: Spring Data JPA (Hibernate).

3.3 База данных

Технология: PostgreSQL. Возможно подключение любой существующей релятивной СУБД.

Архитектура: каждый сервис использует собственную схему базы данных (Schema-per-Service), что обеспечивает изоляцию данных.

3.4 API Gateway

Назначение: маршрутизация запросов к соответствующим сервисам, аутентификация, авторизация, кэширование, ограничение частоты запросов (Rate Limiting).

Технологии: Nginx/Synginx. Возможна настройка через Apache2 (mod_proxy и mod_rewrite).

4 Взаимодействие между компонентами

Пользователь работает с интерфейсом React.

React-приложение отправляет HTTP-запросы к API Gateway.

API Gateway, основываясь на пути запроса (/api/nsi/**, /api/adm/**), перенаправляет его соответствующему микросервису.

Микросервис обрабатывает запрос, взаимодействуя при необходимости со своей базой данных или другими сервисами (например, DSP запрашивает данные о номенклатуре у NSI).

Сервис возвращает ответ через API Gateway обратно в клиентское приложение.

5 Заключение

Представленная микросервисная архитектура обеспечивает высокую масштабируемость, отказоустойчивость и гибкость комплекса «R2S ProWork». Разделение на отдельные сервисы позволяет независимо развивать функциональность, назначать разные команды и применять оптимальные технологические решения для каждой конкретной бизнес-задачи.