?????? ?????????? ????????

Обзор платформы ARTEX, архитектура, протоколы и потоки данных, серверные модули, хранение данных, развёртывание, безопасность, мониторинг и планы развития.

- Обзор платформы
  - О платформе ARTEX
  - Глоссарий
- Архитектура
  - Общая архитектура
  - Протоколы и потоки данных
  - Серверные модули
  - Хранение данных
- Эксплуатация
  - Развёртывание сервера
  - Доступ и безопасность
  - Мониторинг
- Развитие платформы
  - Этапы и задачи разработки
  - Дорожная карта

????? ??????????

Назначение платформы ARTEX, области применения, термины

Обзор платформы

? ????????? ARTEX

ARTEX (Autonomous Realtime EXchange) — платформа для удалённого управления роботами, мониторинга и обучения моделей искусственного интеллекта на реальных и виртуальных устройствах. Платформа объединяет в единую среду три участника: **роботов**, **операторов** (людей) и **модели ИИ**, и обеспечивает между ними обмен видео, телеметрией и командами управления в реальном времени.

> Главный принцип ARTEX: «Не тот побеждает, у кого мощнее железо, а тот, у кого умнее алгоритм». Платформа позволяет одинаково подключать к роботу человека, модель ИИ или их комбинацию — и сравнивать, кто управляет лучше.

??? ?????? ?????????

- **Подключает роботов** любых типов — RC-танки и машинки (ESP32 / STM32), наземных роботов (Raspberry Pi, ROS2), дроны (PX4 / ArduPilot), IoT-датчики, светодиодные устройства. На стороне робота устанавливается SDK `artex-robot` (Python).
- **Коммутирует потоки данных** «всех со всеми»: видео и телеметрия от робота одновременно доставляются оператору и моделям ИИ, а команды от оператора или модели — нужному роботу.
- **Даёт оператору веб-пульт** — FPV-видео от первого лица, телеметрию, кнопки управления — на ноутбуке, телефоне или VR-устройстве.
- **Запускает модели ИИ** в распределённой вычислительной среде (Docker, Kubernetes): детекция объектов (YOLO), ассистент оператора (Assist AI), агенты обучения с подкреплением (RL), пользовательские модели.
- **Собирает датасеты**: каждое действие оператора, кадр видео и значение телеметрии сохраняются, что позволяет обучать следующую версию модели методом Imitation Learning.
- **Администрирует систему**: мониторинг подключённых роботов и операторов, ограничение «видимости» роботов для операторов, настройка хранения данных.

???????? ???????

Термин	Значение

Робот	Любое физическое или виртуальное устройство, подключённое к платформе через SDK `artex-robot`. Передаёт видео и телеметрию, принимает команды.
ХАБ платформы (ARTEX HUB)	Серверное ядро (Go / C++): приём потоков, неблокирующий кэш, коммутация роботов, операторов и моделей, запись в хранилище.
Оператор	Зарегистрированный пользователь, который через веб-интерфейс наблюдает за роботом и управляет им.
Модель ИИ	Программа в вычислительной среде, которая получает от ХАБа видео/телеметрию и возвращает команды управления.
Телеметрия	Набор параметров робота: скорость, координаты, заряд АКБ, данные гироскопа и акселерометра (IMU), состояние агрегатов, признаки (векторы), выделенные бортовыми моделями ИИ.
Команда управления	Управление движением, поворотом камеры, дополнительными аксессуарами. Источник — оператор или модель ИИ.
Видео	Изображение с камер робота, а также данные радаров, лидаров и других сенсоров, преобразованные в видеоформат.
Модуль / конвейер	Функциональный блок сервера с входами и выходами. Модули соединяются в конвейер (pipeline) в визуальном редакторе.
Полигон	Площадка, на которой работают роботы: физическая (реальная трасса) или виртуальная (симуляция).

Подробный список терминов — на странице Глоссарий.

??????? ???????????

Направление	Кому	Что даёт платформа
Обучение	Школьники, кадеты, студенты, аспиранты	Управление реальными роботами, соревнования между классами и командами, изучение ИИ через игру
Соревнования	Команды, лиги	Режимы «Человек vs Человек», «Человек vs ИИ», «ИИ vs ИИ», ELO-рейтинг и публичный лидерборд
Разработка	Университеты, лаборатории	Тестирование моделей ИИ на реальных устройствах, сбор датасетов, эксперименты и публикации

Направление	Кому	Что даёт платформа
Индустрия	Склады, производство, логистика, ведомственные центры	Управление парком автономных систем, мониторинг, маршруты, безопасность; отработка управления БПЛА

??? ???????? — ???? ?????????

Физический полигон — RC-танки, машинки и дроны на реальной трассе. На каждом устройстве — FPV-камера, в реальном времени передаётся телеметрия (скорость, GPS, заряд батареи, датчики).

Виртуальный полигон — симуляция среды (Unity / Gazebo) для прототипирования ИИ: тысячи итераций обучения без риска для оборудования. Модель, обученная в симуляции, переносится на реальное устройство без изменения кода интеграции, потому что виртуальный робот подключается к платформе через тот же протокол, что и реальный.

????????? ???????????

> Реализовано: серверная платформа, SDK `artex-robot` для подключения роботов, передача видео и телеметрии, веб-интерфейс оператора со страницами управления устройствами Robot, LED и Drone, визуальный редактор модулей, модули ClientIO, ConsoleLog, RobotWS и Yolo.

Разработка ведётся по этапам: **Демо-прототип** → **Прототип «Альфа»** для тестирования → развитие полигона и соревнований → экосистема «AI Arena». Подробности — на страницах «Этапы и задачи разработки» и «Дорожная карта».

????? ????????????

Книга	Содержание
Сервер управления роботами	Обзор платформы, архитектура, протоколы, серверные модули, хранилище, развёртывание, безопасность, мониторинг, планы
ПО для робота	SDK `artex-robot`: установка, конфигурация, справочник API, протокол обмена с сервером, примеры, устранение неполадок

Книга	Содержание
Управление роботами с помощью ИИ	Веб-интерфейс оператора, режимы управления и состязаний, подключение и разработка моделей ИИ, конвейеры модулей, сбор данных и обучение

Обзор платформы

?????????

Термины и сокращения, которые используются в документации платформы ARTEX.

Термин	Описание
ARTEX	Autonomous Realtime EXchange — название платформы и её программного фреймворка.
ARTEX HUB, ХАБ	Серверное ядро платформы: принимает потоки от роботов, хранит актуальное состояние в кэше и раздаёт данные операторам и моделям.
Assist AI	Режим «второго пилота»: модель ИИ помогает оператору (подсказывает, корректирует или перехватывает часть функций управления).
DISARM	Команда полного обесточивания силовых цепей робота/дрона (безопасный режим).
d_meta (динамические метаданные)	Изменяемое состояние робота на стороне SDK: `sid`, `ready`, `telem_run`, `video_run`. Запрашивается сервером через heartbeat.
s_meta (статические метаданные)	Паспорт робота: `name`, `type`, `telem_can`, `video_can`. Передаётся серверу при подключении.
ELO	Система рейтинга команд/моделей по результатам поединков.
FPV	First Person View — видео «от первого лица» с камеры робота.
gRPC	Протокол межсервисного взаимодействия модулей сервера.
Heartbeat	Периодический опрос роботов сервером для контроля связи и готовности.
Imitation Learning	Обучение модели на записанных действиях оператора («учимся у человека»).
IMU	Инерциальный измерительный блок: гироскоп + акселерометр.
JWT / RBAC	Токены авторизации и ролевая модель доступа к ресурсам платформы.
K8s	Kubernetes — оркестратор контейнеров, используемый для масштабирования сервисов и моделей ИИ.
MinIO / S3	Объектное хранилище для видео, датасетов и снапшотов моделей.

Термин	Описание
MQTT	Лёгкий протокол publish/subscribe для IoT-датчиков.
Namespace `/robot`	Пространство имён Socket.IO, в котором работают роботы.
Pipeline, конвейер	Цепочка модулей сервера, соединённых через порты в визуальном редакторе.
Порт модуля	Именованный вход или выход модуля, например `robot.video`, `robot.telemetry`, `yolo.detect`.
RL	Reinforcement Learning — обучение с подкреплением.
RTSP	Протокол потоковой передачи видео от камер.
SDK `artex-robot`	Python-библиотека для подключения робота к платформе (`pip install artex-robot`).
SFU	Selective Forwarding Unit — сервер ретрансляции WebRTC-потоков нескольким получателям.
SID	Уникальный идентификатор подключения устройства на платформе.
Socket.IO	Протокол поверх WebSocket, через который SDK робота обменивается событиями с сервером.
Time Series DB	База данных временных рядов для хранения телеметрии.
Vault	Внутреннее хранилище SDK робота: метаданные, последние значения телеметрии и кадры видео.
WebRTC	Протокол передачи видео в браузер с малой задержкой.
WebSocket	Двунаправленный канал между браузером оператора и сервером.
YOLO	Семейство нейросетей для детекции объектов в реальном времени.
Вычислительная среда	Распределённая (удалённая) среда запуска моделей ИИ операторов (Docker, K8s).
Полигон	Физическая трасса или виртуальная симуляция, где работают роботы.
Приоритетное управление	Правило, по которому команда оператора имеет приоритет над командами модели ИИ (режим «подстраховки»).

??????????

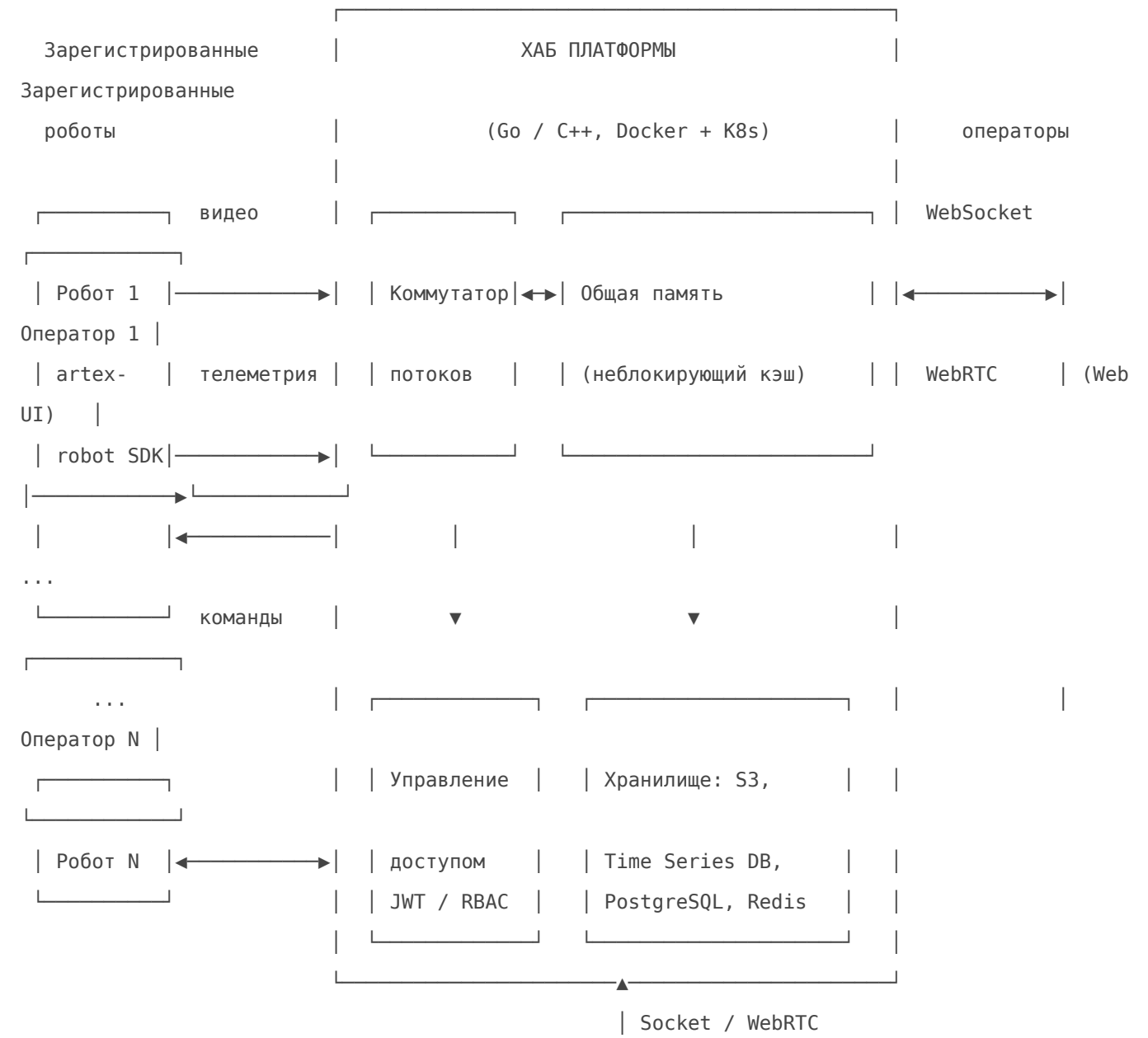
Компоненты, протоколы, серверные модули и хранение данных

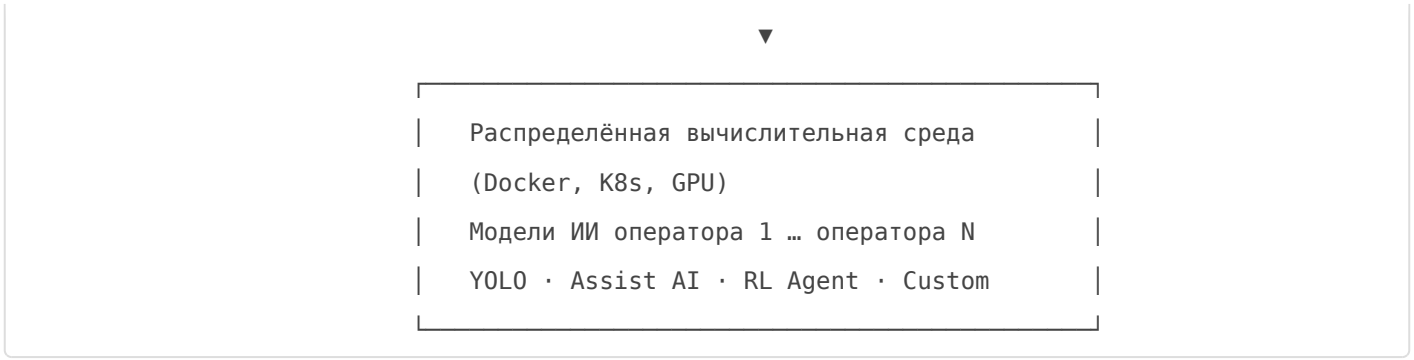
Архитектура

????? ???????????

Платформа ARTEX построена по схеме «звезда»: в центре находится **ХАБ платформы**, к которому независимо подключаются роботы, операторы и модели ИИ. Никто из участников не общается напрямую — все потоки проходят через ХАБ, который коммутирует их, кэширует актуальное состояние и сохраняет данные.

????? ?????





??? ?????? ??????

Между участниками передаются три типа данных. Каждый поток идёт по своему каналу, чтобы тяжёлое видео не задерживало команды управления.

№	Поток	Направление	Содержание
1	**Телеметрия, признаки**	Робот → ХАБ → оператор и модели ИИ	Скорость, координаты, данные гироскопов и акселерометров, заряд АКБ, состояние дополнительных агрегатов, признаки (векторы), выделенные бортовыми моделями ИИ
2	**Управление**	Оператор / модель ИИ → ХАБ → робот	Управление движением, поворотом камеры, дополнительными аксессуарами. Основной источник — модель ИИ; оператор управляет при обучении или «подстраховке» модели
3	**Видео**	Робот → ХАБ → оператор и модели ИИ	Видео с камер, а также данные радаров, лидаров и т.п., преобразованные в видео. Оператору — для управления от первого лица и контроля работы ИИ; моделям — для анализа

?????????? ?????????

Компонент	Технологии	Ответственность

Робот	Python, SDK `artex-robot`, OpenCV	Съёмка видео и передача на сервер; сбор и передача телеметрии; работа бортовых моделей ИИ (обработка видео и данных датчиков); получение и применение команд управления
Платформа (ХАБ)	Go, C++	Идентификация роботов; приём видео и телеметрии, загрузка в кэш и хранилище; неблокирующий кэш с методами `push` и `get`; коммутация роботов, операторов и моделей; подключение операторов и моделей к видео и данным
Оператор (Web-приложение)	React, WebSocket, WebRTC	Приём видео и телеметрии от робота; управление роботом и передача управления на платформу; мониторинг работы моделей ИИ, их входных и выходных данных и логов; адаптивный интерфейс для ноутбука и телефона
Вычислительная среда	Docker, Kubernetes, GPU	Обработка видео (YOLO); обработка временных рядов, RL; формирование команд управления
Хранилище	S3 / MinIO, Time Series DB, PostgreSQL, Redis	Сохранение видео, телеметрии, команд и логов; развёртывание, подключение и тестирование хранилища
Управление платформой (WEB-портал)	Web	Мониторинг подключённых роботов и операторов; ограничение «видимости» роботов для операторов; настройка сохранения данных

???? ????????????? ?????????

Класс устройств	Типичные контроллеры	Способ подключения
RC-танки, машинки	ESP32 / STM32 + бортовой компьютер (Raspberry Pi 4)	SDK `artex-robot`
Наземные роботы	ROS2 / собственное ПО	SDK `artex-robot` (узел-мост)
Дроны (БПЛА)	PX4 / ArduPilot + компаньон-компьютер	SDK `artex-robot`
IoT-датчики	MQTT / GPIO	MQTT-брокер (планируется)

Класс устройств	Типичные контроллеры	Способ подключения
Светодиодные устройства (LED, RGB-лента)	Любой Python-контроллер	SDK `artex-robot` (тип `LED`)
Виртуальные роботы	Unity / Gazebo	SDK `artex-robot` — тот же протокол, что и у реальных

???????? ????????????

1. **Всё через ХАБ.** Роботы, операторы и модели не знают адресов друг друга — это упрощает подключение, контроль доступа и запись данных.
2. **Неблокирующая общая память.** ХАБ хранит только *актуальный* кадр и *актуальный* вектор телеметрии (с отметкой времени). Медленный потребитель не тормозит быстрого: каждый читает последнее значение методом `get`.
3. **Модель и человек равноправны.** Команды от оператора и от модели ИИ имеют одинаковый формат; различается только автор и приоритет.
4. **Модульность.** Серверная логика собирается из модулей в визуальном редакторе с горячей перезагрузкой (hot reload). Каждый сервис — отдельный Docker-контейнер.
5. **Всё записывается.** Видео, телеметрия и команды (включая автора команды) сохраняются для анализа и обучения.

Подробнее: Протоколы и потоки данных, Серверные модули, Хранение данных.

Архитектура

?????????? ? ?????? ??????

На странице описано, по каким протоколам участники обмениваются данными с ХАБом платформы и как движется каждый поток.

??????? ??????? ??????? ?????

№	Связь	Поток	Протокол
1	Робот ↔ ХАБ	Телеметрия, признаки	Socket (в текущей версии SDK — Socket.IO, namespace `/robot`)
2	Робот ↔ ХАБ	Управление	Socket (Socket.IO, событие `command`)
3	Робот → ХАБ	Видео	RTSP / WebRTC или «кадры» через Socket (в текущей версии — JPEG-кадры через Socket.IO)
4	Внутри ХАБа	Все потоки	Общая память — неблокирующий кэш (словарь)
5	ХАБ ↔ оператор	Контроль и обучение робота	WebSocket (данные и команды), WebRTC (видео)
6	ХАБ ↔ модели ИИ (вычислительная среда)	Телеметрия, управление, видео	Socket (телеметрия и управление), WebRTC / Socket (видео)
7	Модули сервера между собой	Служебные вызовы	REST API + gRPC
8	IoT-датчики → ХАБ	Показания датчиков	MQTT (publish/subscribe, QoS 1) — планируется
9	Администратор → сервер	Обслуживание	SSH

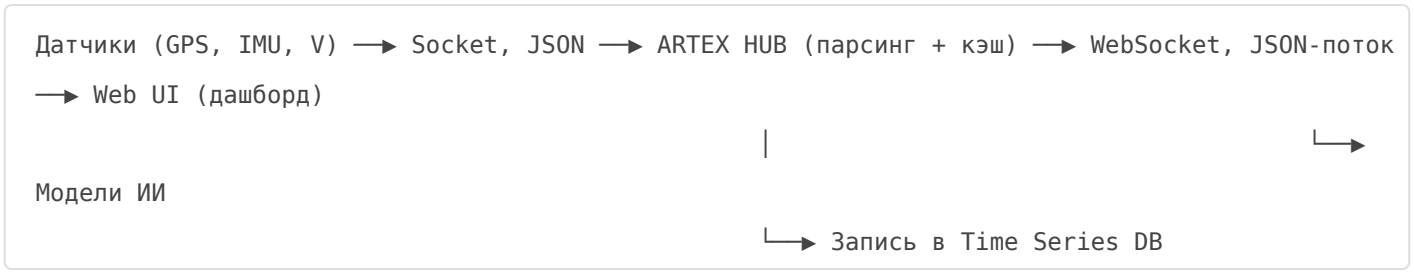
> Протокол робота подробно описан в книге «ПО для робота» на странице Протокол обмена с сервером.

????? 1. ?????



- Робот передаёт видео либо потоком RTSP, либо отдельными кадрами (JPEG, качество 80) через Socket.
- ХАБ хранит в кэше **актуальный кадр** (или набор из X последних кадров) и направляет его в вычислительную среду и оператору.
- Для браузера видео преобразуется в WebRTC; SFU ретранслирует поток нескольким получателям без повторного кодирования.
- Параллельно видео записывается в объектное хранилище S3.

????? 2. ??????????



- Телеметрия передаётся в формате JSON, например: `{"speed": 1.2, "gps": [55.65, 37.66], "battery": 11.8, "imu": {...}, "timestamp": 1790155251.12}`.
- Состав полей не фиксирован: робот может добавить новый датчик без изменения сервера и интерфейса.
- ХАБ хранит **актуальный вектор** телеметрии с учётом времени создания и раздаёт его потребителям.

????? 3. ???????



- Команда содержит автора (`author`) и полезную нагрузку вида `{тип_команды: команда}`.
- ХАБ проверяет права автора на робота и применяет правило приоритета: команда оператора имеет приоритет над командой модели (режим «подстраховки»).
- Команды `Стоп` и `DISARM` обрабатываются вне очереди.

??????? ??????????? ????????

Параметр	Целевое значение
Камера → оператор	< 200 мс
Команда → привод	< 50 мс
Частота телеметрии	до 30 сообщений/с (JSON)
Частота отправки данных SDK по умолчанию	20 сообщений/с (`max_send_freq = 0.05` с)

????????? ???? ???????????? ??????

1. Робот открывает соединение с сервером и передаёт паспорт устройства (`name`, `type`, `telem_can`, `video_can`).
2. Сервер присваивает подключению идентификатор (SID) и регистрирует робота в реестре устройств.
3. Сервер периодически отправляет **heartbeat**: запрашивает динамическое состояние (`d_meta`) и сигнал готовности (`Ready`).
4. Получив `Ready`, робот запускает циклы отправки телеметрии и видео.
5. Сервер пересылает роботу команды от операторов и моделей.
6. При обрыве связи SDK автоматически переподключается (до 5 попыток с задержкой 1–5 с); сервер помечает робота как отключённого и уведомляет интерфейс (порт `robot.disconnected`).

?????? ?????? ?????? ????

ХАБ предоставляет операторам и моделям два базовых метода работы с общей памятью:

Метод	Назначение
`push(key, value)`	Записать новое значение (кадр, вектор телеметрии, команду) с отметкой времени. Не блокирует читателей.
`get(key)`	Получить последнее актуальное значение. Не ждёт появления нового значения.

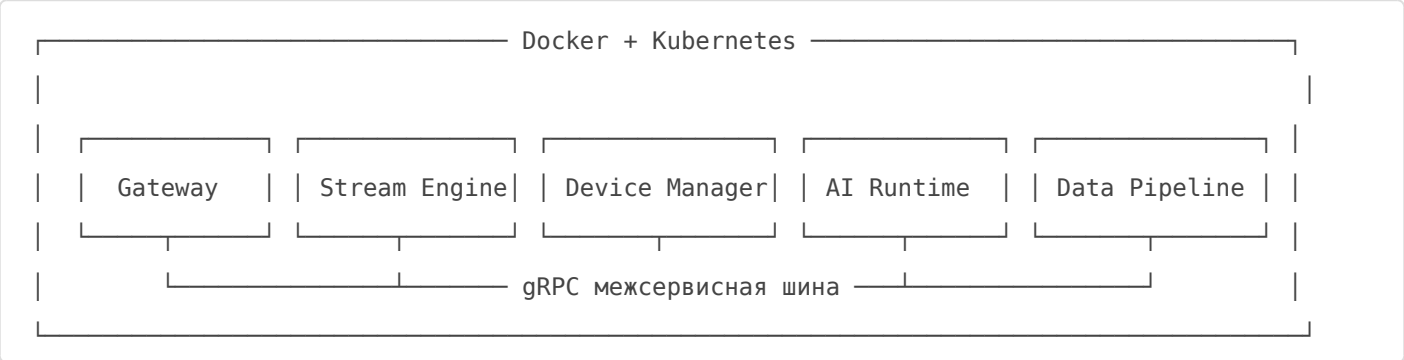
Рекомендуемая схема именования ключей — `<робот>.<поток>[.<канал>]`, например `robot_1.video.rgb`, `robot_1.telemetry`, `robot_1.command`.

Архитектура

?????????? ??????

Серверная часть ARTEX состоит из независимых сервисов на Go и C++. Каждый сервис упакован в Docker-контейнер, оркестрация и масштабирование выполняются Kubernetes. Сервисы взаимодействуют через общую шину gRPC, внешние клиенты — через REST API и WebSocket.

??????? ?????????



Сервис	Назначение	Основные функции
Gateway (API Gateway)	Единая точка входа для внешних клиентов	REST API и WebSocket; JWT-авторизация; ограничение частоты запросов (rate limiting)
Stream Engine	Коммутатор видео	Приём RTSP и кадров, преобразование RTSP → WebRTC, мультиплексирование видео и данных, транскодирование
Device Manager	Управление устройствами	Реестр устройств, подключение устройств, передача команд, контроль статуса и heartbeat
AI Runtime	Среда исполнения моделей	Загрузка моделей (YOLO / RL / Custom), распределение GPU, ведение входов, выходов и логов моделей
Data Pipeline	Запись данных	Сохранение в S3, Time Series DB, PostgreSQL; горячий кэш Redis

?????????? ????????? (pipeline)

Прикладная логика обработки данных собирается из **модулей** в визуальном редакторе. Каждый модуль отвечает за одну задачу и имеет именованные порты — входы и выходы. Пользователь перетаскивает модули на рабочее поле и соединяет порты, формируя конвейер. Изменения применяются без перезапуска сервера (hot reload).

Типовой конвейер автономного управления:

```
 Захват видео ⟶ Детекция (YOLO) ⟶ Трекинг объектов ⟶ Принятие решения ⟶ Команда
устройству
                                                                          |
                                                                          └⟶ Логирование (S3 + DB)
```

????????????? ??????

Модуль	Назначение	Порты
ClientIOModule	Хостит веб-интерфейс управления устройствами	`robot.connected`, `robot.disconnected`, `robot.telemetry`, `robot.video`
ConsoleLogModule	Вывод всех проходящих данных в консоль сервера для отладки	—
RobotWSModule	Мост между физическим устройством и платформой (WebSocket)	`robot.command` (выход к устройству), `robot.telemetry`, `robot.video`, `robot.connected`, `robot.disconnected`
RobotsModule	Источник данных от группы роботов (используется аналогично RobotWSModule)	`robot.telemetry`, `robot.video` и др.
YoloModule	Детекция объектов на видеопотоке в реальном времени	`robot.video` (вход), `yolo.detect` (выход)
VideoRecordModule	Запись видеопотока в файл по командам `video_start_record` / `video_stop_record`	`robot.video`, команды

Подробное описание модулей и их использования в интерфейсе — на странице «Веб-интерфейс управления роботами» книги «Управление роботами с помощью ИИ», примеры конвейеров — на странице Конвейеры модулей.

??????? ?????????? ??????

- Выход можно соединить с несколькими входами — данные будут доставлены всем получателям.
- Порты соединяются по типу данных: `robot.video` ↔ `robot.video`, `robot.telemetry` ↔ `robot.telemetry`.
- ConsoleLogModule можно подключить к любому порту для отладки.

- Для записи данных используйте модули записи (VideoRecordModule) или сервис Data Pipeline.

??????? ??????????? ???????

Интерфейс	Клиенты	Назначение
Socket.IO (namespace `/robot`)	Роботы (SDK)	Регистрация, телеметрия, видео, команды
WebSocket	Web UI оператора, модели ИИ	Данные в реальном времени и команды
WebRTC	Браузер оператора, модели ИИ	Видео с малой задержкой
REST API	Web UI, портал управления, внешние системы	Пользователи, устройства, сессии, настройки
gRPC	Внутренние сервисы	Межсервисное взаимодействие
SSH	Администраторы	Обслуживание серверов

Архитектура

???????? ??????

Платформа сохраняет все данные, проходящие через ХАБ: они нужны для разбора соревнований, отладки моделей и обучения новых моделей ИИ. Для каждого типа данных используется подходящее хранилище.

?????????

Хранилище	Что хранится	Особенности
S3 / MinIO	Видео, датасеты, снапшоты (версии) моделей ИИ, лог-файлы	Объектное хранилище; разделение по бакетам
Time Series DB	Телеметрия: GPS, скорость, IMU, заряд батареи, показания датчиков	Хранение и выборка по времени, агрегация
PostgreSQL	Пользователи, реестр устройств, сессии, права доступа, настройки	Реляционные данные
Redis	Горячий кэш, состояние в реальном времени	In-memory; актуальные значения для ХАБа
Файловая система	Локальные логи и временные файлы	Используется на этапе демо-прототипа

??? ????????????? ? ??? ??????

На каждом сеансе управления роботом ведётся лог данных:

Данные	Хранилище	Содержание записи
Видео	S3	Кадры/фрагменты видео с отметкой времени и идентификатором робота и камеры
Телеметрия	Time Series DB	Вектор телеметрии с временем создания на роботе и временем приёма на сервере
Команды оператора	Time Series DB / S3	Команда, автор, время, робот
Команды моделей ИИ	Time Series DB / S3	Команда, модель, версия модели, время, робот
Выходы моделей	S3	Результаты детекции, логи модели

> Благодаря синхронной записи «кадр — телеметрия — команда оператора» каждая сессия ручного управления превращается в готовый датасет для Imitation Learning.
>
> Подробнее — на странице Сбор данных и обучение моделей.

?????????????? ?????????? ?????????? ??????????

```
s3://artex-video/<robot_sid>/<session_id>/<camera>/<timestamp>.jpg
s3://artex-datasets/<dataset_name>/<version>/...
s3://artex-models/<model_name>/<version>/weights.*
s3://artex-logs/<service>/<date>/...
```

?????????? ?????????? ??????

Параметры сохранения задаются администратором в портале управления платформой:

- какие потоки записывать (видео, телеметрия, команды) — для всех роботов или выборочно;
- частота прореживания видео при записи (например, каждый N-й кадр);
- срок хранения данных;
- включение записи видео по команде оператора (`video_start_record` / `video_stop_record`).

???? ????-??????????

На этапе демо-прототипа логи сохраняются в файлы; развёртывание, подключение и тестирование S3 и Time Series DB выполняются на этапе прототипа «Альфа».

??????????

Развёртывание, доступ, безопасность и мониторинг

Эксплуатация

?????????????? ???????

Руководство для администратора: как развернуть сервер ARTEX и окружающую инфраструктуру.

> Имена образов и переменные окружения ХАБа в примерах ниже приведены как шаблон. Перед развёртыванием сверьте их с актуальным репозиторием платформы.

?????? ???????????????

Компонент	Контейнер	Обязательность
ХАБ платформы (Gateway, Stream Engine, Device Manager)	образ ARTEX HUB	Обязательно
Веб-интерфейс оператора	статический фронтенд (React), раздаётся ХАБом (ClientIOModule) или отдельным веб-сервером	Обязательно
Redis	`redis`	Рекомендуется
PostgreSQL	`postgres`	Обязательно для многопользовательского режима
S3-совместимое хранилище	`minio/minio`	Для записи видео и датасетов
Time Series DB	например, TimescaleDB или InfluxDB	Для записи телеметрии
AI Runtime / модели	отдельные контейнеры с GPU	При использовании моделей ИИ
Prometheus + Grafana	`prom/prometheus`, `grafana/grafana`	Рекомендуется

?????????? ? ???????

Ресурс	Минимум (демо, 1–5 роботов)	Рекомендуется (соревнования, модели ИИ)
CPU	4 ядра	8+ ядер
RAM	8 ГБ	32 ГБ
GPU	не требуется	NVIDIA с поддержкой CUDA для YOLO / RL

Ресурс	Минимум (демо, 1–5 роботов)	Рекомендуется (соревнования, модели ИИ)
Диск	50 ГБ SSD	1 ТБ+ под видео и датасеты
Сеть	100 Мбит/с	1 Гбит/с; ~2–5 Мбит/с на каждый видеопоток
ПО	Linux, Docker, Docker Compose	Kubernetes-кластер

??? 1. ??????????? ?????

```
# Docker и Compose
curl -fsSL https://get.docker.com | sh
sudo usermod -aG docker $USER

# Для моделей ИИ на GPU — NVIDIA Container Toolkit
# https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/
```

Доступ администраторов к серверу — только по SSH-ключам.

??? 2. ??????????????? ???????? (Docker Compose)

```
# docker-compose.infra.yml
services:
  redis:
    image: redis:7
    restart: unless-stopped
    ports: ["6379:6379"]

  postgres:
    image: postgres:16
    restart: unless-stopped
    environment:
      POSTGRES_DB: artex
      POSTGRES_USER: artex
      POSTGRES_PASSWORD: ${POSTGRES_PASSWORD}
    volumes: ["pg_data:/var/lib/postgresql/data"]
```

```
  minio:
    image: minio/minio
    restart: unless-stopped
    command: server /data --console-address ":9001"
    environment:
      MINIO_ROOT_USER: ${MINIO_USER}
      MINIO_ROOT_PASSWORD: ${MINIO_PASSWORD}
    ports: ["9000:9000", "9001:9001"]
    volumes: ["minio_data:/data"]

  timescaledb:
    image: timescale/timescaledb:latest-pg16
    restart: unless-stopped
    environment:
      POSTGRES_PASSWORD: ${TSDB_PASSWORD}
    volumes: ["tsdb_data:/var/lib/postgresql/data"]

volumes:
  pg_data:
  minio_data:
  tsdb_data:
```

```
docker compose -f docker-compose.infra.yml up -d
```

??? 3. ?????? ????

```
# docker-compose.hub.yml (шаблон)
services:
  artex-hub:
    image: <registry>/artex-hub:<version>
    restart: unless-stopped
    ports:
      - "80:80"        # веб-интерфейс, WebSocket, Socket.IO для роботов
    environment:
      REDIS_URL: redis://redis:6379
      POSTGRES_DSN: postgres://artex:${POSTGRES_PASSWORD}@postgres:5432/artex
      S3_ENDPOINT: http://minio:9000
```

```
      JWT_SECRET: ${JWT_SECRET}
```

Роботы по умолчанию подключаются к адресу `http://artex.arcon:80` — это значение параметра `server` в SDK. Настройте DNS-запись (или `/etc/hosts` на роботах) либо укажите реальный адрес сервера в `params.json` каждого робота.

??? 4. ????????

1. Откройте веб-интерфейс в браузере: `http://<адрес-сервера>/`. Должна открыться страница «Подключения».
2. Запустите на тестовом устройстве минимальный пример из раздела Быстрый старт.
3. Убедитесь, что устройство появилось в таблице подключений, а телеметрия обновляется.

????????????? ? Kubernetes

Для продуктивного режима каждый сервис разворачивается отдельным Deployment:

- **Gateway / Stream Engine / Device Manager** — горизонтально масштабируются; для Socket.IO и WebSocket включите «липкие» сессии (sticky sessions) на Ingress.
- **AI Runtime** — Deployment на узлах с GPU (`nodeSelector` / `tolerations`), ресурс `nvidia.com/gpu`.
- **Хранилища** — StatefulSet или управляемые сервисы.
- Секреты (`JWT_SECRET`, пароли БД, ключи S3) — через Kubernetes Secrets.
- Метрики — через ServiceMonitor для Prometheus.

??????????

1. Обновите образ в манифесте/compose-файле.
2. `docker compose pull && docker compose up -d` (или `kubectl rollout restart deployment/<имя>`).
3. Роботы переподключатся автоматически (SDK делает до 5 попыток с задержкой 1–5 с). Если сервер был недоступен дольше — перезапустите ПО на роботах.

Эксплуатация

?????? ? ???????????

Платформа работает с физическими устройствами, поэтому контроль доступа и безопасность — обязательная часть конфигурации.

????

Роль	Права
Администратор	Управление платформой через WEB-портал: мониторинг роботов и операторов, настройка видимости роботов, настройка хранения данных, управление пользователями и API-ключами
Оператор	Подключение к доступным ему роботам, просмотр видео и телеметрии, управление, запуск своих моделей ИИ, мониторинг их работы
Модель ИИ	Сервисная учётная запись: чтение видео и телеметрии назначенных роботов, отправка команд управления
Робот	Устройство: передача видео и телеметрии, получение команд
Зритель (соревнования)	Просмотр трансляции и лидерборда без права управления

?????????

- **JWT** — авторизация пользователей веб-интерфейса и WebSocket-подключений.
- **RBAC** — ролевая модель: какие роботы видны оператору и какие действия ему разрешены.
- **API-ключи** — для моделей ИИ и внешних систем.
- **SSH-ключи** — для администрирования серверов.
- **Rate limiting** в Gateway — защита от перегрузки командами.

????????? ???????

Администратор ограничивает, каких роботов видит каждый оператор. Оператор может подключиться и управлять только видимыми ему роботами; модели ИИ оператора получают доступ к тем же роботам.

> На этапе демо-прототипа форма ограничения «видимости» роботов для операторов реализуется без функционала (только форма). Поле «Безопасность» в таблице подключений зарезервировано под будущие настройки прав доступа.

????????? ???????????

Одновременно роботу могут отправлять команды оператор и модели ИИ. Правила:

1. Команды **Стоп** и **DISARM** от любого авторизованного источника исполняются немедленно.
2. Команда оператора имеет приоритет над командой модели ИИ (режим «подстраховки»).
3. Модель ИИ управляет роботом, пока оператор не вмешался; после вмешательства управление возвращается модели по решению оператора.

??????? ??????????? ????????????

- Перед запуском автономного режима проверьте работу кнопки **Стоп** на конкретном роботе.
- Бортовое ПО должно останавливать приводы при потере связи с сервером (см. Примеры роботов в книге «ПО для робота»).
- Для дронов после посадки всегда выполняйте **DISARM**.
- Используйте отдельную сеть (VLAN / Wi-Fi) для роботов полигона.
- Храните секреты (JWT, пароли, ключи) только в переменных окружения / Secrets, не в репозитории.
- Для публичного доступа используйте HTTPS/WSS (TLS-терминация на Ingress или обратном прокси).

Эксплуатация

??????????

??? ??????????????

Объект	Где смотреть	Показатели
Подключённые роботы	Портал управления, страница «Подключения» веб-интерфейса	Список устройств, SID, тип, статус, время последнего heartbeat
Подключённые операторы	Портал управления	Кто подключён, к каким роботам
Модели ИИ	Web UI оператора (мониторинг моделей), AI Runtime	Входные и выходные данные, команды, логи, загрузка GPU
Сервисы платформы	Prometheus + Grafana	Задержки потоков, частота кадров и телеметрии, ошибки, использование CPU/RAM/сети
Серверная логика	ConsoleLogModule	Все данные, проходящие через выбранные порты конвейера
Робот	Логи SDK (`all.log`, `error.log`)	События подключения, heartbeat, полученные команды, ошибки

???????? ??????? ? ??????

Метрика	Норма	Действие при отклонении
Задержка видео камера → оператор	< 200 мс	Снизить разрешение/качество кадра, проверить сеть
Задержка команда → привод	< 50 мс	Проверить загрузку ХАБа и сеть робота
Пропуск heartbeat	0	Проверить питание и связь робота
Частота телеметрии	стабильна (до 30/с)	Проверить основной цикл робота

????????? ????? ? ?????? ????????

Сервер может запросить у робота файлы логов командой типа `file` — SDK отвечает событием `file` с содержимым:

Команда	Результат
`{"file": "log"}`	Содержимое `all.log`
`{"file": "error"}`	Содержимое `error.log`
`{"file": "clear"}`	Очистка обоих файлов логов

Логи в файлы пишутся, если на роботе задано `"logger_type": "file"`. Подробнее — Протокол обмена с сервером.

Grafana: ?????????????? ??????

- Число подключённых роботов / операторов / моделей.
- Задержки по каждому потоку (p50, p95, p99).
- Пропускная способность видео по роботам.
- Ошибки Gateway (4xx/5xx), отказы авторизации.
- Загрузка GPU и очередь задач AI Runtime.

???????? ??????????

Этапы разработки и дорожная карта

Развитие платформы

????? ? ?????? ???????????

Разработка платформы ведётся по этапам. Для каждого этапа определены задачи по всем компонентам: робот, платформа, оператор, среда вычислений, среда хранения, управление.

???? 1. ????-????????

Цель — показать полный замкнутый контур: робот передаёт видео и телеметрию, оператор и простая модель ИИ управляют роботом через платформу.

Компонент	Задачи
Робот	Передача видео по кадрам; передача телеметрии (АКБ, положение, данные датчиков); приём и применение команд управления
Платформа	Приём данных от робота (видео, телеметрия); приём и передача команд управления от оператора; передача команд управления от моделей; доступ операторов и моделей «всех-ко-всему»; видео — захват актуального кадра (набора X кадров) и направление в среду вычислений и оператору; телеметрия — предоставление актуального вектора с учётом времени создания; ведение лога данных (телеметрия, видео, команды оператора и моделей)
Оператор	Подключение и выбор робота; просмотр видео от робота; приоритетное управление роботом: вперёд, назад, влево, вправо, стоп; управление камерой (поворот); мониторинг управления (ответов) от моделей
Среда вычислений	Обработка кадров (YOLO); обработка телеметрии; формирование команд управления по правилам (см. ниже); передача команд нужному роботу через платформу
Среда хранения	Сохранение логов в файл
Управление	Мониторинг подключённых к платформе роботов; мониторинг подключённых операторов; форма ограничения «видимости» роботов для операторов (пока без функционала)

?????????????? ??????? ??????

Условие	Команда
Напряжение АКБ ниже 11 В	«стоп»
Модель видит белый прямоугольник	«вперёд»
Модель видит красный прямоугольник	«стоп»

Реализация этих правил разобрана на странице Разработка модели ИИ для платформы.

???????? ?????????? ????-?????????

- Робот появляется в списке подключений и передаёт видео и телеметрию.
- Оператор управляет роботом с веб-интерфейса (ноутбук и телефон).
- Модель ИИ получает кадры и телеметрию и останавливает робота по правилам выше.
- Команда оператора перекрывает команду модели.
- Все данные сессии записаны в лог.

???? 2. ???????? «?????» ??? ????????????

- Развёртывание S3 и Time Series DB, запись видео и телеметрии в хранилища.
- Передача видео через RTSP/WebRTC для снижения задержки.
- Функциональная видимость роботов (RBAC), авторизация JWT.
- Мониторинг Prometheus + Grafana.
- Запуск пользовательских моделей в вычислительной среде (Docker, K8s).
- Тестирование на группе роботов на физическом полигоне.

???? 3. ??????? ? ????????????

См. Дорожная карта.

Развитие платформы

???????? ?????

Текущее ядро ARTEX — фундамент. Следующий уровень — открытая экосистема для соревнований и тиражирования моделей ИИ.

	Сейчас	Следующий шаг	Видение
Название	Платформа ARTEX	Полигон и соревнования	AI Arena — экосистема
Содержание	Подключение устройств; real-time видео; модели ИИ (YOLO, RL); REST API и SDK	Физический полигон; виртуальная симуляция; «Битва интеллектуальных роботов»; геймификация и рейтинги	Маркетплейс моделей ИИ; «Kaggle для роботов»; открытый API и SDK; площадка для ведомств

?????????????? ?????

Направление	Реализация
Backend	Серверная платформа Go / C++
Инфраструктура	Docker + Kubernetes
Подключение устройств	SDK `artex-robot` (Socket.IO), SSH для администрирования
Видео	Кадры через Socket, RTSP → WebRTC
Операторы	WebSocket real-time, Web UI (React)
API	REST API + gRPC
ИИ	YOLO, Assist, RL, пользовательские модели
Хранилище	S3 + Time Series DB + PostgreSQL
SDK	`pip install artex-robot`
Мониторинг	Prometheus + Grafana

?????? ARTEX

1. **Не один робот, а платформа** для любых автономных систем.
2. **Не симуляция, а реальные устройства** с видео и телеметрией в реальном времени (плюс виртуальный полигон для обучения).

3. **Не только управление, а обучение ИИ** — Imitation Learning на каждом действии оператора.
4. **Не закрытая система, а открытый SDK** — подключение устройства за минуты.
5. **Не проект на бумаге, а работающий прототип** с реальными устройствами.

????????

Руководитель проекта — Алексей Маслов, Telegram: @MaslovAlexey. Документация: docs.mephi-ai.ru.