Платформа Rightech IoT Core

Rightech IoT Core (RIC) фреймворк-инструмент разработчиков для создания, развертывания и управления решениями в области IoT. RIC не зависит от конкретного оборудования и протоколов. Поэтому легко объединять разные устройства под одним решением. Основной упор направлен на упрощение процесса разработки и внедрения IoT-решений. Сервисы платформы позволяет управлять устройствами, собирать и анализировать данные, обеспечивать безопасность, интеграцию с различными ИС и прочие механизмы, необходимые для создания комплексных IoT-систем.

Ключевые свойства платформы

Мультитенантность и изоляция проектов

Каждый проект работает в собственном изолированном пространстве с независимыми пользователями, устройствами, настройками и правами доступа.

Это позволяет:

- Нескольким клиентам или подразделениям использовать одну инстанцию платформы
- Системным интеграторам запускать проекты для разных заказчиков
- Командам разработчиков работать над несколькими проектами одновременно

Дискреционная система разграничения прав

Принцип "запрещено всё, что не разрешено явно" (all-deny по умолчанию).

- Гибкая настройка прав для каждой роли — от администратора до оператора с доступом только на чтение
- Логирование всех действий пользователя

- Контроль каждого доступа к данным
- Соответствие корпоративным стандартам безопасности

Масштабируемость и отказоустойчивость

Автоматическое масштабирование в зависимости от количества устройств, пользователей и нагрузки.

- Нет необходимости вручную добавлять серверы при росте проекта
- Начинать можно с прототипа и масштабировать до тысяч устройств и миллионов пользователей без остановки сервиса
- Инфраструктура адаптируется под текущие потребности

Kubernetes-ready и SOA-архитектура

Платформа развёрнута в контейнеризированной среде на базе Kubernetes:

- Автоматическое масштабирование сервисов
- Высокая доступность и отказоустойчивость
- Простое развертывание обновлений
- Переносимость между облачными провайдерами
- Соответствие современным стандартам DevOps

Сервис-ориентированная архитектура (SOA):

- Каждый сервис выполняет строго определённый набор функций
- Внутреннее взаимодействие через gRPC
- Гибкая настройка контрактов между сервисами

Платформа готова к работе в гибридных и мультиоблачных средах, поддерживает on-premise установку для организаций с высокими требованиями к безопасности данных.

AI-ready: фундамент для машинного обучения

Платформа предоставляет готовую базу для внедрения AI и ML:

- **Структурированные данные** — вся телеметрия хранится в единой модели с контекстной привязкой

- **Контекстная информация** — данные связаны с объектами, временем, местоположением
- **Потоки событий** — возможность real-time анализа в режиме реального времени
- **Интеграция с ML-системами** — API для подключения внешних AI-сервисов
- **Автоматизация на основе AI** — сценарии могут реагировать на результаты AI-анализа

Это позволяет внедрять предиктивную аналитику, обнаружение аномалий, оптимизацию процессов без изменения архитектуры IoT-системы.

Безопасность

- **Двухфакторная аутентификация** — дополнительная защита аккаунтов
- **X.509 сертификаты** — защита соединений на уровне устройств
- TLS шифрование — все данные передаются в зашифрованном виде
- **Ролевая модель доступа** — гибкая настройка прав для разных типов пользователей
- **Изоляция проектов** — данные одного проекта недоступны пользователям другого

Реализованные принципы в рамках архитектуры

Открытые стандарты и протоколы. При разработке платформы использовались исключительно открытые стандарты и протоколы для обеспечения интероперабельности между сервисами и системами.
API-ориентированность. Внутренние сервисы предоставляют внутренний API для взаимодействия с другими сервисами. Внешний RESTful API в специализированном сервисе позволяет взаимодействовать внешним приложениям и сервисам. API

качественно определенно и задокументировано, что облегчает как внутреннюю разработку так и интеграции.
Безопасность. Обеспечена безопасность данных и коммуникации между сервисами, с учетом особенности облачных сред.
Легкость развертывания и управления. Использование средств автоматизации развертывания, управления конфигурацией и мониторинга значительно повышают качество жизненного цикла как всей платформы Rightech IoT Core, так и отдельных ее сервисов.

Основная шина данных платформы Rightech IoT Core

В качестве внутренней шины данных платформы Rightech IoT Core используется специализированное открытое ПО - RabbitMQ.

RabbitMQ

RabbitMQ - мощный брокер сообщений, который часто используется в качестве шины данных (message bus) в распределенных информационных системах. Сама по себе шина данных представляет собой некий механизм передачи сообщений между различными компонентами системы, обеспечивая быстрое асинхронное взаимодействие и обмен данными между ними. Описанные ниже характеристики и особенности позволили нам использовать RabbitMQ в полной мере и обеспечить все необходимые возможности:

- Асинхронность.
- Гибкость и масштабируемость.
- Надежность.
- Протокол AMQP (Advanced Message Queuing Protocol).
- Обработка различных типов сообщений.
- Управление очередями и обменами.

Внутренний интерфейс взаимодействия сервисов Rightech IoT Core

В качестве внутренней API взаимодействия сервисов платформы используется современная, мощная и открытая технология - gRPC.

gRPC

gRPC (gRPC Remote Procedure Call) - высокопроизводительный и универсальный фреймворк для разработки удаленных процедурных вызовов (RPC). Используется в платформе Rightech IoT Core в качестве мощного инструмента реализации внутренних интерфейсов взаимодействия между сервисами. Далее представлены ключевые аспекты данной технологии, которые и позволили его использовать в полной мере.

Простота описания контрактов и сервисов

Protocol Buffers (ProtoBuf): gRPC использует Protocol Buffers для определения структуры данных и интерфейсов сервисов, что крайне удобно и полностью подходит в рамках архитектуры платформы Rightech IoT Core. ProtoBuf является языко-независимым механизмом сериализации данных, что упрощает определение и обмен данными между сервисами.
Определение сервисов и методов в gRPC осуществляется с использованием языконезависимого синтаксиса, что делает формирование интерфейсов простым и читаемым.

Мультиязычность

В рамках архитектуры платформы существует свобода в выборе языков программирования. В различных внутренних сервисах используются функционально и технически “уместные” языки программирования. gRPC обеспечивает поддержку нескольких

языков программирования (Java Script, Python, Go, C++, C#, и другие).

Бинарная сериализация и эффективность:

gRPC использует бинарный формат сериализации данных, что обеспечивает компактность и эффективность передачи данных внутри сети. gRPC основан на протоколе HTTP/2, что способствует уменьшению задержек и обеспечивает возможность многоканального (multiplexing) взаимодействия, улучшая эффективность сетевого взаимодействия.

Автоматическая Генерация Кода

gRPC обеспечивает автоматическую генерацию клиентского и серверного кода на основе определений ProtoBuf, что устраняет необходимость вручную создавать код нашим разработчикам для обработки данных и взаимодействия.

Современные механизмы обнаружения ошибок и безопасность

gRPC предоставляет механизмы для обнаружения ошибок во время выполнения, что способствует отслеживанию состояния и диагностике проблем. Позволяет осуществлять интеграцию с различными механизмами безопасности, такими как SSL/TLS для шифрования данных и аутентификации.

Поддержка синхронных и асинхронных взаимодействий между сервисами

gRPC поддерживает как синхронные, так и асинхронные вызовы, что обеспечивает гибкость в разработке и развитии платформы с разными требованиями к взаимодействию сервисов.

Внешний программный интерфейс Rightech IoT Core

В качестве внешнего программного интерфейса реализованы принципы так называемого RESTful API, предложенного ученым Жаком Ройтфельдом, специализирующимся на IP и HTTP в частности. Подробнее с документацией внешнего RESTful API можно ознакомиться по ссылке из названия раздела (https://rightech.io/ru/developers/http).

RESTful API

RESTful API (Representational State Transfer) архитектурный стиль для проектирования внешних программных интерфейсов (API), который основан на принципах определения всех доступных пользователю ресурсов и способах взаимодействия с ними.

Ресурсы и URI (Uniform Resource Identifier)

Ресурсы. RESTful API ориентирован на работу с ресурсами, которые представляют собой сущности или данные. Например, в рамках платформы Rightech IoT Core - пользователи, роли, модели, объекты, обработчики, автоматы и т.д.

URI. Каждый ресурс идентифицируется уникальным URI. URI представляет собой адрес ресурса, который обеспечивает единообразие в обращении к данным.

Операции CRUD (Create, Read, Update, Delete)

Операции. RESTful API поддерживает четыре основные операции: создание (POST), чтение (GET), обновление (PUT/PATCH) и удаление (DELETE). Эти операции соответствуют стандартным CRUD-операциям.

Семантика Операций. Операции имеют семантику, соответствующую действиям с ресурсами: создание, получение, обновление или удаление.

Формат представление ресурсов

Форматы передачи данных. RESTful API может обмениваться данными в различных форматах, таких как JSON, XML, HTML и другие. В платформе Rightech IoT Core используется JSON как наиболее востребованный на текущий момент при построении межсервисного взаимодействия.

Stateless и доступ

Отсутствие состояния. RESTful API не хранит состояние клиента на сервере между запросами. Каждый запрос клиента содержит все необходимые данные для выполнения операции.

Сессии и токены. Аутентификация и управление сессиями осуществляются с использованием токенов, что делает API масштабируемым и легко распределенным. Токены в платформе Rightech IoT Core имеют дополнительную расширенную функциональность - срок действия.

Коды состояния HTTP

Стандартные коды состояния. RESTful API использует стандартные коды состояния HTTP для передачи информации о результате выполнения запроса (например, 200 OK, 404 Not Found, 500 Internal Server Error).

Человекочитаемый вид. Коды состояния HTTP обеспечивают читаемость для человека и понимание статуса выполнения запроса.

Безопасность и аутентификация

HTTPS и токены. RESTful API обеспечивает безопасность данных с использованием протокола HTTPS, а для аутентификации используются токены.

Авторизация. Механизмы авторизации могут включать в себя различные стратегии, такие как OAuth 2.0.

Кэширование

RESTful API поддерживает кэширование данных для оптимизации производительности и снижения нагрузки на сервер.

Масштабируемость

RESTful API легко масштабируется горизонтально, поскольку каждый запрос содержит все необходимые данные.

Документация API

Самодостаточная документация. RESTful API обеспечивает самодостаточную документацию, так как структура ресурсов и операции являются явными в определениях API. С документацией подробнее можно ознакомиться по адресу (https://rightech.io/ru/developers/intro)

Swagger и OpenAPI. Дополнительные инструменты, такие как Swagger и OpenAPI, позволяют автоматически создавать документацию API. В платформе Rightech IoT Core документация реализованы с

принципами OpenAPI, но при потребности может быть легко переформирована в спецификацию Swagger.

Технологический стек платформы Rightech IoT Core

Благодаря сервисориентированной архитектуры платформы Rightech IoT Core технологический стек является независимым и ограничивается лишь технологической мощностью команды. Основной технологический стек компании : **Node.js, React, TypeScript, C++, Go, Kotlin, Swift, Python, MongoDB, PostgreSQL, Redis, TimeScaleDB, Loki, Grafana, Gitlab CI/CD, Kubernetes, gRPC, RabbitMQ.**

Node.js

Node.js - среда исполнения JavaScript (Typescript), построенная на движке V8 от Google. Позволяет разрабатывать серверные приложения на языке JavaScript в рамках клиент серверного взаимодействия архитектуры платформы Rightech IoT Core. В контексте стека технологий, Node.js предоставляет ряд преимуществ и инструментов для создания современных и масштабируемых сервисов. Основные функциональные преимущества:

- Асинхронная и неблокирующая архитектура.
- Функциональный пакетный менеджер npm.
- Эффективная работа с JSON.
- Модульная архитектура.
- Мощный фреймворк для создания API - Express.js.
- WebSocket и Real-Time Communication.
- Коробочная совместимость с NoSQL СУБД и MongoDB.
- Инструменты сборки и тестирования
- Поддержка и развитие технологии.

Node.js используется в сервисах *клиент серверного взаимодействия и построения пользовательских интерфейсов, автоматов логики,*

хранения телеметрии, оповещений и интеграций, построения отчетов и др.

C++

C++ - популярный, мощный и многофункциональный язык программирования, который широко используется в различных областях разработки программного обеспечения.
Основные функциональные преимущества и предпосылки его использования в стеке технологий платформы Rightech IoT Core следующие:

- Производительность и эффективность. C++ предоставляет низкоуровневое управление памятью и ресурсами, а также возможность написания кода, близкого к машинному языку.
- Многозадачность и параллелизм.
- Нативная компиляция кода.
- ООП и шаблонизация
- Статическая типизация и управление памятью. Позволяет построить сервис с высоким уровнем безопасности.
- Легкая переносимость на встроенные системы, при необходимости переиспользования кода на устройстве.
- Поддержка и развитие технологии.

C++ используется в сервисах *межпротокольного взаимодействия (более 20 различных сервисов) и различных агентах, встраиваемых на различные устройства или контуры предприятий.*

Go

Go, также известный как Golang, является открытым и статически типизированным языком программирования, разработанным Google. Он был создан с учетом простоты разработки, эффективности выполнения и легкости поддержки.

В платформе Rightech IoT Core нашел широкое применение для создания сервисов, решающих различные сложные и нетипичные задачи, где и показал свою эффективность.
Выделяем следующие функциональные преимущества и особенности:

- Производительность и эффективность.
- Многозадачность и параллелизм. Go предоставляет механизм горутин (goroutines) и каналов (channels) для удобной реализации многозадачности и параллелизма.
- Простота и читаемость кода. Отсутствие наследований и обобщений.
- Эффективная работа с сетью и I/O.
- Компилируемость.
- Модульность и наличие пакетов.
- Поддержка большинства современных стандартов.
- Встроенные средства тестирования.
- Поддержка и развитие технологии.

Go используется в сервисах *авторизации и аутентификации устройств, ботов, потоковых обработчиков, serverless, геосервисе и др.*

React

React — библиотека для создания пользовательских интерфейсов, разработанная Facebook. Позволяет эффективно строить интерфейсы с использованием компонентного подхода.
На текущий момент в рамках пользовательского интерфейса осуществлен полный переход на использование технологии React и JavaScript. В качестве основных характеристик и преимуществ построения интерфейсов платформы Rightech IoT Core выделяем следующие:

- Компонентная архитектура.
- Эффективный рендеринг интерфейсов и виртуальный DOM.

- One-way data binding, как мощный инструмент управления состоянием компонентов.
- React Hooks.
- Легкая интеграция с другими технологиями.
- Поддержка серверного рендеринга.
- React Native. Мощный инструмент разработки под мобильные платформы.
- Поддержка и развитие технологии.

React используется *для построения пользовательских интерфейсов платформы Rightech IoT Core основного сервиса по клиент-серверному взаимодействию.*

Kotlin

Kotlin - статически типизированный язык программирования, который работает на платформе Java Virtual Machine (JVM). Введенный компанией JetBrains, Kotlin является официальным языком программирования для Android и также может использоваться для разработки серверных приложений, веб-приложений и даже нативных мобильных приложений.
В контуре решений платформы Rightech IoT Core используется благодаря следующим возможностям:

- 100% Совместимость с Java. Различные приложения контура платформы были исторически написаны на Java и данная особенность являлась крайне важной.
- Современный синтаксис и удобство разработки.
- Компилируемость.
- Многоплатформенность разработки. Технология позволяет получать приложения как под Android так и под iOS.
- Эффективная асинхронность.
- Поддержка и развитие технологии.

Kotlin используется в компании для построения *нишевых приложений под Android, а также приложения в рамках контура платформы Rightech IoT Core - сервисного обслуживания.*

Swift

Swift - мощный и современный язык программирования, разработанный компанией Apple. Предназначен для создания приложений под операционные системы iOS, macOS, watchOS и tvOS.

В рамках компании Rightech используется как основной полноценный официальный язык программирования под iOS. Выделяем следующие преимущества:

- Разработка для iOS и macOS и интеграция из коробки с технологиями Apple.
- Статическая типизация и управление памятью.
- Выразительный современный синтаксис.
- Декларативный подход к созданию интерфейсов - SwiftUI
- Мощный пакетный менеджер и интегрированная среда разработки.
- Поддержка и развитие технологии.

На текущий момент Swift используется в компании для построения *нишевых приложений под iOS. В планах создание приложения в рамках контура платформы Rightech IoT Core для сервисного обслуживания.*

Основные сервисы платформы Rightech IoT Core

С архитектурой платформой Rightech IoT Core можно познакомиться по ссылке из названия раздела (https://disk.yandex.ru/d/GqEcjrNqTc5mUQ). Архитектура представлена с помощью компонентно-подобной нотации диаграммы UML.

Сервис межпротокольного взаимодействия (ric-gate)

Сервис межпротокольного взаимодействия. Позволяет организовать двустороннее взаимодействие с физическими устройствами по их различным протоколам. Отвечает за сериализацию и диссериализацию данных в единый универсальный внутренний протокол взаимодействия.

Сервис авторизации и аутентификации устройств (ric-auth)
Сервис, управляющая авторизацией и аутентификацией устройств.

Веб-сервис (ric-web)
Веб-сервис с обширной функциональностью, пользовательским графическим интерфейсом и REST API

Сервис симуляции устройств (ric-bot)
Сервис, имитирующий работу реальных устройств с протоколами mqtt и wialon

Сервис хранения телеметрии (ric-logger)
Высоконагруженный сервис, отвечающий за сохранение поступающей телеметрии как в режиме реального времени, так и в отложенном формате.

Сервис обработки данных (ric-handler)
Сервис обработки данных. Может функционировать с потоками данных от устройства в режиме реального времени, так и по расписанию, событию и хуку. Работает как с актуальными, так и с ретроспективными данными. Он может выполнять фильтрацию, расширение и преобразование данных.

Сервис логики (ric-logic)
Сервис для автоматизации бизнес-процессов, взаимодействия устройств и создания автоматизированных механизмов управления и оповещения. Сервис основан на ядре детерминированных конечных автоматов (Determined Finite State Machines). Графический

интерфейс позволяет гибко настраивать сценарий автоматизации практически любого бизнес-процесса.

Геосервис (ric-geo)
Сервис для работы с геоданными. Контроль входов и выходов из геозон, проверка географических маршрутов, внутреннее и внешнее позиционирование. Сервис имеет собственный внутренний тайловый сервер.

Сервис оповещения и интеграций (ric-notify)
Сервис для взаимодействия с внешними информационными системами и сервисами. В основном используется для генерации уведомлений во внешнюю среду.

Сервис отчетов (ric-reports)
Генерирует отчеты за выбранный период как на основе шаблонов, так и пользовательского типа. Имеются механизмы автоматической генерации. Рендеринг осуществляется в 4 популярных форматах документов.

Сервис работы с lora стеком (ric-lora-stack)
Сервис для взаимодействия с устройствами, работающими в технологическом стеке lorawan

Сервисное приложение (ric-service)
Мобильное приложение для обслуживающего персонала. Позволяет обслуживать объекты, введенные в платформу. Контролируйте сроки и предоставляйте доступ к их управлению.

Приложение - версия платформы (ric-app)
Приложение с базовыми возможностями платформы. Версии для iOS и Android.

Архитектура платформы

С архитектурой платформой Rightech IoT Core можно познакомиться по ссылке из названия раздела (https://disk.yandex.ru/d/GqEcjrNqTc5mUQ). Архитектура представлена с помощью компонентной диаграммой UML.