



IROS Core

Руководство администратора системы

Конфигурирование и развёртывание

ООО «ИНТЕЛМЕТ ТЕХНОЛОДЖИС»

www.intelmet.ru

© ООО «ИНТЕЛМЕТ ТЕХНОЛОДЖИС»

Аннотация

В документе приведено описание конфигурирования и развёртывания мета-операционной системы IROS Core (IntelMet Robot Operating System) на операционной системе Astra Linux Special Edition.

IROS Core — программная платформа управления роботизированными и производственными комплексами. Платформа предоставляет ядро сервисов, конфигурацию узла, консоль управления, веб-интерфейс оператора и программный интерфейс REST.

Документ предназначен для администраторов системы, инженеров по внедрению и специалистов технической поддержки.

1. Что такое IROS Core

IROS Core (IntelMet Robot Operating System) — мета-операционная система управления роботизированными комплексами. Это каркас для создания распределённых приложений реального времени: аппаратная абстракция, контроль исполнительных устройств, обмен сообщениями, журналирование и набор расширяемых сервисов.

Платформа построена по модульному принципу. Модули реализуют стандартный интерфейс сервиса, включаются и заменяются через конфигурацию без пересборки ядра. Узлы образуют одноранговую сеть: регистрируются в IROS, обмениваются данными по сокетам IROS поверх TCP/IP и могут быть распределены по нескольким компьютерам.

Состав платформы:

- ядро и модель сервисов (старт, останов, цикл обработки);
- файл конфигурации узла `server.xml`;
- консоль управления по telnet и локальная консоль;
- веб-интерфейс оператора WebService;
- программный интерфейс RESTService для внешних систем и веб-интерфейса;
- сервисы оборудования, технического зрения, обмена и журналирования.

Клиентские библиотеки реализованы для C#, C++ и F#. Оператор работает через браузер. Интеграция со смежными системами выполняется по REST, сокетам и отраслевым протоколам (Modbus, MQTT, OPC).

2. Общие положения

2.1. Рабочая среда

IROS Core эксплуатируется на операционной системе Astra Linux Special Edition производства ГК «Астра», очередное обновление 1.8. Основной уровень защищённости — «Воронеж». Уровень «Смоленск» применяется по требованиям объекта. Редакция «Орёл» применяется там, где сертификат средства защиты не требуется.

Платформа развёртывается:

- на сервере под управлением Astra Linux Special Edition;
- на рабочей станции под управлением Astra Linux Special Edition;

- на промышленном компьютере комплекса под управлением Astra Linux Special Edition.

Операторский интерфейс не требует установки клиентского приложения: он открывается в браузере по адресу веб-сервиса. Данные интерфейса поступают из REST-интерфейса того же узла.

2.2. Свойства платформы

Горизонтальный перенос: один каркас используется на сервере цеха, на станции комплекса и на узле оборудования. Вертикальный перенос: микроядерная архитектура позволяет строить как локальный комплекс, так и систему уровня цеха.

Процесс описывается как переход от события к событию. Взаимодействие нескольких комплексов задаётся общими сигналами, очередями и сервисами обмена, без жёсткой привязки модулей друг к другу на этапе сборки.

3. Требования к аппаратуре и программному обеспечению

3.1. Операционная система

- Astra Linux Special Edition 1.8, уровень защищённости «Воронеж» — основная поддерживаемая ОС.
- Astra Linux Special Edition 1.8, уровень защищённости «Смоленск» — по требованиям объекта.
- Astra Linux Special Edition 1.8, уровень «Орёл» — для объектов без требований по сертифицированной защите.

Требуется архитектура x86-64 и подключённый расширенный репозиторий дистрибутива: в нём поставляется среда выполнения .NET 8.

3.2. Программное обеспечение

Назначение	Состав
Среда выполнения	.NET 8 (пакет dotnet-runtime-8.0 из расширенного репозитория Astra Linux SE 1.8)
Сборка на объекте	.NET 8 SDK (пакет dotnet-sdk-8.0), при необходимости пересборки узла
Веб-интерфейс	Браузер на рабочей станции оператора. Отдельная установка клиента не требуется
СУБД	СУБД PostgreSQL или иная СУБД, поддерживаемая в составе дистрибутива IROS. Выбор задаётся сервисом базы в конфигурации узла

3.3. Аппаратные требования

Роль	Требования
Узел IROS Core	Astra Linux SE 1.8 x86-64; 4 ГБ ОЗУ или более; от 500 МБ под данные платформы; от 200 МБ под базу
Сервер IROS Core	Astra Linux SE 1.8 x86-64; процессор 2 ГГц или выше; 4 ГБ ОЗУ или более; от 1 ГБ под данные; от 2 ГБ под базу
Рабочее место оператора	Браузер, сетевой доступ к порту веб-интерфейса узла

На сервере поддерживают не менее 100 МБ свободного дискового пространства. При интенсивной записи в базу предпочтителен массив RAID 10.

3.4. Каналы связи

Оконечные каналы могут быть непостоянными. Узел работает в цикле автономии и синхронизируется при восстановлении связи.

Соединение	Минимум	Назначение
Wi-Fi	20 Кбит/с; отсутствие связи не дольше 6 часов	Терминалы, справочники, объекты учёта
Мобильный канал	20 Кбит/с; время автономии не ограничено	Мобильные терминалы
Ethernet	64 Кбит/с	Узлы IROS, сервер базы, кластер

4. Порядок установки

Шаг	Действие	Результат
1	Подготовить Astra Linux SE 1.8 x86-64 и расширенный репозиторий	Доступна среда .NET 8
2	Установить среду выполнения	dotnet --info показывает .NET 8
3	Разместить файлы платформы	Каталог /opt/intelnet/iros с исполняемым файлом и server.xml
4	Настроить server.xml и конфигурацию WebService	Заданы адрес, сервисы, база, вкладки пульта
5	Запустить узел и открыть веб-интерфейс	Консоль отвечает, пульт открывается в браузере

5. Установка системы

5.1. Среда выполнения

```
sudo apt update
sudo apt install -y dotnet-runtime-8.0
dotnet --info
```

Для пересборки узла на объекте вместо среды выполнения устанавливают пакет dotnet-sdk-8.0.

5.2. Размещение файлов

Каталог развёртывания — /opt/intelnet/iros. В каталоге размещают исполняемый файл IROS, файл server.xml, сборки сервисов и каталоги данных. Рядом с исполняемым файлом обязательно присутствует server.xml.

Службу запускают от отдельной учётной записи iros:

```
sudo useradd --system --home /opt/intelnet/iros --shell /usr/sbin/nologin iros
sudo chown -R iros:iros /opt/intelnet/iros
```

5.3. Автозапуск

Постоянная работа оформляется модулем systemd /etc/systemd/system/iros.service:

```
[Unit]
```

```
Description=IROS Core
```

```
After=network.target
```

```
[Service]
```

```
User=iros
```

```
WorkingDirectory=/opt/intelmet/iros
```

```
ExecStart=/opt/intelmet/iros/IROS -m -c /opt/intelmet/iros/server.xml
```

```
Restart=on-failure
```

```
[Install]
```

```
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable --now iros
```

6. Запуск и консоль

6.1. Запуск

Ключ `-m` запускает консольный режим. Ключ `-c` задаёт путь к конфигурации. Без `-c` используется `server.xml` в текущем каталоге.

```
cd /opt/intelmet/iros
```

```
./IROS -m -c /opt/intelmet/iros/server.xml
```

Процесс занимает мьютекс `Global\IROS`. Повторный запуск завершается, пока работает первый экземпляр. Локальная консоль принимает команды до строки `q`.

6.2. Telnet-консоль

Сетевая консоль — основной способ управления промышленным компьютером.

```
telnet <адрес узла>
```

```
GET STATUS
```

`GET STATUS` возвращает температуру контроллера (`CPUTemp`), занятую память (`Memory`), время работы с последнего включения (`WorkingTime`) и идентификатор главного хоста (`Machine`).

7. Система обозначений

- Ключевые слова команд в руководстве даны заглавными буквами. Ввод допускает любой регистр.
- Между командой и параметром ставится пробел. Лишние пробелы и табуляции игнорируются.
- Команда завершается клавишей `Enter`.
- Пропущенный параметр, если после него есть другие, отделяется запятой.
- Числа десятичные, если не указано иное. Расстояния задаются в миллиметрах, единица измерения при вводе опускается.
- Строка передаётся в двойных кавычках.
- Имя переменной начинается с буквы и содержит буквы, цифры, точку и подчёркивание. Длина — до 15 символов.

8. Консольные команды

Команда	Назначение
GET STATUS	Состояние узла: CPUTemp, Memory, WorkingTime, Machine
GET MACHINES	Список активных хостов узла
SET MACHINE	Назначить текущий хост. Параметр — имя хоста
GET PROXY	Идентификатор хоста текущего контекста
GET FIELD	Значение свойства текущего хоста
EXEC	Запуск метода, программы или функции текущего хоста
COMPILE	Изменение программы метода или свойства без остановки узла
GET TASK	Выгрузка контекста текущего задания в указанный XML-файл
GET PROCESSES	Список выполняемых процессов
FORCE ALERT	Инициация исключительного события для проверки реакции
GET QUEUE	Текущая очередь документов обмена
GET SERVICE STATUS	Статус сервиса или всех сервисов, если идентификатор не задан
SAVE DATABASE	Перенос текущего журнала в архив

9. Программы

Программа — последовательность инструкций одного процесса. Имя содержит не более 15 символов и начинается с буквы. Различают программы управления комплексом (кинематика и исполнительные устройства) и программы производственного процесса (фоновый контроль сигналов).

Программы создают на C#, C++ или F#. Строка содержит номер шага, необязательную метку, инструкцию и комментарий после //. Метка — число от 1 до 9999 или имя до 15 символов с двоеточием. Модуль включают и заменяют через `server.xml`, если он предоставляет интерфейс сервиса.

10. Сервисы

Сервис — единица расширения платформы. Он описывается в `server.xml`: имя, тип, область действия и параметры. Глобальный сервис может стартовать вместе с узлом. В конфигурацию объекта включают только те сервисы, которые нужны этому комплексу. Ниже — состав платформы.

10.1. Ядро и данные

Сервис	Назначение
LoggingService	Журнал событий: консоль, база, подписчики. Команды Recent, Show, BackupLog
DatabaseService	Доступ к базам данных узла
SettingsService	Параметры процессов и хостов
RegistryService	Реестр зарегистрированных узлов
SaveVariablesService	Сохранение переменных узла между запусками
WatchService	Контроль работоспособности узла. Команда GetStatus
SnapshotService	Снимок состояния узла
ProcessingRequestService	Приём и исполнение консольных команд

Сервис	Назначение
UpdateService	Обновление состава файлов узла. Команда InitiateUpdate
EventAlertService	Журнал исключительных событий. Команды Alert, Confirm, GetAlerts, GetCatalog
SchedulerService	Расписание фоновых задач узла
DBTraceService	Трасса операций с базой
WebTraceService	Трасса веб-запросов узла

10.2. Доступ и обмен

Сервис	Назначение
RESTService	Программный интерфейс узла и конфигурактор. Порт по умолчанию 8887
WebService	Веб-интерфейс оператора. Данные берёт у RESTService. Порт по умолчанию 8888
TelnetServerService	Консоль управления по сети
ListenerService	Внешние подключения к узлу и его программам
DirectRequestListenerService	Прямые запросы к узлу без очереди
TCPServerService	Сервер TCP для смежных систем
ListenerTelegramService	Приём телеграмм по согласованному формату
FTPService	Файловый доступ к каталогам узла
FTPStreamService	Потоковая выдача файлов
MQTTServerService	Брокер MQTT на узле
MQTTClientService	Клиент MQTT к внешнему брокеру
WebhookService	Исходящие уведомления. Команды SendWebhook, Channels, Queue
SMTPService	Отправка почтовых уведомлений
SSHServerService	Доступ к узлу по SSH

10.3. Промышленные протоколы

Сервис	Назначение
PROFINETServerService	Узел PROFINET IO-Device. Контроллер видит станцию и обменивается циклическими входами и выходами
S7ServerService	Образ контроллера S7: области DB, M, I, Q, обмен PUT/GET
S7ConnectionService	Клиент обмена с контроллером S7
ModbusServerService	Сервер Modbus для смежного контроллера
ModbusClientService	Клиент Modbus к внешнему устройству
OPCService	Обмен по OPC
PLCClientService	Клиент программируемого контроллера
PingService	Карта сети и проверка доступности узлов. Команды AddDevice, RemoveDevice, CheckDevice, GetMap
EEIPService	Обмен Ethernet/IP

10.4. Видео, зрение и сцена

Сервис	Назначение
MJPEGServerService	Публикация видеопотока MJPEG для пульта
RTSPClientService	Приём RTSP. Команды StartGrab, StopGrab, PauseGrab
FFMPEGClientService	Клиент захвата видео. Команды StartGrab, StopGrab, GetStatus, ES
FFMPEGService	Управление процессами захвата и записи. Команды Record, StopRecord, GetStatus
FFMPEGWriterService	Запись ролика и титра. Команды Create, Close, WA
VideoWriterService	Запись видеофрагмента в файл
ImageRepositoryService	Хранилище кадров
ScanHubService	Сбор телеграмм сканеров и выдача одного подтверждённого кода
XYZService	Трёхмерная сцена объекта: единицы, зоны, точки. Команда Objects
Scene2DService	Двумерная сцена объекта
MeasureHubService	Сбор результатов измерений нескольких источников
ICVService	Запуск программ технического зрения

10.5. Сервисы оборудования объекта

Связь с конкретным прибором объекта оформляется отдельным сервисом соединения. Его включают, только если прибор стоит на этом комплексе. Типовые сервисы соединения: CoherentConnectionService, SensorikaISD5ConnectionService, LiDARLDCPCConnectionService, SickLMS10ConnectionService. Параметры — адрес прибора, порт и режим опроса — задаются в конфигурации сервиса и в свойствах конфигулятора.

10.6. PROFINETServerService

Сервис публикует узел как устройство PROFINET IO-Device. Инженерный контроллер видит станцию как модуль и ведёт с ней циклический обмен входами и выходами. Загрузку программы контроллера сервис не принимает: он отдаёт образ входов и принимает образ выходов.

В конфигураторе сервис появляется в секции «Сервисы». Свойства собраны в группу Base.

Параметр	По умолчанию	Назначение
Iface	пусто	Сетевой интерфейс, на котором слушаются кадры PROFINET. Список интерфейсов даёт команда ListIfaces
StationName	iros	Имя станции, которое видит контроллер
VendorId	1171	Идентификатор производителя устройства
Deviceld	1	Идентификатор устройства в проекте контроллера
InputSize	64	Размер образа входов, байт
OutputSize	64	Размер образа выходов, байт
Ip	пусто	Адрес станции в сети PROFINET
Mask	255.255.255.0	Маска сети
Gateway	0.0.0.0	Шлюз

Команды сервиса в конфигураторе и в консоли:

- Status — состояние станции;
- ListIfaces — список сетевых интерфейсов узла;

- ReadOutput — чтение образа выходов, параметры offset и length;
- WriteInput — запись образа входов, параметры offset и hex.

Порядок ввода: выбрать интерфейс командой ListIfaces, записать Iface, StationName, Ip, Mask и размеры образов, выполнить Status и убедиться, что контроллер видит станцию. Образ входов пишет машина или смежный сервис командой WriteInput. Образ выходов читается командой ReadOutput.

```
<add name="PROFINETServerService"
  type="Intelmet.SteelTrace.Machine.PROFINETServerService, PROFINETServerService.dll">
  <configuration>
    <StationName>iros</StationName>
    <VendorId>1171</VendorId>
    <DeviceId>1</DeviceId>
    <InputSize>64</InputSize>
    <OutputSize>64</OutputSize>
    <Ip>192.168.1.20</Ip>
    <Mask>255.255.255.0</Mask>
    <Gateway>192.168.1.1</Gateway>
  </configuration>
</add>
```

10.7. ScanHubService

Сервис принимает телеграммы сканеров, накапливает считывания и отдаёт один наиболее вероятный код. Формат телеграммы:

```
\01\02\03[Код],[CAM],[OBJECTSIGNAL]\04\05
```

OBJECTSIGNAL равен 0 или 1 и показывает наличие объекта перед сканером. Шаблон кода задаётся параметром Pattern: символы разделены запятыми, префикс V помечает вариативную позицию. CheckObjectSensor включает подтверждение по датчику объекта. Времена ожидания и хранения задаются в секундах.

10.8. FFMPEGService

FFMPEGService запускает процессы ffmpeg по именованным конфигурациям. Сервис используют для двух задач: записать поток камеры в файл и опубликовать живой кадр для пульта. Сам сервис кадры не хранит: он собирает командную строку ffmpeg, запускает процесс и следит за ним.

На узле должен быть установлен ffmpeg и доступен в пути поиска программ. На Astra Linux Special Edition пакет ставят из репозитория дистрибутива. Без ffmpeg конфигурация сервиса загружается, но процесс захвата не стартует.

Параметры сервиса

Параметр	Назначение
ShowLogLevel	Подробность диагностики. 0 — тихий режим. Меняется командой SetLogLevel
FFMPEGConfigs	Список конфигураций. Конфигурация — именованный набор аргументов ffmpeg
FFMPEGRTSPs	Список входящих RTSP-потоков: идентификатор, адрес и описание

Конфигурация потока

Элемент FFMPEGConfig описывает один запуск ffmpeg. Поля:

- **Id** — имя конфигурации. По нему конфигурацию выбирает команда записи и автозапуск.
- **Desc** — пояснение для администратора. На запуск не влияет.
- **AutoStart** — true запускает эту конфигурацию при старте сервиса. Для живого предпросмотра ставят true, для записи по событию — false.
- **Args** — командная строка ffmpeg без имени программы. Пробелы и переводы строк сжимаются в один пробел.

В строке Args допускаются подстановки в фигурных скобках. Сервис заменяет найденный ключ значением вызова. Если ключа нет, скобки остаются как есть.

Подстановка	Откуда берётся
{Id}	Идентификатор конфигурации
{FileName}	Имя файла из команды записи
{duration}	Длительность записи из команды, миллисекунды

Адрес камеры в Args задают целиком, включая учётные данные потока. Учётные данные в документ объекта не выносят открытым текстом сверх необходимости: их хранит файл конфигурации узла.

Два рабочих вида

Запись в файл. AutoStart оставляют false. В Args стоит шаблон: транспорт RTSP, вход камеры, кодек и выход {FileName}. Запись начинает команда Record с именем файла, идентификатором конфигурации и длительностью. Останавливает команда StopRecord по номеру сеанса.

Живой кадр для пульта. AutoStart ставят true. В Args поток камеры перекодировается в MJPEG и публикуется на локальном порту узла. Пульт и карточка потока берут этот адрес. Такая конфигурация работает постоянно, команду Record для неё не вызывают.

Команды

Команда	Параметры	Действие
GetStatus	нет	Состояние запущенных контроллеров: поток, файл, кадр
ShowProcess	нет	Список процессов ffmpeg на узле
ShowStdOut	idx	Вывод контроллера с указанным номером
Record	fileName, configId, duration, handleId	Начать запись по конфигурации configId в файл fileName
StopRecord	handleId, duration	Остановить запись сеанса handleId
ES	нет	Аварийный останов процессов сервиса
SetLogLevel	logLevel	Задать ShowLogLevel

duration в команде Record задаётся в миллисекундах. Значение 0 у автозапуска означает работу без ограничения времени. handleId — номер сеанса. Если передать 0, сервис назначает номер сам и возвращает его.

Порядок настройки

- Установить ffmpeg и проверить, что команда ffmpeg находится в пути.
- Добавить FFMPEGService в server.xml.

- Описать конфигурацию с уникальным Id. Для предпросмотра включить AutoStart, для записи по событию оставить false.
- В Args указать транспорт RTSP, адрес камеры, кодек и выход: файл с {FileName} либо локальный порт MJPEG.
- Применить файл через секцию Update конфигуратора и перезапустить сервис.
- Выполнить GetStatus и ShowProcess. Для предпросмотра открыть адрес порта в браузере. Для записи вызвать Record и проверить файл.

Пример конфигурации записи и живого кадра. Адрес камеры и учётные данные заменяют на значения объекта:

```
<add name="FFMPEGService"
  type="Intelmet.SteelTrace.Machine.FFMPEGService, FFMPEGService.dll">
  <configuration>
    <ShowLogLevel>0</ShowLogLevel>
    <FFMPEGConfigs>
      <FFMPEGConfig Id="REC" AutoStart="false" Desc="Запись камеры 1">
        <Args>-y -rtsp_transport tcp -t {duration} -i rtsp://user:password@192.168.1.50/stream -
an -c:v libx264 -preset fast "{FileName}"</Args>
      </FFMPEGConfig>
      <FFMPEGConfig Id="CAM1" AutoStart="true" Desc="Живой кадр камеры 1">
        <Args>-rtsp_transport tcp -i rtsp://user:password@192.168.1.50/stream -an -c:v mjpeg -q:v
5 -r 10 -listen 1 -f mjpeg http://0.0.0.0:9192/{Id}.mjpg</Args>
      </FFMPEGConfig>
    </FFMPEGConfigs>
  </configuration>
</add>
```

Карточка пульта типа stream ссылается на порт живого кадра. Кнопка на схеме с командой Record вызывает запись через RESTService, передавая fileName и configId.

10.9. Состав, переносимый на .NET 8

Ниже — сервисы платформы, которые переносятся с .NET Framework 4.5.2 на .NET 8 и входят в состав IROS Core. Сервис без оконного интерфейса переносится как проект net8.0. В конфигурацию объекта включают только нужные.

Уже разобранные выше сервисы здесь не повторяются: ядро, доступ, промышленные протоколы, видео и сцена, PROFINETServerService, FFMPEGService, ScanHubService.

Идентификация и учёт

Сервис	Назначение
IdentityService	Пользователи и роли узла. Справочник учётных записей
CredentialService	Хранение учётных данных подключений

Журналы

Сервис	Назначение
LogService	Базовый журнал сообщений узла
NLogService	Журнал через NLog
SerilogService	Журнал через Serilog
NetLogService	Сетевой журнал
TraceService	Трасса выполнения для разбора

Базы и запросы

Сервис	Назначение
DatabaseConnectionService	Подключение к СУБД, поддерживаемой в составе дистрибутива IROS
OracleService	Подключение к Oracle
QueryService	Именованные запросы к базе
QueryExService	Расширенные запросы к базе
TagsService	Теги и их значения
SequencesService	Счётчики и последовательности номеров
TelemetryService	Поток телеметрии узла
PerformanceCountersService	Счётчики производительности узла

Сообщения

Сервис	Назначение
TelegramService	Обмен телеграммами внутреннего формата
ClientTelegramService	Клиент отправки телеграмм
MaxService	Уведомления в мессенджер MAX
ConsoleService	Консольный канал команд

Зрение, маркировка и распознавание

Сервис	Назначение
ImageService	Работа с изображениями узла
LogoStoreService	Хранилище логотипов маркировки
DataMatrixDotService	Формирование и разбор Data Matrix
DMAPService	Карта данных маркировки
InigmaService	Распознавание кодов
InplantaService	Обмен с системой Inplanta
NeuroService	Вызов нейросетевого распознавания
OnnxService	Исполнение модели ONNX
LEDService	Управление светодиодной индикацией
LaserDescService	Описание и параметры лазерной маркировки
IntelScanService	Сервис сканирующего комплекса
ProcessorService	Обработчик потока данных зрения

Процессы и служебные

Сервис	Назначение
ProcessOrchestrationService	Оркестрация процессов комплекса
ParadigmService	Исполнение парадигмы процесса
Listener	Базовый слушатель подключений
NATService	Трансляция адресов для узла
UpdateRebootService	Перезапуск узла после обновления
UPSService	Состояние источника бесперебойного питания

Отраслевые

Сервис	Назначение
MMKCommService	Обмен с контуром ММК
MMKTrackingService	Слежение за объектом в контуре ММК
SlabTrackingService	Слежение за слябом
SlabTrackingService2	Слежение за слябом, вторая реализация
WebMachineService	Веб-доступ к машине комплекса
WebPrinterService	Веб-доступ к принтеру маркировки

Что на .NET 8 не переносится

- MSSQLCEService — не переносится на Astra Linux. На Astra базу ведут сервисы баз данных, поддерживаемые в составе дистрибутива IROS.
- RoborealService — не переносится на Astra Linux. На Astra зрение ведут ICVService, OnnxService и ScanHubService.
- Каталог TagsService - 2 — дубликат TagsService, отдельным сервисом не является.

Сервисы PROFINETServerService, S7ServerService и WebhookService входят в текущий состав IROS и переносятся на .NET 8 наравне с остальными. Их назначение и настройка приведены выше.

11. Конфигурирование узла

11.1. Файл server.xml

Конфигурация узла лежит рядом с исполняемым файлом и называется server.xml. Кодировка — UTF-8. Корневой элемент — configuration. Обязательные секции: Runtime, databaseCatalog, serviceSettings.

11.2. Секция Runtime

Задаёт адрес узла. Сервисы читают его атрибутом getFromRuntime.

```
<Runtime>
  <IP>
    <ServiceIP>192.168.1.10</ServiceIP>
  </IP>
</Runtime>
```

11.3. Секция databaseCatalog

Описывает базы узла. Имя базы используется сервисами журнала и настроек. Подключение к СУБД, поддерживаемой в составе дистрибутива IROS, задаётся в configuration соответствующего сервиса базы.

11.4. Секция serviceSettings

Список сервисов. name — имя в консоли и в ссылках. type — тип и сборка. configuration — параметры сервиса. Область scope: Global, Orchestration или Tasklet.

```
<serviceSettings>
```

```

<services>
  <add name="DatabaseService"
    type="Intelmet.SteelTrace.Machine.DatabaseService, Machine.Framework.dll"/>
  <add name="RESTService"
    type="Intelmet.SteelTrace.Machine.RESTService, RESTService.dll"/>
  <add name="WebService"
    type="Intelmet.SteelTrace.Machine.WebService, WebService.dll">
    <configuration>
      <Port>8888</Port>
      <Title>IROS</Title>
    </configuration>
  </add>
</services>
</serviceSettings>

```

12. Конфигуратор сервисов

Конфигуратор — веб-интерфейс настройки машин и сервисов узла. Он входит в состав RESTService и открывается в браузере по адресу `http://<адрес узла>:8887/`. Отдельная программа на рабочее место администратора не устанавливается.

Доступ закрыт ключом. Ключ задаётся свойством `AccessTokens` сервиса `RESTService` и передаётся в поле `Bearer`. Кнопка «Сохранить» запоминает ключ в браузере. Кнопка «Обновить» перечитывает каталог машин, сервисов и открытый объект.

Экран разделён на каталог слева и карточку объекта справа. В каталоге две секции: «Машины» и «Сервисы». Справа, после выбора объекта, открываются секции его настройки. Внизу страницы расположены секции обновления файла конфигурации, видео и кольца журнала. Они относятся ко всему узлу.

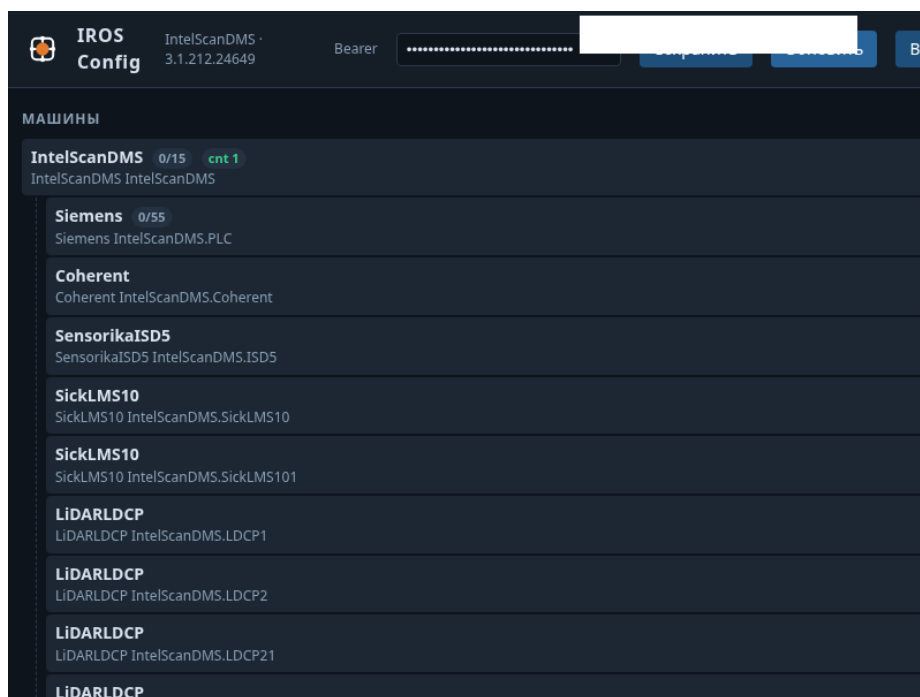


Рисунок 1. Конфигуратор сервисов: машины, сервисы, карточка объекта

12.1. Секция «Машины»

Секция показывает дерево машин узла. У каждой машины выводятся имя, тип, область (domain) и состояние. Если у машины есть подчинённые машины, они раскрываются уровнем ниже.

Выбор машины открывает её карточку в секции «Объект». Дальнейшая настройка выполняется уже в секциях карточки: свойства, команды, процессы, счётчики, регистры и конфигурация файла. Сама секция «Машины» состав не редактирует: состав задаётся файлом server.xml и загруженными сборками.

12.2. Секция «Сервисы»

Секция показывает сервисы, зарегистрированные на узле. У сервиса выводятся имя, тип, область и состояние. Выбор сервиса открывает его карточку в секции «Объект».

Сервис попадает в этот список, если он описан в serviceSettings файла server.xml и успешно стартовал. Сервис, которого нет в конфигурации, в конфигураторе не появляется. Порядок настройки нового сервиса: добавить его в server.xml, перезапустить узел, выбрать сервис в этой секции и заполнить его свойства.

12.3. Секция «Объект»

Секция открывается после выбора машины или сервиса. В заголовке — имя, состояние, вид объекта (машина или сервис), тип и область. Если объект входит в дерево, показываются родитель и подчинённые. Ниже заголовка идут секции настройки выбранного объекта. Пустая секция не скрывается: в ней выводится пометка, что элементов нет.

Свойства

В секцию попадают только открытые свойства и поля, помеченные как настраиваемые. У такого свойства есть описание, краткое имя (Alias) и группа (Group). Группа задаёт заголовок блока. Порядок групп задаётся номером группы.

В строке свойства выводятся краткое имя, описание, тип и текущее значение. Если свойство доступно для записи, значение редактируется в поле и записывается кнопкой «Записать». Запись применяется сразу, без перезапуска узла. Свойство только для чтения показывается, но кнопки записи у него нет.

Группы типичного сервиса зависят от самого сервиса. У RESTService это порт, адрес привязки, имя и список ключей доступа. У WebService — порт, адрес, заголовок пульта, имя REST-сервиса, запасной адрес и интервал опроса. У сервиса оборудования — адреса контроллеров, таймауты и режимы, которые автор сервиса вынес в настройку.

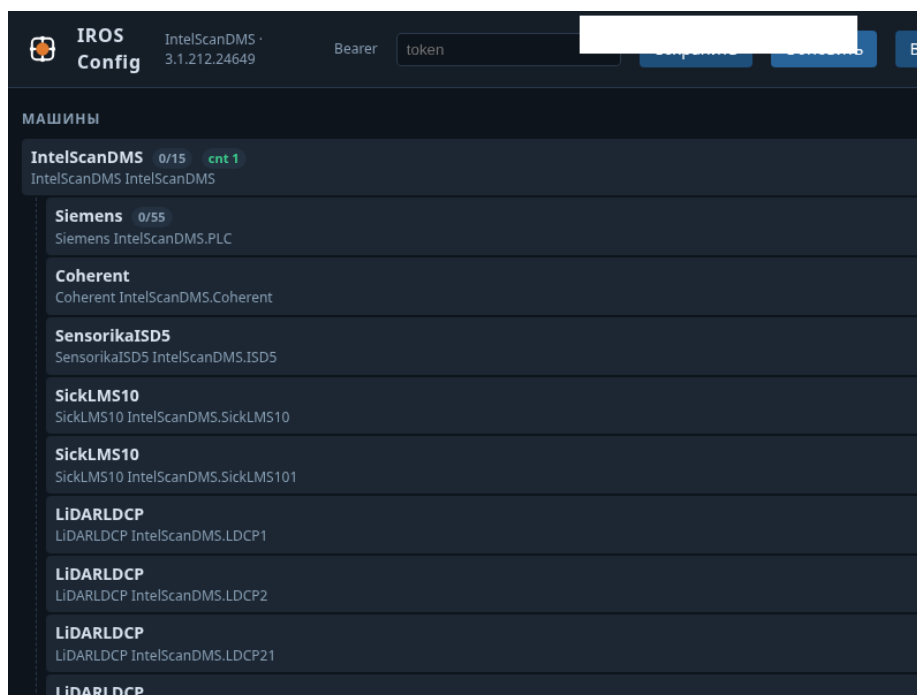


Рисунок 2. Свойства выбранного сервиса, сгруппированные по назначению

Поле свойства	Что задаёт администратор
Имя	Краткое имя из Alias. По нему свойство вызывается и из консоли
Описание	Назначение поля. Задаётся автором сервиса, в конфигураторе не меняется
Тип	Число, строка или логическое значение. Ввод проверяется по типу
Значение	Рабочее значение. Кнопка «Записать» сохраняет его в работающий сервис
Группа	Блок, в котором стоит свойство. Одна группа — одна тема настройки

Команды

Секция показывает методы объекта, помеченные как команды. У команды есть описание, краткое имя, группа и список параметров. Обязательный параметр отмечен звёздочкой. Команды собраны по группам в том же порядке, что и свойства.

Чтобы выполнить команду, администратор заполняет параметры и нажимает «Выполнить». Вызов уходит в работающий сервис и не требует правки файла. Результат выводится под кнопкой. Последние вызовы сохраняются в блоке недавних команд этой сессии и запускаются повторно без нового ввода.

Типовые команды конфигуратора: переподключение REST у WebService, снимок раскладки пульта, чтение памяти узла, запуск и останов процесса, обновление конфигурации из файла. Состав команд у каждого сервиса свой.

Процессы

Секция показывает процессы выбранной области: имя, идентификатор, тип, группу, состояние и время последней активности. В заголовке — сколько процессов выполняется из общего числа.

У процесса доступны действия «Start» и «Abort». Start запускает процесс, Abort прерывает выполнение. Действие применяется сразу. Если у области процессов нет, секция сообщает об этом.

Счётчики

Секция показывает счётчики области: имя, описание, значение, единицу измерения и состояние. Счётчик отражает накопленное или текущее измерение сервиса (число циклов, число считанных кодов, время операции). Значение в этой секции читается. Изменение уставки, если оно предусмотрено, выполняется через секцию свойств или команду сервиса.

Регистры

Секция показывает регистры области: имя, идентификатор, подсказку и значение. Значение регистра — признак включено или выключено. Регистр используют для дискретных сигналов оборудования и флагов сервиса. Установка регистра, если она разрешена сервисом, выполняется командой объекта, а не правкой файла.

Конфигурация файла

Секция показывает блок configuration выбранного сервиса из файла service.xml. Простые поля выводятся таблицей «поле — значение». Вложенные элементы (списки вкладок, карточек, точек) выводятся отдельной таблицей по атрибутам. Если блок пуст, секция сообщает, что секция configuration пустая. Рядом доступен исходный фрагмент XML.

Эта секция нужна, чтобы сверить, что файл содержит именно те параметры, с которыми сервис стартовал. Правка сложной вложенной структуры (вкладки и карточки пульта, списки адресов) выполняется в файле service.xml. После загрузки файла применяют команду Update, описанную ниже.

12.4. Секция «Update / service.xml»

Секция работает с файлом конфигурации узла целиком, а не с одним свойством.

Действие	Что происходит
Update	Узел перечитывает service.xml и применяет состав сервисов и их параметры. Текущий файл перед записью сохраняется как копия с суффиксом .bak
Скачать service.xml	Браузер получает текущий файл конфигурации узла для правки и хранения
Загрузить service.xml	Содержимое выбранного файла записывается как новый service.xml. Предыдущий файл сохраняется в копию

Порядок смены конфигурации объекта: скачать файл, внести секции сервисов, загрузить файл, выполнить Update, обновить каталог и проверить свойства выбранного сервиса. Ошибочный файл возвращают из копии .bak.

12.5. Секция «Видео»

Секция показывает потоки MJPEG, которые публикует сервис видеонаблюдения узла. У потока выводятся имя и адрес. Кнопка открытия запускает просмотр кадра в браузере. Если сервис видеонаблюдения на узле не настроен, секция сообщает, что потоков нет. Включение и адрес камеры задаются в свойствах сервиса видеонаблюдения, а не в этой секции.

12.6. Секция «Кольцо лога»

Секция показывает последние сообщения журнала узла, которые RESTService держит в кольцевом буфере. У записи выводятся время и текст. Число записей задаётся полем count. Секция используется при проверке записи свойства и выполнения команды: сообщение о применении появляется здесь без подключения к telnet.

12.7. Порядок настройки сервиса

- Открыть конфигуратор и ввести ключ Bearer.
- В секции «Сервисы» выбрать сервис. Если его нет — добавить описание в server.xml и перезапустить узел.
- В секции свойств заполнить группы: адреса, порты, режимы, интервалы. Каждое значение записать кнопкой «Записать».
- Проверить команды: выполнить команду без необратимого действия и убедиться, что ответ пришёл.
- В секции процессов убедиться, что нужные процессы в состоянии выполнения. Остановленный процесс запустить.
- Сверить секцию конфигурации файла с введёнными значениями.
- Для вложенных списков (вкладки пульта, карточки, таблицы адресов) править service.xml через секцию Update и применить Update.
- Повторить проверку в секциях свойств, процессов и кольца лога.

13. Конфигурирование через web

Веб-интерфейс оператора предоставляет сервис WebService. Он не хранит состояние комплекса сам: при старте находит RESTService того же узла, получает его адрес и дальше читает данные по HTTP. Раскладка пульта — вкладки и карточки — задаётся в конфигурации сервиса и применяется без пересборки.

Пульт открывается в браузере по адресу `http://<адрес узла>:8888/`. Порт по умолчанию — 8888. Отдельный клиент на рабочее место не устанавливается.

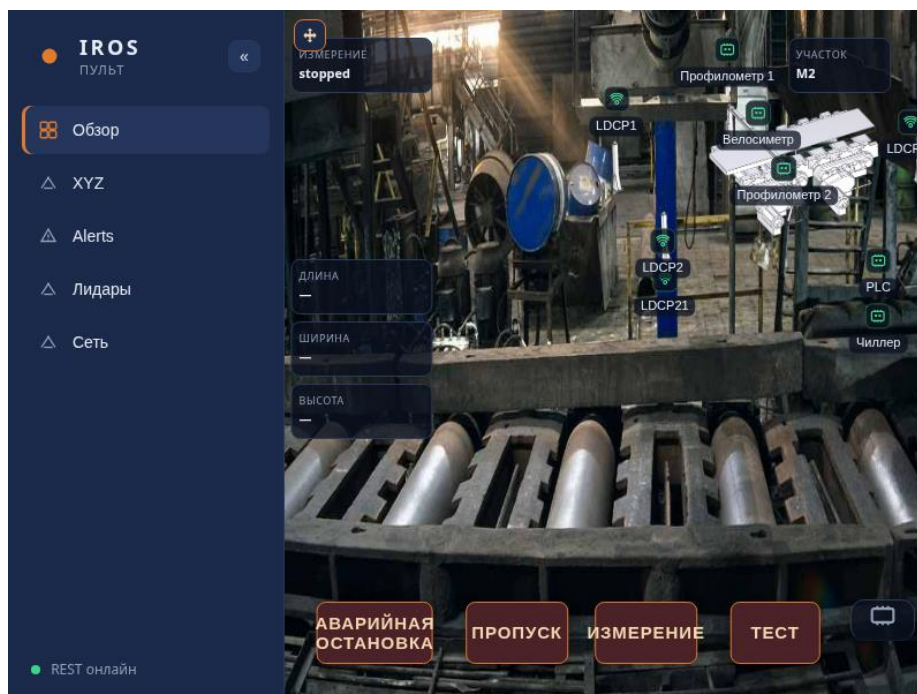


Рисунок 3. Пульт оператора. Вкладка со схемой объекта и пиктограммами оборудования

13.1. Параметры WebService

Параметр	По умолчанию	Назначение
Port	8888	HTTP-порт интерфейса

Параметр	По умолчанию	Назначение
Ip	пусто	Адрес привязки. Пустое значение — все интерфейсы
Title	IROS	Заголовок окна пульта
RestServiceName	RETSERVICE	Имя сервиса REST в каталоге узла
RestUrl	пусто	Запасной адрес REST, если сервис в процессе не найден
RestToken	пусто	Запасной ключ доступа к REST
PollSeconds	10	Интервал опроса карточек, секунды

Команды сервиса в консоли узла: BindRest — переподключить RESTService; Dashboard — снимок текущей раскладки; Memory — текущий объём памяти узла и процесса.

13.2. Вкладки

Вкладка описывается элементом View. Карточки привязываются к вкладке полем ViewId.

Поле	Назначение
Id	Идентификатор вкладки. На него ссылается ViewId карточки
Title	Подпись вкладки в пульте
Icon	Значок вкладки
Path	Дополнительный путь вкладки
Default	Вкладка открывается при входе в пульт

13.3. Конфигурация карточки

Карточка — элемент раскладки вкладки. Она описывается элементом Card внутри списка Cards конфигурации WebService. Поле ViewId связывает карточку с вкладкой. Поле Type задаёт вид карточки и набор полей, которые читает пульт.

Одна вкладка содержит одну или несколько карточек. Карточки вкладки выводятся сеткой. Ширина карточки в колонках сетки задаётся полем Cols. Пульт запрашивает снимок раскладки и строит экран только по нему: карточка, которой нет в конфигурации, на вкладке не появляется.

Поле Card	Назначение
ViewId	Идентификатор вкладки, на которой стоит карточка
Type	Вид карточки: markdown, picture-elements, stream, memory, alerts, xyz, button, network
Title	Заголовок карточки в пульте
Entity	Идентификатор объекта данных, если карточка привязана к сущности
Domain	Область сервиса или машины, из которой карточка читает состояние
Content	Текст карточки markdown
Icon	Значок карточки из встроенной коллекции
Image	Файл фона схемы для picture-elements. Файл лежит в каталоге данных узла
Fit	Вписывание фона: contain, cover или fill
Anchor	Точка привязки фона
Stream	Адрес видеопотока карточки stream
Channel	Номер канала потока. 0 — основной канал

Поле Card	Назначение
Cols	Ширина карточки в колонках сетки вкладки
PictureElements	Список пиктограмм и кнопок поверх схемы

Тип picture-elements показывает схему объекта. Фон берётся из Image через интерфейс файлов веб-сервиса. Поверх фона стоят элементы PictureElement. Положение Left и Top задаётся в процентах от ширины и высоты схемы, поэтому схема остаётся читаемой при изменении размера окна.

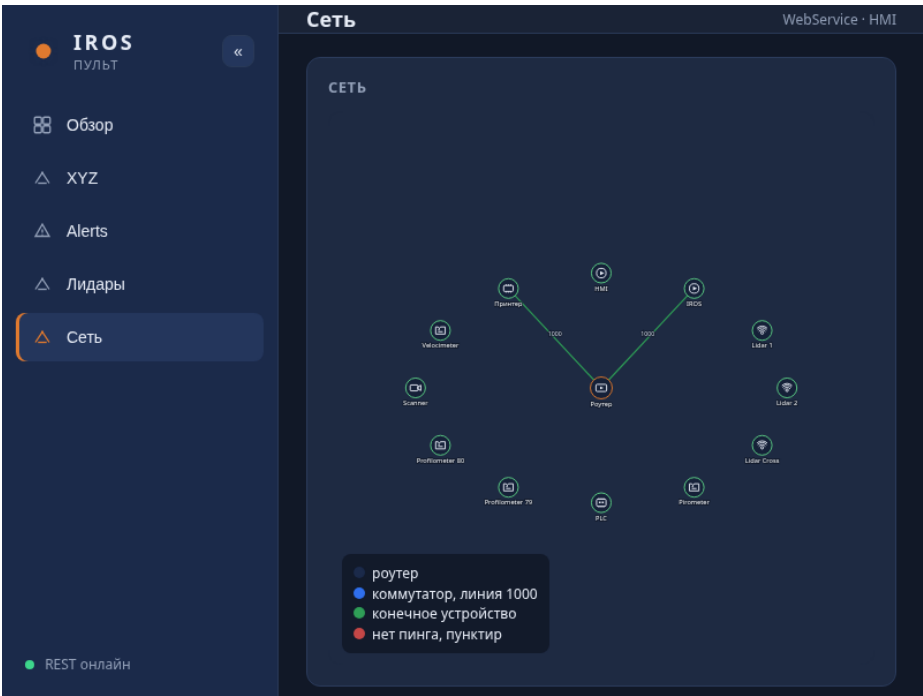


Рисунок 4. Карточка picture-elements: пиктограммы оборудования на схеме

Элемент схемы описывается вложением PictureElement.

Поле PictureElement	Назначение
Type	state-icon — пиктограмма состояния; button — кнопка команды
Icon	Значок: lidar, plc, alarm, camera, robot, measure, process, memory, stream, motor, state
Title	Подпись элемента на схеме
Left, Top	Положение в процентах от левого верхнего угла схемы
Width, Height	Размер элемента в процентах
Entity, Domain	Объект и область, чьё состояние показывает пиктограмма
Attribute	Поле состояния, которое читает пиктограмма
Command	Имя команды для типа button
Body	Тело запроса команды. По умолчанию пустой объект
Path, Format, Text, Image	Дополнительный путь, формат, подпись и собственное изображение
Layer	Слой отрисовки, если элементы перекрываются

Кнопка с типом button вызывает команду REST: область Domain и команда Command образуют вызов /api/services/{Domain}/{Command}. Тело берётся из Body. Так с схемы запускают поток, подтверждают событие или выполняют команду машины без отдельного клиента.

Остальные типы карточек:

- markdown — текст из Content, без обращения к оборудованию;
- stream — видеопоток канала Channel поверх изображения Image. При обрыве кадра пульт восстанавливает соединение;
- memory — занятость памяти узла и процесса. Данные приходят из интерфейса памяти веб-сервиса;
- alerts — журнал событий области Domain с кнопкой подтверждения;
- xyz — трёхмерная сцена объекта: единицы, зоны и точки из сервиса сцены;
- network — карта узлов сети по данным сервиса проверки связи.

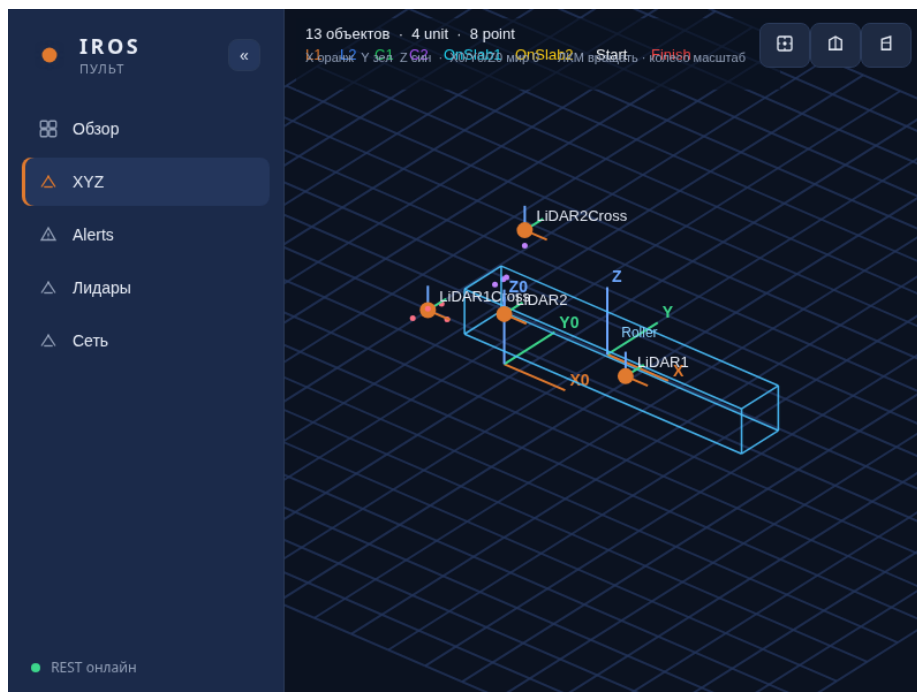


Рисунок 5. Карточка xyz: сцена объекта

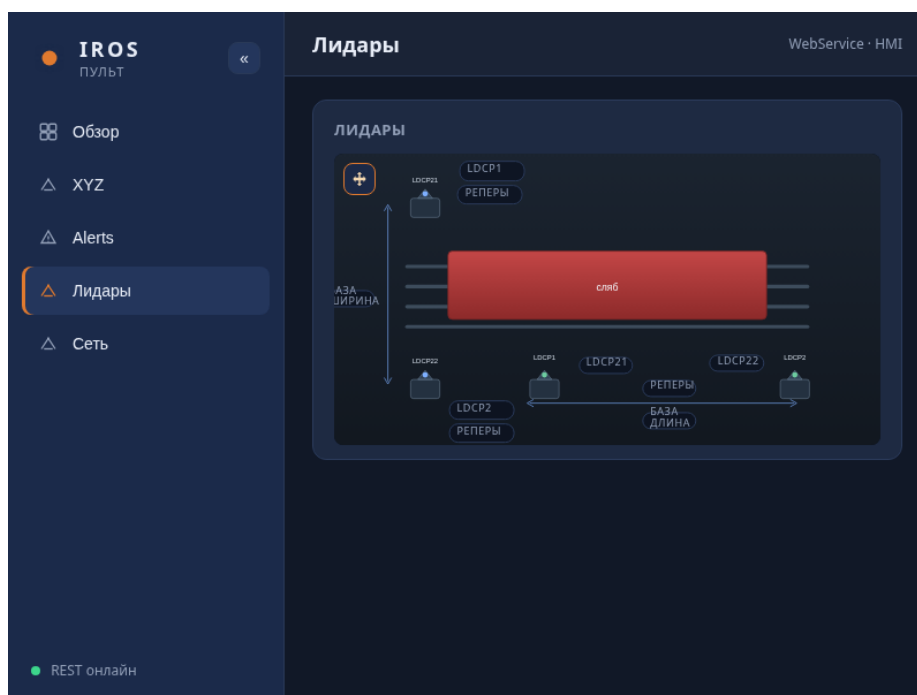


Рисунок 6. Карточка сети: узлы и состояние связи

Дата	Время	Длительность	Номер	Класс	Тип	Текст сообщения
13.10.26	12:59:59	24:52:42	10-04-001	10-04	Тревога	Ошибка соединения лидара LDSP1 x76536

Рисунок 7. Карточка журнала событий

Пример карточки схемы с пиктограммой и кнопкой:

```
<Cards>
  <Card ViewId="line" Type="picture-elements" Title="Участок"
    Image="plan.png" Fit="contain" Cols="1">
    <PictureElements>
      <PictureElement Type="state-icon" Icon="lidar"
        Left="22" Top="38" Title="Лидар 1"
        Domain="LiDAR" Entity="LD1"/>
      <PictureElement Type="button" Icon="process"
        Left="60" Top="55" Title="Пуск"
        Domain="Line" Command="Start" Body="{ }"/>
    </PictureElements>
  </Card>
  <Card ViewId="line" Type="alerts" Title="События"
    Domain="EventAlertService"/>
  <Card ViewId="scene" Type="xyz" Title="Сцена" Domain="XYZService"/>
</Cards>
```

Файл фона размещают в каталоге данных узла. Карточка запрашивает его через интерфейс файлов веб-сервиса. После правки списка карточек файл конфигурации применяют через секцию Update конфигулятора и открывают пульт заново.

13.4. Применение конфигурации

Изменение вкладок и карточек вносится в конфигурацию WebService и применяется перезапуском сервиса или узла. Пульт при открытии запрашивает снимок раскладки и строит вкладки по нему. Интервал обновления карточек задаёт PollSeconds. Сворачиваемая боковая панель пульта сохраняет выбранную вкладку на время сессии браузера.

14. Интерфейс WebService

WebService публикует HTTP-интерфейс пульта. Браузер обращается только к этому интерфейсу. Запросы состояния и команды, которые относятся к комплексу, веб-сервис

передаёт в RESTService и возвращает ответ браузеру. Прямое чтения баз и оборудования веб-сервис не выполняет.

Базовый адрес: `http://<адрес узла>:<Port>/`. По умолчанию `http://<адрес узла>:8888/`. Ответы данных передаются в JSON, кодировка UTF-8. Статические файлы интерфейса (страница, стили, сценарии, значки) отдаёт сам веб-сервис.

14.1. Методы интерфейса

Метод и путь	Назначение
GET /	Страница пульта
GET /api/health	Проверка доступности веб-сервиса и связи с REST
GET /api/dashboard	Снимок раскладки: заголовок, порт, интервал, вкладки, карточки, адрес REST
GET /api/memory	Память узла и рабочий набор процесса IROS
GET /api/alerts	Журнал событий. Параметр domain задаёт источник, по умолчанию EventAlertService
POST /api/alerts/confirm	Подтверждение событий журнала. Параметр domain — источник
GET /api/ftp	Файл схемы или изображения по параметру path
GET /api/xyz	Сцена объекта: единицы, зоны, точки. Параметр domain — источник сцены
GET /api/network	Карта сетевых узлов. Параметр domain — источник
GET, POST /api/rest/...	Передача запроса в RESTService. Путь после /api/rest совпадает с путём REST

14.2. Снимок раскладки

GET /api/dashboard возвращает объект с полями title, port, pollSeconds, rest, views и cards. Поле rest сообщает, найден ли RESTService, его имя и базовый адрес. Массив views — вкладки. Массив cards — карточки с типом, привязкой к вкладке и элементами схемы. Пульт строит экран только по этому снимку и не хранит раскладку у себя.

14.3. Передача в REST

Путь /api/rest/<путь> веб-сервис пересылает в RESTService как /<путь>, сохраняя метод и тело запроса. Так пульт вызывает команды комплекса, не зная адреса REST напрямую. Если RESTService не подключён, веб-сервис возвращает код 503 и сообщение о несвязанном REST.

Кнопка на схеме с типом button формирует вызов команды через этот же канал: REST-метод сервиса Domain и команда Command. Тело запроса кнопки задаётся полем Body элемента, по умолчанию пустой объект.

14.4. Видео и схемы

Карточка потока запрашивает изображение канала через веб-сервис. Канал 0 — основной поток. При обрыве кадра интерфейс восстанавливает соединение без перезагрузки страницы. Страница отдельного потока доступна по пути /stream.html.

Файл фона схемы запрашивается методом GET /api/ftp с параметром path. Веб-сервис отдаёт файл из каталога данных узла и выставляет тип содержимого по расширению. Отсутствующий файл возвращается с кодом 404.

14.5. Журнал и подтверждение

GET /api/alerts возвращает журнал источника, указанного параметром domain. POST /api/alerts/confirm подтверждает события этого источника. Подтверждение фиксируется в журнале узла через RESTService. Карточка типа alerts на пульте использует эти два метода.

14.6. Проверка интерфейса

- Страница `http://<адрес узла>:8888/` открывается и показывает заголовок из параметра Title.
- GET /api/health возвращает успешный статус.
- GET /api/dashboard содержит вкладки и карточки из конфигурации.
- Поле `rest.found` равно `true`: RESTService найден и связан.
- Карточка схемы загружает файл фона, пиктограммы стоят в заданных процентах.
- Кнопка на схеме выполняет команду и не требует отдельного клиента.

15. Проверка после развёртывания

- На узле доступна среда выполнения .NET 8.
- Процесс IROS запущен от учётной записи iros, повторный запуск не создаёт второй экземпляр.
- Telnet-консоль принимает GET STATUS и возвращает Machine, Memory и WorkingTime.
- GET SERVICE STATUS показывает сервисы из server.xml, включая RESTService и WebService.
- Конфигуратор открывается по порту RESTService, ключ Bearer принимается, выбранный сервис показывает свойства и команды.
- Веб-интерфейс открывается по настроенному порту, снимок раскладки совпадает с конфигурацией.
- Журнал узла пишется в настроенную базу. Архив журнала снимается командой SAVE DATABASE или средствами СУБД.

Резервируют базу журналов и каталог /opt/intelmet/iros, включая server.xml. Файл открытой базы копируют штатным резервированием СУБД, а не как занятый файл. Свободное место контролируют и под данные, и под копии.