Документация на ПО «ПАМИР»

[1. Определения и аббревиатуры 6](#_Toc1)

[2. Инструкция по установке 9](#_Toc2)

[Поддерживаемые операционные системы 9](#_Toc3)

[3. Инструкция пользователя 14](#_Toc4)

[3.1. Введение 14](#_Toc5)

[3.1.1. СРМ 14](#_Toc6)

[3.1.1.1. ТКЕ 15](#_Toc7)

[3.1.1.2. КЕ 15](#_Toc8)

[3.1.1.3. Связи 16](#_Toc9)

[3.1.1.4. Атрибут 16](#_Toc10)

[3.1.1.5. Правила наименования 17](#_Toc11)

[3.1.1.6. Правила идентификации 17](#_Toc12)

[Разрешение конфликтов 17](#_Toc13)

[3.1.1.7. Мониторинг 18](#_Toc14)

[Основные настройки Prometheus 18](#_Toc15)

[Настройки экспортеров 18](#_Toc16)

[Настройки заданий (jobs) 18](#_Toc17)

[Настройки целей (targets) 18](#_Toc18)

[3.1.1.7.1. Экспортеры 19](#_Toc19)

[Blackbox 19](#_Toc20)

[SNMP 19](#_Toc21)

[SQL 19](#_Toc22)

[3.1.1.8. Метрики 21](#_Toc23)

[Шаблон цели 21](#_Toc24)

[3.1.1.9. Индикаторы здоровья 22](#_Toc25)

[3.1.1.9.1. Индикатор метрики 22](#_Toc26)

[3.1.1.9.2. Расчетный индикатор 24](#_Toc27)

[3.2. Функции ПО 24](#_Toc28)

[3.2.1. Дашборды 24](#_Toc29)

[3.2.1.1. Управление дашбордами с главной страницы 24](#_Toc30)

[3.2.1.2. Управление дашбордами со страницы списка дашбордов 25](#_Toc31)

[3.2.1.3. Окно создания дашборда 26](#_Toc32)

[3.2.1.4. Панель инструментов дашборда 27](#_Toc33)

[3.2.1.5. Управление виджетами 30](#_Toc34)

[3.2.1.5.1. Создание виджета 30](#_Toc35)

[3.2.1.5.2. Изменение виджета 31](#_Toc36)

[3.2.1.5.3. Окно создания виджета 31](#_Toc37)

[Источники данных: 36](#_Toc38)

[3.2.1.5.4. Виджет «Лист тревог» 38](#_Toc39)

[3.2.1.5.5. Виджеты «Диаграмма с областями» и «График» 39](#_Toc40)

[3.2.1.5.6. Виджет «Карточка с атрибутами» 41](#_Toc41)

[3.2.1.5.7. Виджет «Дерево компонентов» 42](#_Toc42)

[3.2.1.5.8. Виджет «Таблица» 45](#_Toc43)

[3.2.1.5.9. Виджет Спидометр 49](#_Toc44)

[3.2.1.5.10. Виджет «Новая метрика» 50](#_Toc45)

[3.2.1.6. Механика построения дашбордов 52](#_Toc46)

[3.2.2. СРМ 53](#_Toc47)

[3.2.2.1. Фильтрация СРМ 53](#_Toc48)

[3.2.2.2. Построение TQL 53](#_Toc49)

[Фильтрация по ТКЕ 54](#_Toc50)

[Фильтрация по значению атрибутов 56](#_Toc51)

[Связанные ТКЕ 57](#_Toc52)

[Мощность связи 59](#_Toc53)

[3.2.2.3. Навигация по графу СРМ 62](#_Toc54)

[3.2.3. Моделирование СРМ 78](#_Toc55)

[3.2.3.1. Создание атрибутов 78](#_Toc56)

[3.2.3.2. Настройка ТКЕ 80](#_Toc57)

[3.2.3.2.1. Создание ТКЕ 80](#_Toc58)

[3.2.3.2.2. Привязка атрибутов 80](#_Toc59)

[3.2.3.3. Реестр индикаторов здоровья 82](#_Toc60)

[3.2.3.4. Привязка метрик 84](#_Toc61)

[3.2.3.5. Привязка индикаторов здоровья 88](#_Toc62)

[3.2.4. Наполнение СРМ 94](#_Toc63)

[3.2.4.1. Мануальное создание КЕ 94](#_Toc64)

[3.2.4.2. Импорт КЕ 94](#_Toc65)

[Импорт КЕ из файла в формате csv 95](#_Toc66)

[Импорт графа из файла в формате json 95](#_Toc67)

[3.2.4.3. Добавление КЕ через обнаружение 97](#_Toc68)

[3.2.4.3.1. Настройка атрибутов 105](#_Toc69)

[Фетчеры 105](#_Toc70)

[Парсеры 106](#_Toc71)

[3.2.4.3.2. Настройка дополнительных связей 107](#_Toc72)

[Фетчеры 107](#_Toc73)

[Парсеры 107](#_Toc74)

[3.2.4.3.3. Запуск обнаружения 107](#_Toc75)

[Через планировщик 107](#_Toc76)

[Мануально из СРМ 108](#_Toc77)

[3.2.5. Управление доступом 109](#_Toc78)

[Алгоритм применения правил 110](#_Toc79)

[Фильтрация пользователей 112](#_Toc80)

[3.2.6. Управление пользователями 112](#_Toc81)

[3.2.6.1. Алгоритм аутентификации и авторизации 112](#_Toc82)

[Аутентификация с указанным доменом 113](#_Toc83)

[Аутентификация без указания домена 113](#_Toc84)

[3.2.6.2. Управление локальными пользователями 114](#_Toc85)

[Реестр пользователей 114](#_Toc86)

[Добавление пользователя 115](#_Toc87)

[Изменение пользователя 117](#_Toc88)

[Создание дополнительных полей пользователя 118](#_Toc89)

[3.2.6.3. Настройка LDAP 119](#_Toc90)

[Основные понятия LDAP 119](#_Toc91)

[Настройка LDAP 120](#_Toc92)

[3.2.7. Отправка уведомлений 123](#_Toc93)

[3.2.7.1. Мои уведомления 123](#_Toc94)

[3.2.7.2. Внешние сервисы 126](#_Toc95)

[3.2.7.3. Шаблоны сообщений 129](#_Toc96)

# Определения и аббревиатуры

|  |  |
| --- | --- |
| **Базовое понятие, аббревиатура** | **Описание** |
| ИТ | Информационные технологии |
| ПО | Программное обеспечение |
| СРМ, модель | Сервисно-ресурсная модель (модель) – это логическая модель сервиса, описывающая состав и взаимосвязи КЕ (ресурсов), которые совместно обеспечивают предоставление сервиса на согласованном уровне. |
| КЕ | Конфигурационная единица. Это ресурс определенного типа. Конфигурационные единицы в составе СРМ отражают реальные информационные системы и сервисы, их модули, компоненты ИТ-инфраструктуры, документацию и т.п. |
| ТКЕ, Тип КЕ | Тип определяет принадлежность КЕ к ресурсу определенного вида. КЕ с аналогичными свойствами сгруппированы в ТКЕ. |
| БД | База данных |
| БДКЕ | База данных конфигурационных единиц. Англоязычный термин – CMDB (Configuration Management Data Base). |
| СУБД | Система управления базой данных |
| Регулярные выражения (regular expressions) | Формальный язык математических выражений для работы с текстом, для поиска и осуществления манипуляций с подстроками, основанный на использовании метасимволов. Для поиска используется строка-образец (pattern, шаблон, маска), состоящая из символов и метасимволов и задающая правило поиска. |
| Фетчер | Механизм обеспечивающий получение данных с объекта контроля |
| Парсер | Механизм обеспечивающий разбор текстовый конфигурации по определенному правилу |
| Парсинг | Синтаксический разбор и анализ данных с использованием парсера |
| Шаблонизатор | ПО, позволяющее использовать шаблоны для генерации конечных документов |
| Перцентиль | Величина, которая характеризует попадание элементов массива в статистически заданный диапазон, т.е. это неравенство, для которого N элементов массива меньше или равны заданному числу. |
| ФОИВ | Федеральный орган исполнительной власти |
| Индикатор здоровья | Детализированные измерения, отражающие состояние КЕ на основе конкретной рабочей характеристики. HI – обобщенное название для Индикатора доступности и Индикатора производительности. |
| Индикатор метрики | Индикатор, задающий правило обработки метрик и формирующий правила оповещения ([alerting-rules](https://prometheus.io/docs/prometheus/latest/configuration/alerting_rules/#alerting-rules)). |
| Расчетный индикатор | Индикатор, описывающий правило расчета индикатора здоровья на основе связанных ТКЕ. |
| AD | Active Directory, служба каталогов. |
| LDAP | Lightweight Directory Access Protocol, «легковесный протокол доступа к каталогам») –  протокол прикладного уровня, который обеспечивает доступ к службе каталогов посредством протокола TCP/IP.  Позволяет производить операции аутентификации (bind), поиска (search) и сравнения (compare), а также операции добавления, изменения или удаления записей. |
| FQDN | Fully Qualified Domain Name, полное определенное (квалифицированное) имя домена |
| SSL/TLS | SSL (Secure Sockets Layer; уровень защищённых cокетов) – криптографический протокол для безопасной связи. В настоящее время используется TLS (Transport Layer Security), протокол безопасности транспортного уровня, реализующий асимметричные алгоритмы шифрования |
| TQL | Topology Query Language (TQL) - это язык запросов для описания топологии графа. Он позволяет описывать различные свойства графов, такие как количество вершин и ребер, наличие циклов, связность и т.д. TQL является мощным инструментом для анализа и визуализации СРМ. |
| Alerting rules (Правила оповещения) | Правила оповещения определяют условия для оповещения на основе выражений Prometheus expression и отправляют уведомления о срабатывании оповещений во внешнюю службу. |
| ABAC (Attribute-Based Access Control: | Разграничение (контроль) доступа на основе атрибутов   – модель контроля доступа к объектам, основанная на анализе правил для атрибутов объектов или субъектов, при которой действительные права доступа предоставляются пользователям посредством использования политик, которые объединяют атрибуты вместе. |
| MAC (Mandatory Access Control) | **Мандатное управление** доступом – разграничение доступа субъектов к объектам, основанное на назначении метки конфиденциальности для информации, содержащейся в объектах, и выдаче официальных разрешений (допуска) субъектам на обращение к информации соответствующего уровня конфиденциальности. |
| DAC (Discretionary Access Control) | **Избирательное управление** доступом – управление доступом субъектов к объектам на основе списков управления доступом или матрицы доступа |
| ACL (Access Control List) | Список  управления доступом. |
| Дашборд | Интерактивная аналитическая панель и графический интерфейс c расположенными в пределах экрана (или фрейма) требуемыми ключевыми метриками, показателями целей или процессов. С помощью метрик можно выявлять и анализировать тренды. |

# Инструкция по установке

Для установки приложения ПАМИР на систему пользователя должны быть соблюдены ряд технических требований.

###### Поддерживаемые операционные системы

Debian-based, дистрибутив с ядром Linux не ниже версии 10, включая:

* Ubuntu не ниже версии 20.04;
* Astra Linux Special Edition версии 1.7;
* Debian не ниже версии 10.13;

В операционной системе должны быть установлены пакеты:

* Docker версии не ниже 20.10;
* Docker Compose версии не ниже 2.10.0 (Docker Compose идет в комплекте с дистрибутивом приложения ПАМИР и будет установлен, если только в системе уже не существует docker compose нужной версии).

###### Содержимое дистрибутива

В ходе установки будут развернуты Docker контейнеры:

- Prometheus и Alertmanager (система мониторинга);

- PostgreSQL (СУБД SQL);

- Redis (NoSQL СУБД);

- RabbitMQ (брокер сообщений);

- OpenLDAP (сервер для протокола LDAP);

- Nginx (HTTP-сервер и обратный прокси-сервер);

- Victoria Metrics (СУБД для Prometheus);

- Prometheus MQ (конвертация общения с Prometheus в очередь сообщений);

- Fluent-bit (обработчик лог-сообщений);

- ряд Exporter (экспортеры метрик для Prometheus);

Приложение ПАМИР использует порты (при необходимости могут быть переопределены в docker-compose):

- 80, 443, 8001, 8002, 8004 (порты, использующиеся непосредственно приложением);

- 5432 (PostgreSQL);

- 5672, 15672 (RabbitMQ);

- 6379 (Redis);

- 389, 636 (OpenLDAP);

- 9090 (Prometheus);

- 9093 (Prometheus Alertmanager);

- 8428 (Victoria Metrics);

- 24224 (Fluent-bit);

- 8005 (IPS Analyzer).

Дистрибутив приложения ПАМИР реализован в виде образа формата iso. Для установки ПАМИР данный образ необходимо смонтировать любым доступным способом (через системные/сторонние приложения). В корне приложения расположен скрипт install.sh, который произведет непосредственную установку из CLI (command line interface). Для установки необходимо вызвать ./install.sh install и следовать указаниям инсталлятора. Скрипт автоматически произведет все необходимые проверки и загрузит все требуемые образы в docker, после чего запустит их средствами docker-compose.

###### Пошаговая установка

1. Монтирование образа iso.

sudo mount -o loop ./pamir-40010872-40010872-1693310661-astra1.7.iso /media/cdrom



1. Запуск скрипта установки.

cd /media/cdrom/install

./install.sh install

В случае наличия всех требуемых компонентов и соблюдения всех требований к системе вы увидите перед собой сообщение об успешной установке (см. Рисунок 1. Запуск скрипта установки ). В ходе установки скрипт при необходимости установит на вашу систему нужную версию docker-compose, если она отсутствует на ней. Сразу после установки приложение будет запущено по указанному адресу. Логи по умолчанию находятся в папке .pamir вашей родительской директории, например /home/user/.pamir.

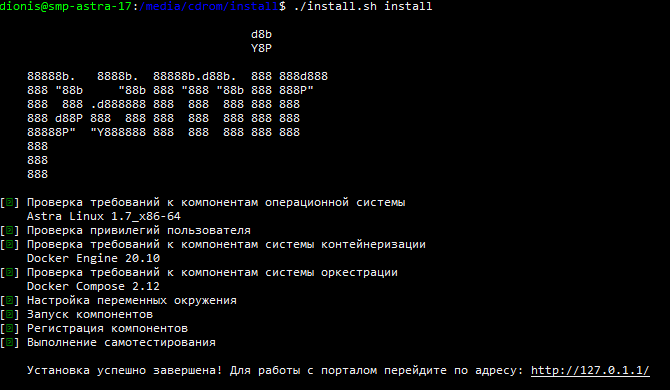


Рисунок 1. Запуск скрипта установки

1. Опции инсталлятора.

Скрипт может принимать на вход ряд ключей (см. Рисунок 2. Опции инсталлятора ):

* install – установка приложения;
* uninstall – удаление приложения;

И ряд флагов:

* -h показ помощи и выход;
* -d показ отладочной информации;
* -q не интерактивный режим (напр., адрес сервера устанавливается автоматически);
* -p удаление развернутых сервисов (не использующихся docker контейнеров, томов, образов и пр.);

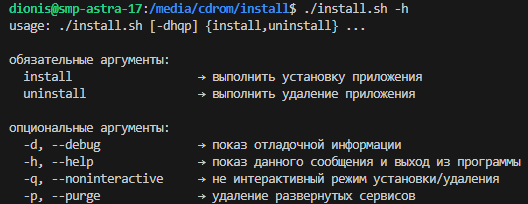


Рисунок 2. Опции инсталлятора

1. Варианты ошибок.

Не установлен docker нужной версии (см.Рисунок 3. Проверка требований ):

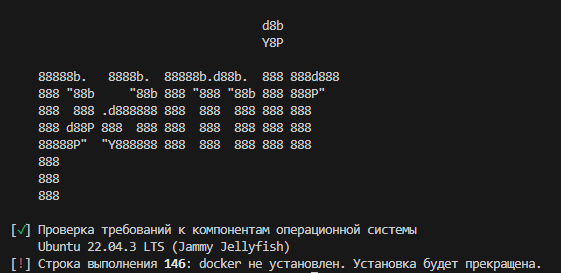


Рисунок 3. Проверка требований

Решение: установить docker версии не ниже 20.10.

1. Остановка приложения

Для остановки необходимо запустить скрипт с нужным ключом (см. Рисунок 4. Остановка приложения )

./install.sh uninstall

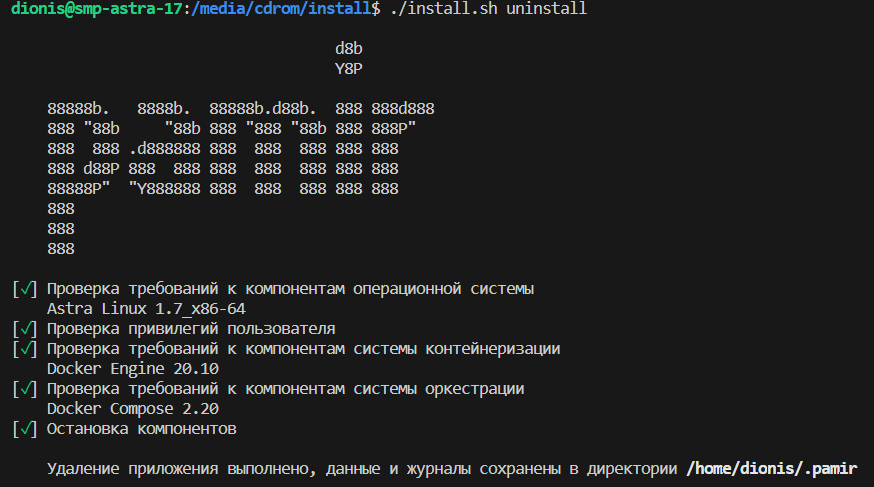


Рисунок 4. Остановка приложения

Приложение будет остановлено, данные сохранены в папке .pamir родительской директории. Образы, томы и подсети контейнеров docker не удаляются. Заново запустить приложение можно через

./install.sh install

1. Удаление приложения

Для удаления необходимо запустить скрипт с нужным ключом и флагом

./install.sh -p uninstall

Приложение будет остановлено, все логи и данные стерты, контейнеры удалены с вашего устройства (см. Рисунок 5. Удаление приложения ).

Внимание! На версиях docker выше 23.0 удаления томов может быть не произведено из-за особенностей нового поведения docker. В таких случаях убрать ненужные тома приложения можно самостоятельно через команду:

docker volume rm $(docker volume ls -q --filter="label=com.docker.compose.project=pamir")

В противном случае данные с томов от предыдущего запуска приложения будут перенесены на новый запуск, в том числе содержимое базы данных.

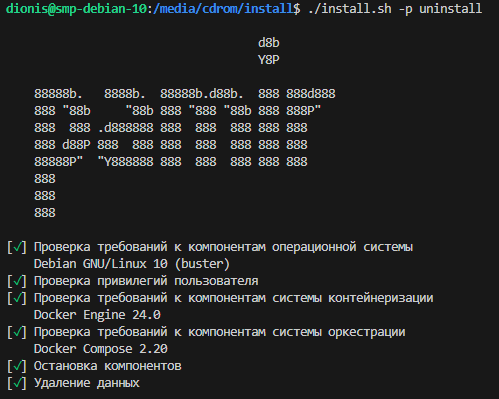
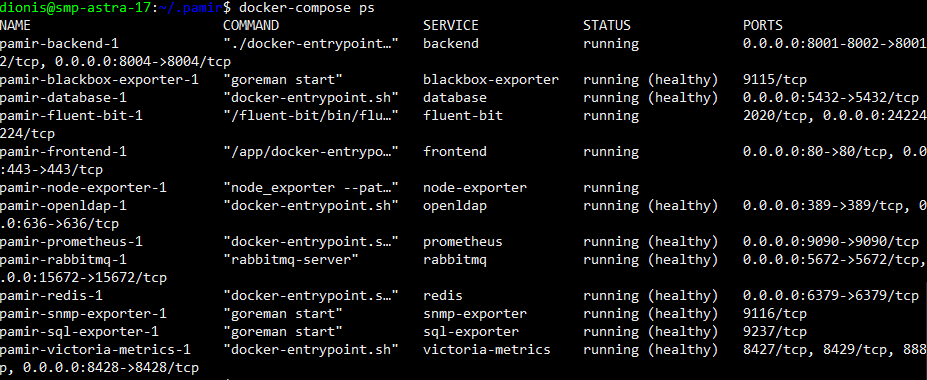


Рисунок 5. Удаление приложения

1. При необходимости большего управления над процессом установки или в случае отсутствия дистрибутива, запуск и остановка приложения могут осуществляться через docker-compose. Для этого необходимо сначала перейти в рабочую папку приложения (по умолчанию /home/\*user\*/.pamir).

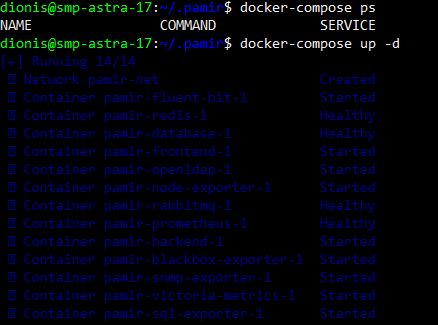
1. Далее, можно получить список всех функционирующих контейнеров приложения:

docker-compose ps



1. Запуск приложения с созданием контейнеров:

docker-compose up –d



1. Остановка приложения с удалением контейнеров и сетей:

docker-compose up –d



1. Запуск приложения:

docker-compose start



1. Остановка приложения с сохранением связанных контейнеров, томов, сетей :

docker-compose stop



По умолчанию в приложении существует учетная запись суперпользователя с логин/пароль admin/admin. При первом входе новый пользователь будет переадресован на окно смены пароля.

# Инструкция пользователя

## Введение

ПАМИР – система проактивного мониторинга информационных ресурсов. ПАМИР реализует концепции:

* представление объектов короля в виде сервисно-ресурсной модели;
* автоматическая постановка объектов короля на мониторинг;
* расчет влияния объектов контроля друг на друга и на предоставляемый сервис;
* разграничение доступа к объектам контроля и информации мониторинга на основе правил ABAC;
* кастомизируемые дашборды, интегрированные с сервисно-ресурсной моделью.

#### СРМ

**Сервисно-ресурсная модель (СРМ)** – это логическая модель сервиса, описывающая состав и взаимосвязи конфигурационных единиц (ресурсов), которые совместно обеспечивают предоставление сервиса на согласованном уровне. СРМ представляет собой ориентированный граф. Уровень декомпозиции (то есть, глубина детализации СРМ) зависит от строения предметной области и требований по ее контролю. Узлы СРМ называются конфигурационными единицами (КЕ). Связи между узлами могут также называться отношениями.

СРМ является ядром ПАМИР. Функции мониторинга, визуализации, управление доступом тесно связаны с СРМ.

Мониторинг использует СРМ для:

* настройки и привязки шаблонов мониторинга;
* расчета влияния и распространения статусов индикаторов здоровья по СРМ.

Визуализация использует СРМ для:

* ограничения видимости объектов дашборда.

Управление доступом использует СРМ для описания правил сопоставления атрибутов пользователя и атрибутов объектов контроля.

#### ТКЕ

Каждая конфигурационная единица СРМ принадлежит к определенному типу. **Тип конфигурационной единицы (ТКЕ)** определяет и структурирует данные, которые может хранить КЕ. На уровне ТКЕ определяются потенциальные связи между КЕ. Реальные связи между КЕ могут быть построены, только если существует потенциальная связь между соответствующими ТКЕ.

Можно провести аналогию с ООП. Класс объекта это ТКЕ. Экземпляр класса это КЕ.

ТКЕ так же как классы в ООП могут наследоваться. При наследовании дочерний ТКЕ сохраняет поведение родительского.

Наследуются:

* атрибуты;
* связи.

Поведение унаследованного свойства можно переопределить.

#### КЕ

**Конфигурационная единица (КЕ)** – объект предметной области, состояние которого нужно контролировать. КЕ может быть как материальным, так и абстрактным объектом. Например, сервер, виртуальная машина, жесткий диск, база данных, SAAS сервис, VPN туннель.

КЕ является экземпляром ТКЕ. КЕ хранит данные в атрибутах, определенных на ТКЕ.

#### Связи

Связи определяют влияние КЕ друг на друга. Иными словами, они описывают отношения между КЕ. Например, база данных установлена на операционной системе, сервер находится в стойке, ФОИВ используется защищенный VPN туннель.

Отношения, сформулированные на понятном человеку языке, можно систематизировать и обобщить. В СРМ выделяется 3 типа связей:

* композиция;
* агрегация;
* ассоциация.

**Композиция** – строгий тип отношений «Состоит из». Используется для описания отношений между целым и частью, где целое и часть не могут быть разделены. При удалении целого части тоже перестают существовать.

Примеры:

* комната – стены;
* База данных – таблица.

**Агрегация** – свободный тип отношений «Содержит» – используется для описания отношений между целым и частью, где целое и часть могут быть разделены.

Примеры:

* серверная стойка – сервер;
* группа сервисов – сервис.

**Ассоциация** – свободный тип отношений «Использует» – используется для описания связей между различными типами КЕ, при которых один КЕ использует функциональные возможности другого.

Примеры:

* клиент – сервер;
* компьютер – принтер.

Каждая конкретная связь между двумя КЕ является однонаправленной. Между двумя КЕ можно построить более одного типа связей, но не более одной связи каждого типа.

#### Атрибут

**Атрибут** – строительный блок ТКЕ. Атрибут позволяет хранить типизированные данные.

Типы данных атрибутов:

* строка;
* число;
* дата и время;
* логический;
* список строк;
* JSONB;
* файл.

Атрибут создается и хранится в реестре атрибутов. При создании должны быть указаны обязательные поля атрибута:

* название;
* описание;
* тип данных.

После добавления в реестр атрибуты становятся доступными для использования в ТКЕ.

#### Правила наименования

Правила наименования определяют название, которые будет использовать КЕ при отображении.

Правило наименования записывается в виде шаблона в текстовом виде. В качестве шаблонизатора применяется Jinja2. Переменные шаблона должны быть записаны в квадратных скобках [[ ]].

В шаблонизаторе доступны переменные:

* attributes – атрибуты, доступные на ТКЕ.

Во всех используемых в приложении шаблонизаторах, для переменных, доступны кастомные фильтры:

* set\_if\_not\_value – установить указанное значение при отсутствии значения у переменной, аргумент: значение которое нужно использовать;

'[[attributes.name|set\_if\_not\_value("значение которое нужно использовать")]]'

* skip\_if\_not\_value – пропустить переменную при отсутствии у переменной значения (подставит пустую строку), аргументов нет;

'[[attributes.name|skip\_if\_not\_value]]'

* datetimeformat – значение переменной в формате iso даты преобразовать в указанный формат, аргумент: строка с форматом вывода даты (в виде кодов C стандарта), по умолчанию используется "%d %B %Y %H:%M";

'[[attributes.date|datetimeformat]]'

'[[attributes.date|datetimeformat("%d.%B.%Y")]]'

При использовании в шаблонах переменных без значения будет возникать ошибка, последствия которой зависят от контекста в котором применяется шаблонизатор (пропущено какое-либо действие, возврат пользователю уведомления о ошибке и пр.)

#### Правила идентификации

Правила идентификации позволяют избежать дублирования при создании КЕ. Правило идентификации записывает условие уникальности для экземпляров ТКЕ. Оно в логической форме оперирует атрибутами ТКЕ. При создании и изменении КЕ правило вычисляется для существующих в СРМ КЕ. Если правило находит КЕ в СРМ, то они считаются идентичными и объединяются либо системой, либо пользователем.

Связь Композиция оказывает влияние на идентификацию. Если КЕ является композируемым, то идентификация проводится только среди его сверстников в рамках композирующего КЕ. Если КЕ не является композируемым, то идентификация выполняется на всем перечне однотипных КЕ, существующих в СРМ.

###### Разрешение конфликтов

Если в результате идентификации в СРМ найдено более одного КЕ, то эти КЕ считаются дублирующимися. СРМ будет избавляться от дублирования, объединяя несколько КЕ в один.

Алгоритм объединения:

* из списка дублирующихся КЕ выбирается наиболее старый. В дальнейшем только он останется в системе, остальные будут удалены;
* в выбранном КЕ обновляются атрибуты, собранные со всех КЕ;
  + если атрибут присутствует одновременно в нескольких КЕ, то из них сохраняется наиболее свежий;
* на выбранный КЕ переносятся связи со всех КЕ.

#### Мониторинг

ПАМИР осуществляет мониторинг средствами Prometheus. ПАМИР позволяет конфигурировать Prometheus и экспортеры через единую консоль. Конфигурация применяется автоматически, без необходимости перезагрузки.

###### Основные настройки Prometheus

Основные настройки Prometheus выполняются в yaml формате в меню Настройки > Мониторинг > Prometheus.

Файл конфигурации можно просматривать и редактировать.

Редактировать можно все, кроме разделов:

* rule\_files:
* scrape\_configs:

rule\_files генерируется автоматически на основе правил индикаторов здоровья.

scrape\_configs генерируется автоматически на основе метрик, привязанных к ТКЕ.

###### Настройки экспортеров

Настройки экспортеров выполняются в меню Настройки > Мониторинг > Задания и экспортеры.

###### Настройки заданий (jobs)

Настройки задний выполняются в меню Настройки > Мониторинг > Задания и экспортеры.

Настройка заданий выполняется в соответствии с правилами <scrape\_config>.

###### Настройки целей (targets)

Список целей для каждого задания формируется автоматически на основе метрик, связанных с ТКЕ.

Для формирования целей в конфигурацию задания подставляется http\_sd\_config, по которому Prometheus получает актуальный список целей.

Статические настройки целей и service discovery могут быть указаны в настройках заданий.

##### Экспортеры

###### Blackbox

Prometheus Blackbox exporter.

Экспортер настроен на получение конфигурации через Redis, функционирующий в рамках ПАМИР.

Использует ключ - /exporter/blackbox

###### SNMP

Prometheus SNMP exporter.

Экспортер настроен на получение конфигурации через Redis, функционирующий в рамках ПАМИР.

Использует ключ - /exporter/snmp

Особенности - настройки аутентификации могут быть указаны в качестве URL параметров. Snmp exporter функционирующий в рамках ПАМИР не должен принимать внешние вызовы с параметрами аутентификации, указанными в URL, это не безопасно.

Параметры аутентификации, доступные через URL:

* version;
* community;
* security\_level;
* username;
* password;
* auth\_protocol;
* priv\_protocol;
* priv\_password;
* context\_mame.

###### SQL

Экспортер настроен на получение конфигурации через Redis, функционирующий в рамках ПАМИР.

Использует ключ - /exporter/sql

Поддерживаются следующие базы данных:

* Postgres;
* ClickHouse;
* AWS Athena;
* MS-SQL;
* MySQL;
* Snowflake;
* Vertica;
* Materialize.

При написании запросов для данного экспортера необходимо иметь в виду, что модель данных Prometheus присваивает метрике ровно одно значение с плавающей точкой, возможно, дополнительно идентифицируемое набором нолей или меток (лейблов). Эти метки должны иметь типы string или text.

---

# jobs is a map of jobs, define any number but please keep the connection usage on

# the DBs in mind jobs (карта заданий, определяет любое их количество, при этом нужно

# учитывать использование соединения в базах данных):

# each job needs a unique name, it's used for logging and as an default label

# (каждое задание для логирования должно иметь уникальное имя)

- name: "example"

# interval defined the pause between the runs of this job

# (интервал между заданиями)

interval: '5m'

# cron\_schedule when to execute the job in the standard CRON syntax

# (расписание cron в стандартном cron-синтаксисе)

# if specified, the interval is ignored

# (если задан, то интервал игнорируется)

cron\_schedule: "0 0 \* \* \*"

# connections is an array of connection URLs (массив соединений URL)

# each query will be executed on each connection

# (каждый запрос будет выполнен на своем соединении)

connections:

- 'postgres://postgres@localhost/postgres?sslmode=disable'

# startup\_sql is an array of SQL statements (startup\_sql - это массив SQL-инструкций)

# each statements is executed once after connecting

# (каждая инструкция выполняется единожды после подключения)

startup\_sql:

- 'SET lock\_timeout = 1000'

- 'SET idle\_in\_transaction\_session\_timeout = 100'

# queries is a map of Metric/Query mappings

# (запросы - это карта сопоставлений метрик/запросов)

queries:

# name is prefied with sql\_ and used as the metric name

# (имя имеет префикс sql\_ и используется в качестве имени метрики)

- name: "running\_queries"

# help is a requirement of the Prometheus default registry, currently not

# used by the Prometheus server. Important: Must be the same for all metrics

# with the same name! справка является требованием реестра Prometheus по умолчанию,

# в настоящее время не используется сервером Prometheus.

# Важно: Должно быть одинаковым для всех показателей с тем же именем!

help: "Number of running queries"

# Labels is an array of columns which will be used as additional labels.

# Must be the same for all metrics with the same name!

# All labels columns should be of type text, varchar or string

# Метки - это массив столбцов, которые будут использоваться в качестве

# дополнительных меток. Должно быть одинаковым для всех показателей с одинаковым

# именем! Все столбцы меток должны иметь типы text, varchar или string

labels:

- "datname"

- "usename"

# Values is an array of columns used as metric values. All values should be

# of type float Значения - это массив столбцов, используемых в качестве

# значений метрики. Все значения должны быть типа float

values:

- "count"

# Query is the SQL query that is run unalterted on the each of the connections

# for this job Запрос - это SQL-запрос, который выполняется без изменений при каждом

# из подключений для этой работы

query: |

SELECT datname::text, usename::text, COUNT(\*)::float AS count

FROM pg\_stat\_activity GROUP BY datname, usename;

# Consider the query failed if it returns zero rows

allow\_zero\_rows: false

#### Метрики

**Метрика** – контролируемая величина объекта мониторинга. Например, загрузка процессора, утилизация памяти.

Для сбора метрик используется Prometheus.

Для правильной привязки к СРМ метрика должна быть корректно сконфигурирована.

###### Шаблон цели

Шаблон цели служит для формирования prometheus targets для выбранного задания (job). Шаблон описывается в yaml формате. В качестве шаблонизатора применяется Jinja2. Переменные шаблона должны быть записаны в квадратных скобках [[ ]].

В шаблонизаторе доступны переменные:

* fields – системные поля. Их количество определено и не может изменяться;
  + ci\_id – идентификатор КЕ;
  + ci\_name[[1]](#footnote-2) – наименование КЕ;
  + ci\_type\_id – идентификатор ТКЕ;
  + metric\_id – идентификатор метрики;
* attributes – атрибуты, доступные на ТКЕ.

Поля ci\_id, ci\_type\_id, metric\_id обязательны для заполнения.

Пример:

- labels:

ci\_id: [[fields.ci\_id]]

ci\_name: '[[fields.ci\_name]]'

ci\_type\_id: [[fields.ci\_type\_id]]

metric\_id: [[fields.metric\_id]]

targets:

- '[[attributes.ssh\_host]]'

#### Индикаторы здоровья

**Индикатор здоровья** – качественное состояние объекта контроля с точки зрения определенной характеристики.

Статусы индикаторов здоровья (статусы расположены в порядке возрастания критичности):

* N/A – нет данных;
* Normal – нормальное состояние;
* Warning – предупреждение;
* Error – ошибка;
* Critical – критическая ошибка.

Примеры индикатора здоровья:

* загрузка процессора;
* утилизация сетевого интерфейса;
* доступность.

Типы индикаторов здоровья:

**Индикатор метрики** – индикатор, рассчитывающий свое состояние на основе метрики. Описывается в виде alert rule prometheus.

**Расчетный индикатор** – индикатор, рассчитывающий свое состояние на основе индикаторов здоровья на смежных КЕ. Расчетным индикатором можно реализовать восходящее и нисходящее влияние на смежные КЕ.

##### Индикатор метрики

Индикатор метрики оценивает измеряемую величину и сопоставляет ее с одним из статусов.

Для оценки используется alert rule prometheus.

- for: 1m

expr: probe\_success{} == 0

labels:

severity: 4

annotations:

summary: Устройство {{ $labels.ci\_name }} недоступно

description: Устройство {{ $labels.ci\_name }} недоступно

Правила индикаторов метрик должны быть написаны на yaml в виде alert rule prometheus. Правила поддерживают шаблонизацию. В качестве шаблонизатора применяется Jinja2.

В шаблонизаторе доступны переменные:

* fields – системные поля. Их количество определено и не может изменяться;
  + ci\_id – идентификатор КЕ;
  + ci\_name[[2]](#footnote-3) – наименование КЕ;
  + ci\_type\_id – идентификатор ТКЕ;
  + metric\_id – идентификатор метрики;
* attributes – атрибуты, доступные на ТКЕ.

Переменные шаблона должны быть записаны в квадратных скобках [[ ]].

Критичность

Критичность в правиле указывается с помощью метки severity. Критичность должна быть указана в виде кода критичности (целого числа от 1 до 5):

* 1 – N/A – нет данных;
* 2 – Normal – нормальное состояние;
* 3 – Warning – предупреждение;
* 4 – Error – ошибка;
* 5 – Critical – критическая ошибка.

Эскалация

Правила индикаторов метрик поддерживают эскалацию. Для этого критичность должна быть записана в следующем виде.

<код критичности> —<интервал времени>-> код критичности —<интервал времени>-> Код критичности ...

Интервал времени должен быть указан в виде целого числа и единицы измерения. Поддерживаемые единицы измерения:

* секунды – sec;
* минуты – min;
* часы – hour;
* дни – day.

Интервалы времени должны быть записаны одним числом и не могут быть комбинированы. Например, интервал в полтора часа должен быть записан как 90min, а не как 1hour30min.

Пример:

severity: 3 —15min-> 4 —15min-> 5

##### Расчетный индикатор

Расчетный индикатор позволяет рассчитывать статус на основе индикаторов смежных КЕ.

Правила расчета:

* по лучшему – принимает наименее критичное значение;
* по худшему – принимает наиболее критичное значение;
* по среднему – рассчитывает среднее значение критичности;
* перцентиль – принимает наиболее критичное значение из указанной в процентном отношении (от 0 до 100%) группы КЕ.

## Функции ПО

### Дашборды

Дашборды отображают данные мониторинга в привязке к СРМ. Дашборды доступны на главной странице ПАМИР после авторизации пользователя.

Каждый дашборд ограничен областью видимости КЕ. Сделано это путем ассоциации с сохраненным TQL.

Дашборды могут быть вложенными.

Каждый дашборд открывается в отдельной вкладке.

#### Управление дашбордами с главной страницы

Дашборды отображаются на главной странице. Каждый дашборд отображается в отдельной вкладке (см. Рисунок 6. Элементы управления вкладками дашбордов).

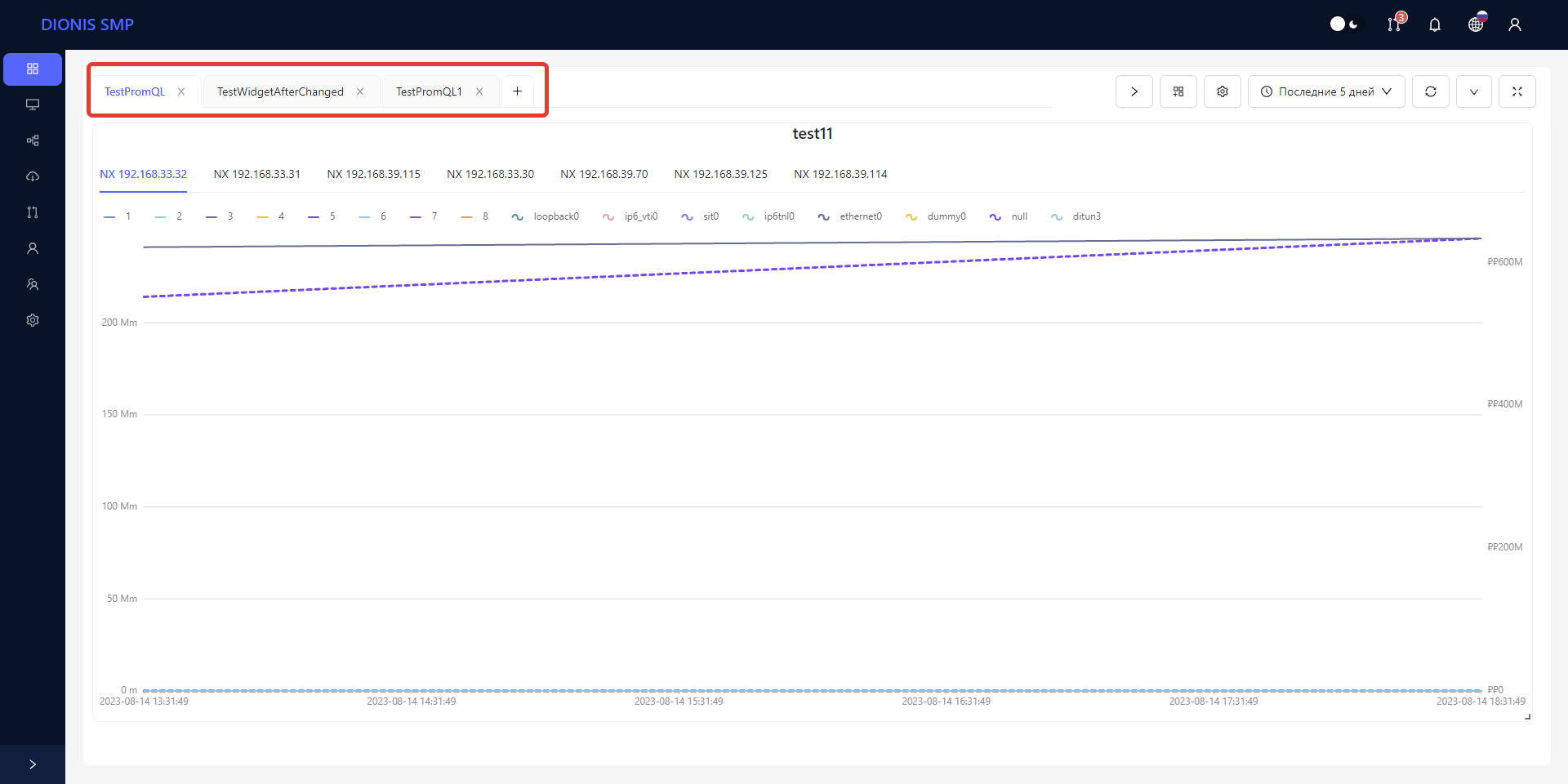


Рисунок 6. Элементы управления вкладками дашбордов

Открытые вкладки дашбордов сохраняются в привязке к каждому пользователю. После входа в систему пользователь увидит ранее открытые дашборды.

Для добавления дашборда необходимо кликнуть на плюс в панели вкладок. После клика откроется модальное окно с возможностью выбора существующего или созданием нового дашборда.

Для закрытия вкладки с дашбордом необходимо нажать на перекрестие вкладки. Сам дашборд при этом не удаляется. Его можно впоследствии восстановить из списка существующих дашбордов.

#### Управление дашбордами со страницы списка дашбордов

Список дашбордов показан на примере (см. Рисунок 7. Страница список дашбордов ).

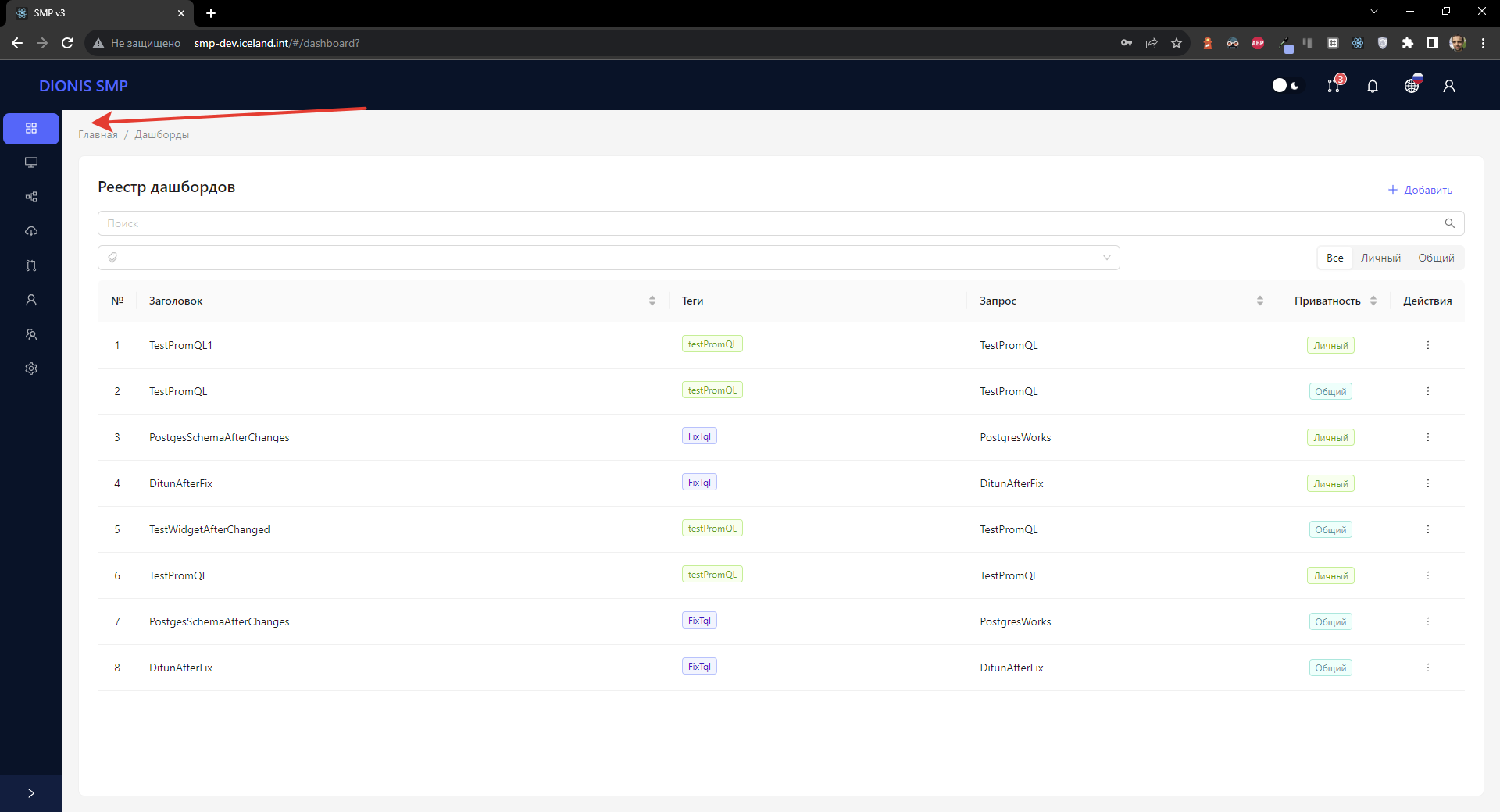


Рисунок 7. Страница список дашбордов

Для поиска дашбордов предусмотрены различные критерии (см. Рисунок 8. Поиск дашбордов ):

1. поиск по названию – фильтрует дашборды по заголовку;
2. поиск по тегу – фильтрует дашборды, оставляя только указанные теги;
3. фильтрация по уровню доступа – фильтрует дашборды, оставляя только указанный уровень доступа;
4. сортировки – сортирует выбранный столбец таблицы по возрастанию/убыванию значений.

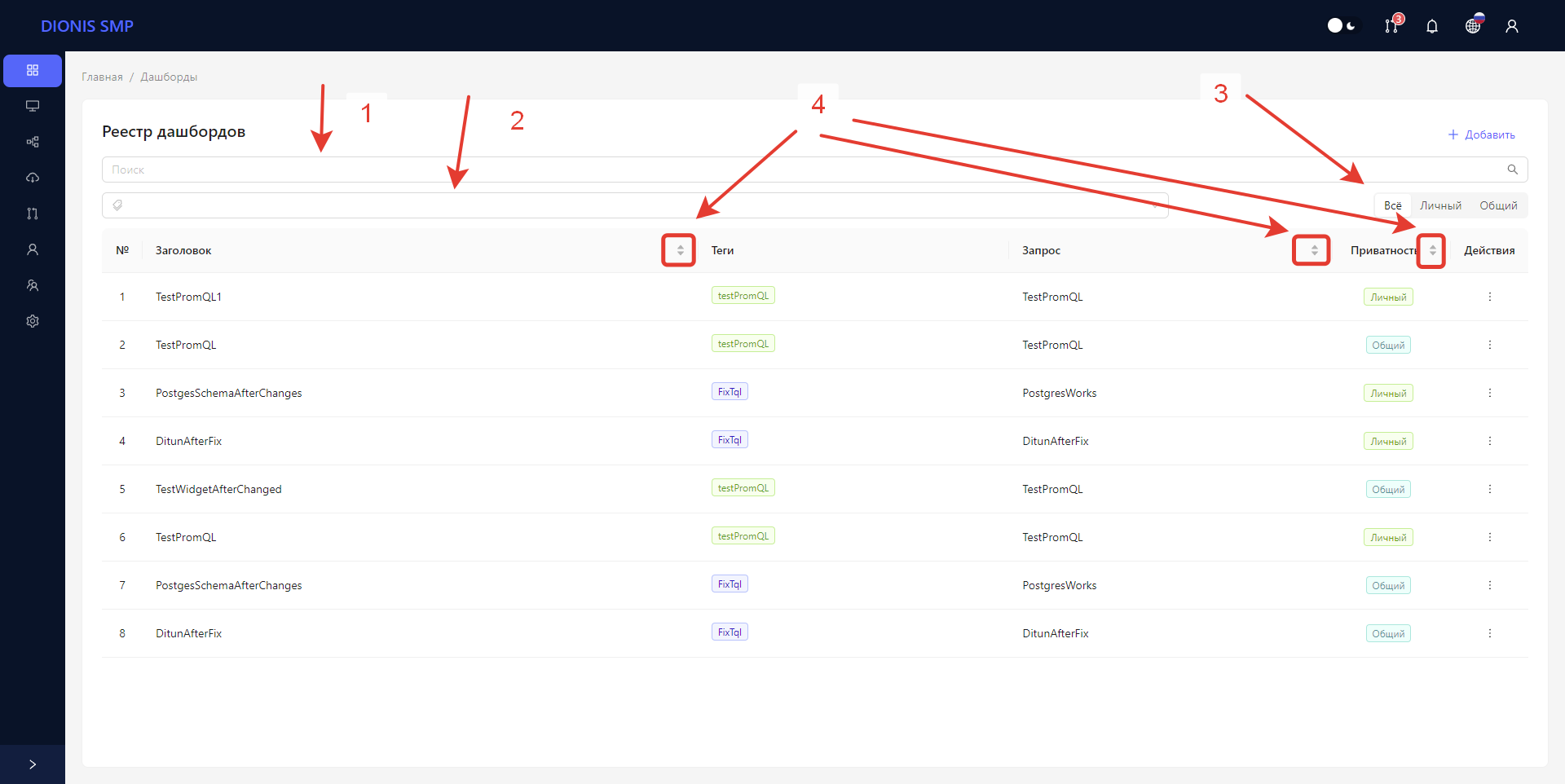


Рисунок 8. Поиск дашбордов

Для открытия дашборда необходимо навести мышку на иконку и выбрать пункт . Далее дашборд откроется в новой вкладке на главной странице (см. Рисунок 9. Действия с дашбордами ).

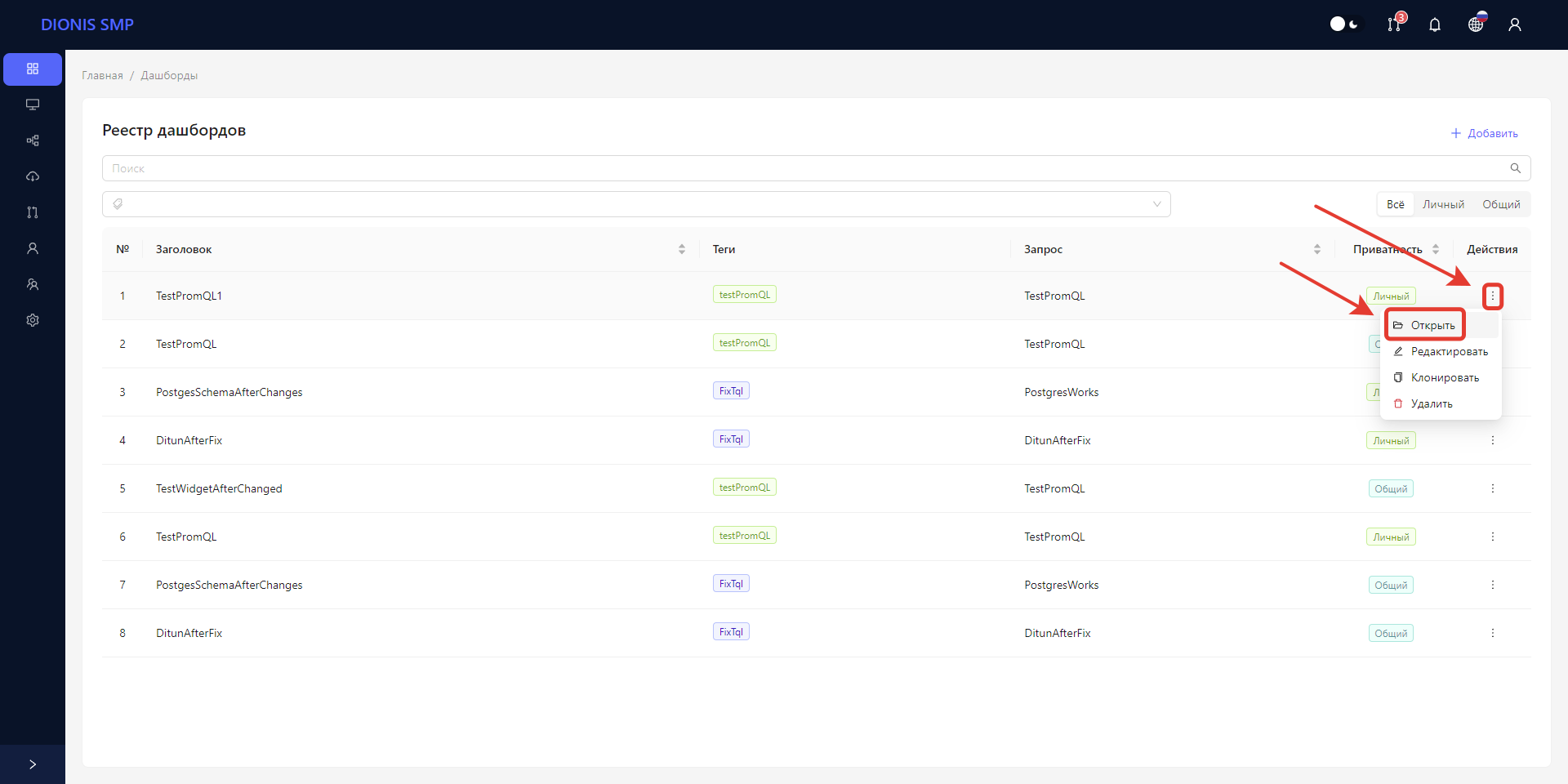


Рисунок 9. Действия с дашбордами

Для изменения дашборда необходимо навести мышку на и выбрать пункт .



Для удаления дашборда необходимо навести мышку на и выбрать пункт . Затем удаление нужно подтвердить .



Для клонирования дашборда необходимо навести мышку на и выбрать пункт . После чего создастся клон дашборда с таким же именем, виджетами, но с модификатором **«Личный»**.



#### Окно создания дашборда

Окно создания дашборда (см. Рисунок 10. Окно создания дашборда) состоит из полей :

1. название – то, как будет называться дашборд. Поле является обязательным для заполнения;
2. TQL запрос – на основе какого TQL запроса будут строиться виджеты;
3. теги – необязательное поле. Помогает группировать и искать нужные дашборды;
4. принадлежность – дашборд может быть «Общим» или «Частным» – если дашборд общий – его видят все пользователи, обладающие к нему доступом, если дашборд частный – то только текущий пользователь;
5. шаблон – необязательное поле. Дает возможность протестировать дашборд на урезанном количестве данных, что актуально для встраивания дашборда в дерево Виджет:
   1. выбираем, в какой узел TQL будет проставлен выбранный КЕ;
   2. выбираем КЕ.

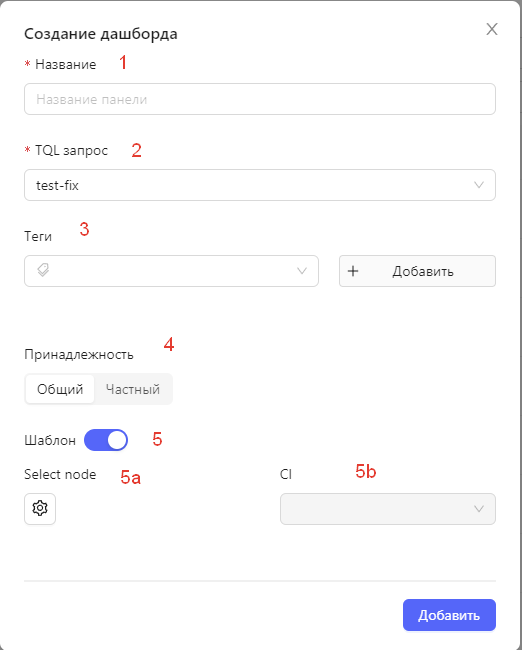


Рисунок 10. Окно создания дашборда

#### Панель инструментов дашборда

Панель инструментов расположена справа от вкладок дашбордов. Панель инструментов позволяет (см. Рисунок 11. Панель инструментов дашборда ):

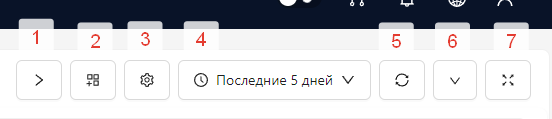


Рисунок 11. Панель инструментов дашборда

1. Скрыть/Раскрыть панель управления дашбордами;

Раскрытый вид (см. Рисунок 12. Раскрытый вид дашборда ):



Рисунок 12. Раскрытый вид дашборда

Скрытый вид (см. Рисунок 13. Скрытый вид дашборда ):



Рисунок 13. Скрытый вид дашборда

1. Добавить виджет. Виджет можно свободно масштабировать и передвигать по дашборду (см. Рисунок 14. Добавление виджета );

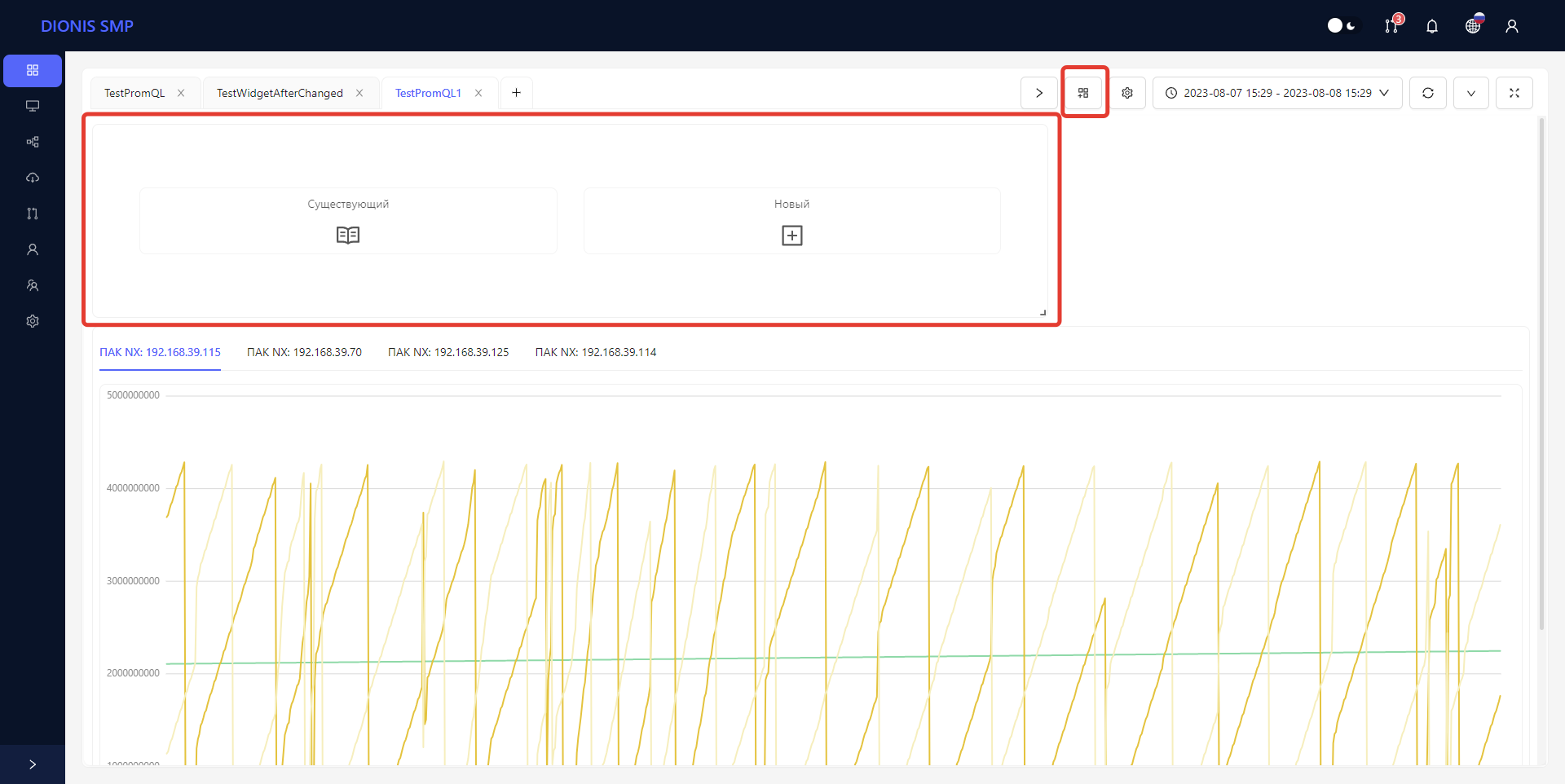


Рисунок 14. Добавление виджета

1. Настроить параметры дашборда;
2. Настроить интервал времени, за который виджеты будут отображать данные (см. Рисунок 15. Интервал отображения данных );

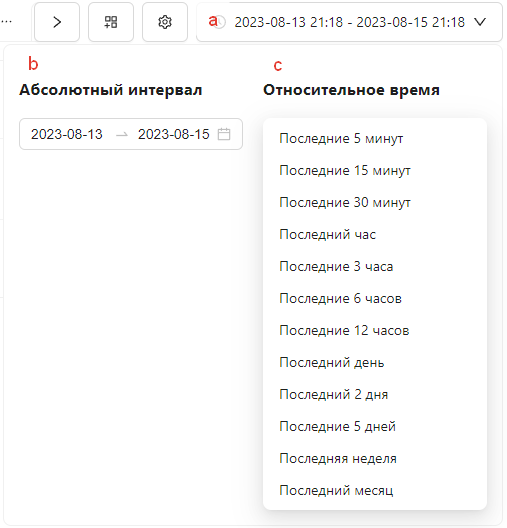


Рисунок 15. Интервал отображения данных

* 1. Отображение текущего интервала – показывает текущий интервал, также при клике открывает всплывающее окно с настройкой интервала;
  2. Абсолютный интервал – выставляем диапазон дат, за которые необходимо получить информацию;
  3. Относительное время – диапазон разница с текущим временем – за который получаем данные для дашборда;

1. Принудительно обновить данные;
2. Настроить интервал обновления – промежуток времени, через который дашборд запросит обновление данных (см. Рисунок 16. Интервал обновления данных );



Рисунок 16. Интервал обновления данных

1. Открыть дашборд на весь экран.

#### Управление виджетами

Каждый виджет является независимой единицей, встроенной в ячейку дашборда.

##### Создание виджета

После создания ячейки в дашборде появится возможность создать новый виджет или выбрать существующий (см. Рисунок 17. Создание виджета ):

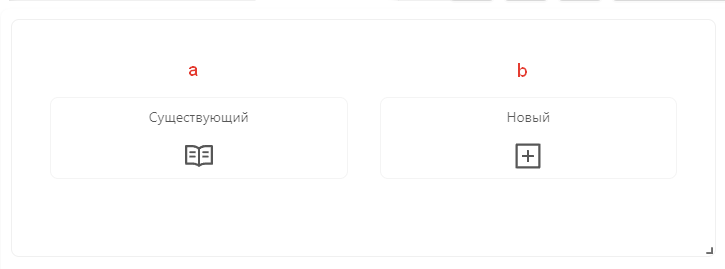


Рисунок 17. Создание виджета

1. будет склонирован существующий виджет;
2. будет создан новый виджет.

##### Изменение виджета

Для того чтобы изменить виджет, необходимо (см. Рисунок 18. Действия над виджетами ):

* навести мышку на верхнюю часть ячейки. Появится окно редактирования виджета;

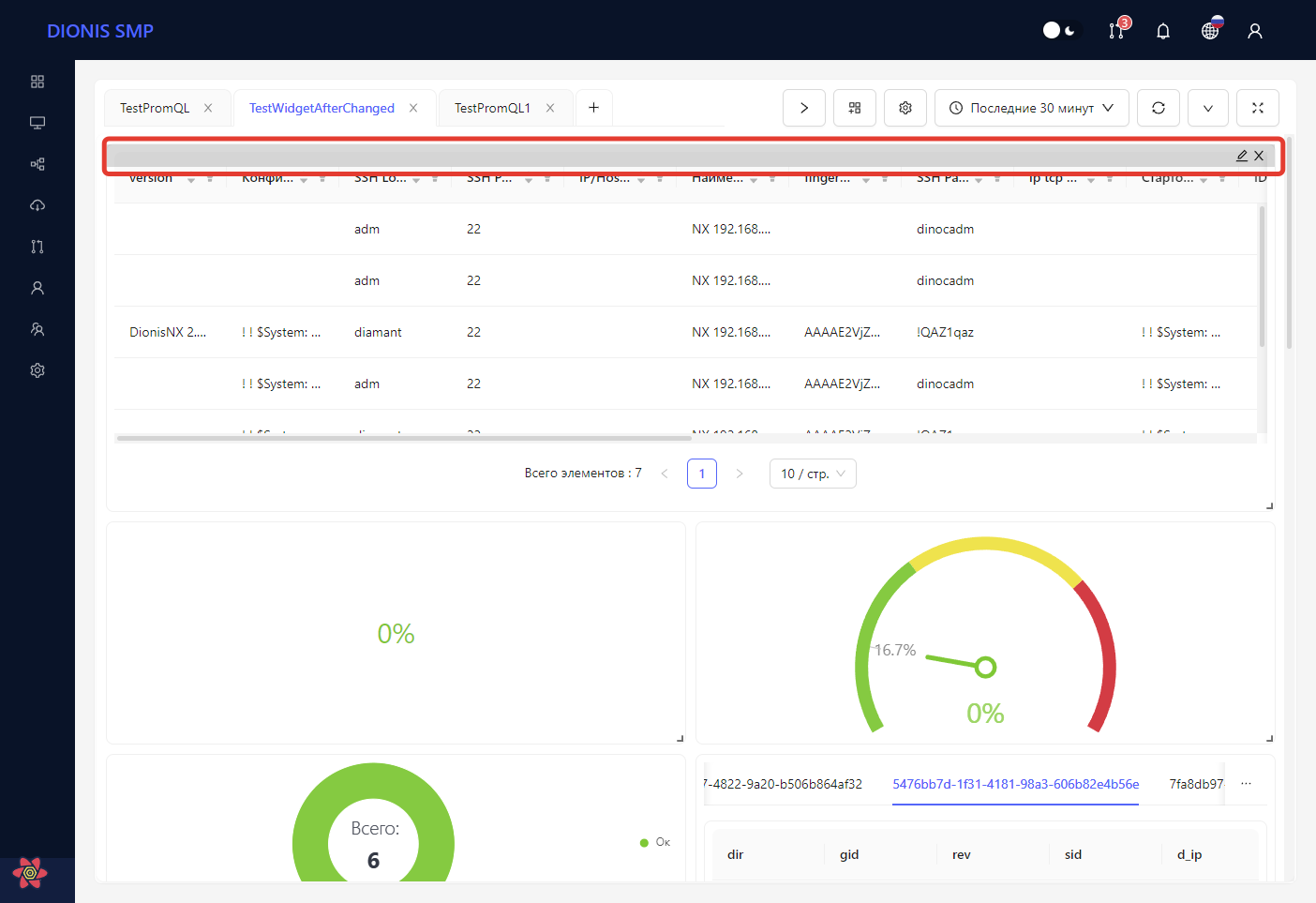


Рисунок 18. Действия над виджетами

* нажать на кнопку редактировать – откроется окно редактирования виджета.

Для того чтобы удалить виджет, необходимо:

* Нажать на кнопку удалить – будет удалена ячейка и сам виджет.

##### Окно создания виджета

Для создания и редактирования виджета предназначено окно (см. Рисунок 19. Окно создания виджета ).

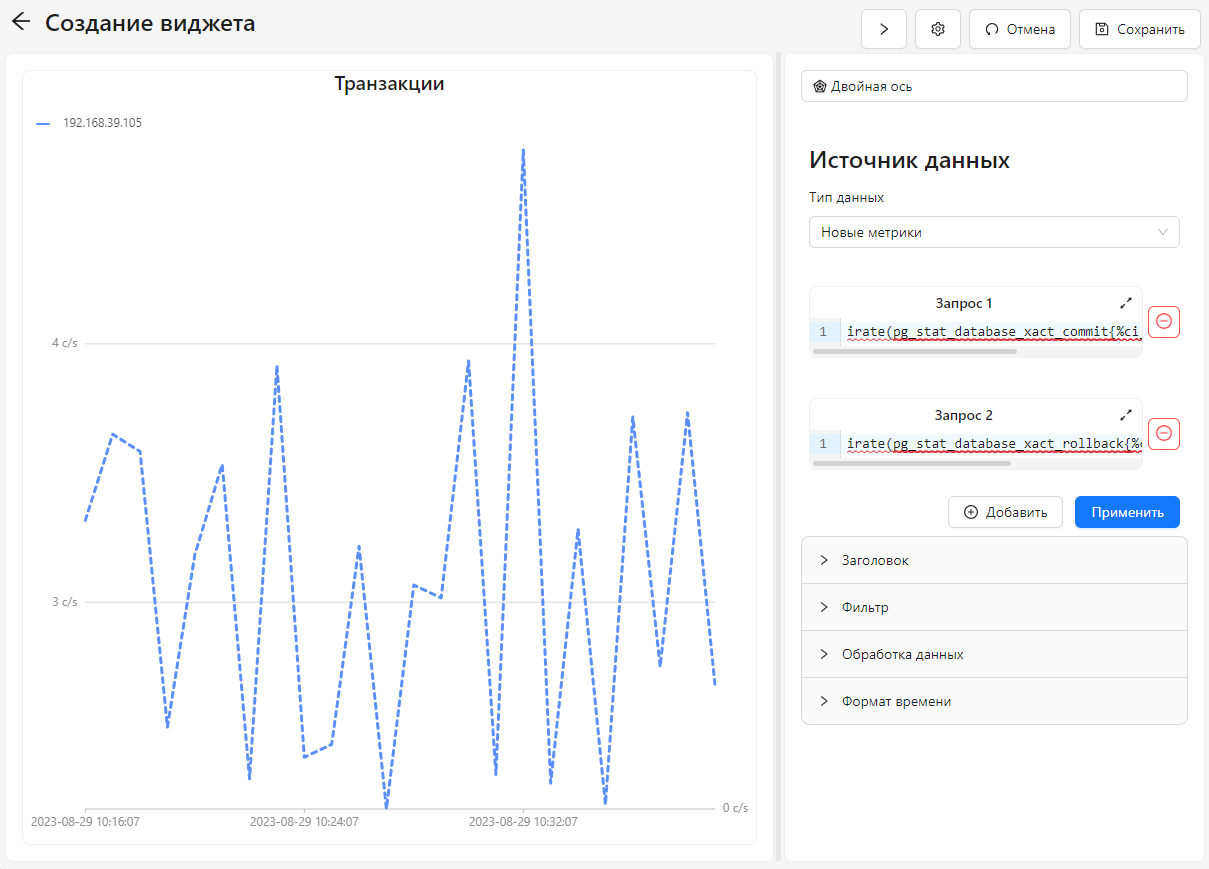


Рисунок 19. Окно создания виджета

Окно делится на три части :

* шапка;
* блок предпросмотр;
* настройка конфига виджета.

Шапка (см. Рисунок 20. Шапка виджета ) состоит из:

1. Кнопка назад – закрывает модально окно;
2. Скрыть/Раскрыть панель управления;
3. Настройки дашборда – можно поменять дашборд под виджет прямо из окна создания;
4. Отмена – откатывает изменения на начальные;
5. Сохранить – сохраняет изменения.



Рисунок 20. Шапка виджета

Окно предпросмотра позволяет просматривать как будет выглядеть виджет после его настройки на дашборде (см. Рисунок 21. Окно предпросмотра виджета ).

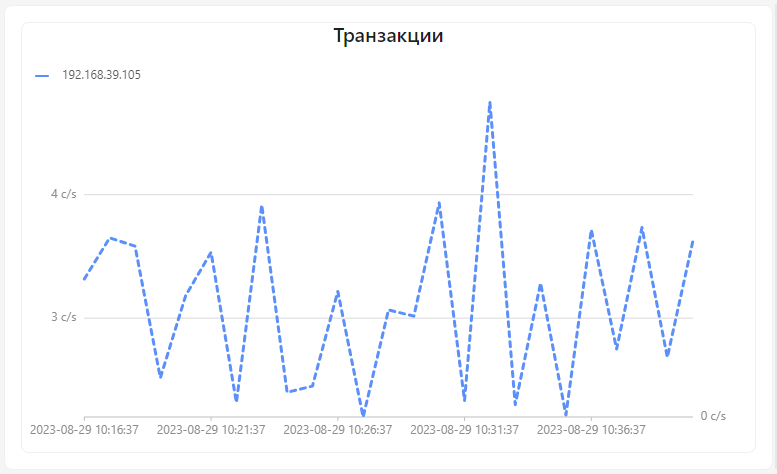


Рисунок 21. Окно предпросмотра виджета

Окно настройки конфигурации виджета имеет два состояния:

* Выбор виджета – выбор типа виджета для добавления на дашборд (см. Рисунок 22. Выбор виджета ):

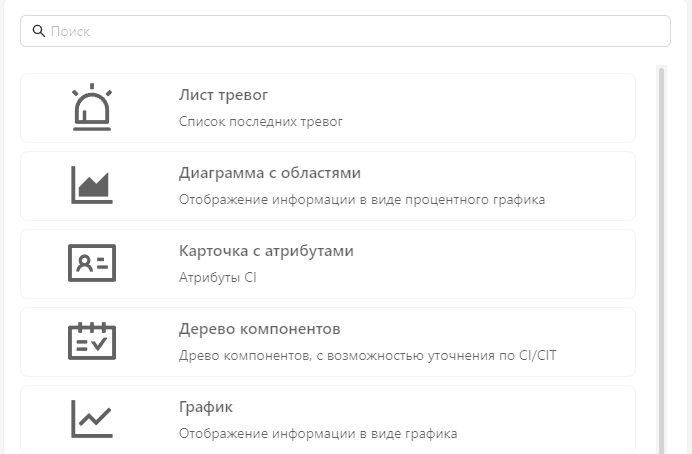


Рисунок 22. Выбор виджета

* Окно настройки виджета – настраиваем конфигурацию виджета (см. Рисунок 23. Настройки виджета ):

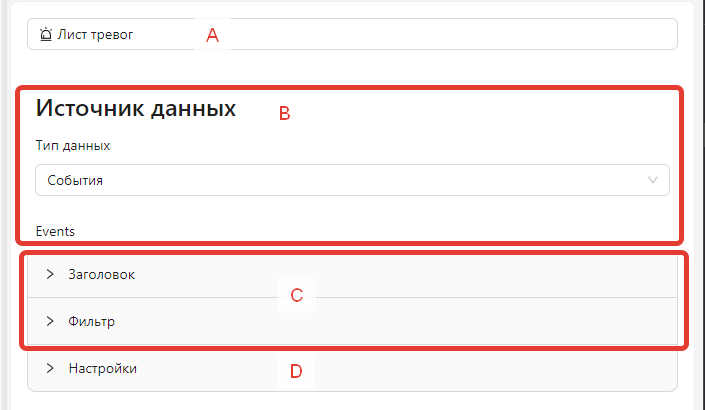


Рисунок 23. Настройки виджета

* 1. тип текущего виджета;
  2. источник данных – один тип виджета может уметь отрисовывать информацию на основе различных типов данных;
  3. стандартные настройки для всех виджетов;
     1. заголовок – отображается в шапке ячейки дашборда (см. Рисунок 24. Заголовок виджета ):

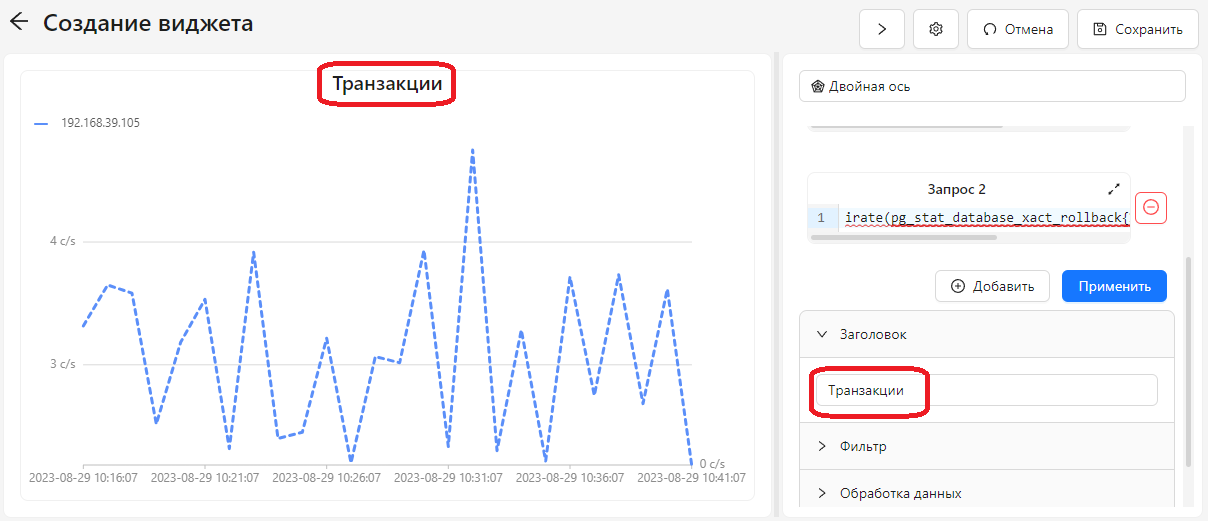


Рисунок 24. Заголовок виджета

* + 1. фильтр (см. Рисунок 25. Фильтр ТКЕ и КЕ ) – фильтрует данные из TQL для отображения в виджете. **Важно!** Если необходимо создать универсальный виджет, который можно будет клонировать – желательно избегать заполнения данной конфигурации, т.к. она высчитывается для каждого TQL отдельно;

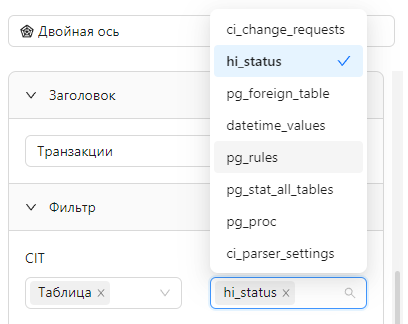


Рисунок 25. Фильтр ТКЕ и КЕ

* 1. Дополнительные настройки – расширение настроек, индивидуально для каждого типа виджетов.

###### Источники данных:

* аттрибут;
* событие;
* граф;
* метрика – требуется заполнить запрос для корректной работы связанных виджетов (см. Рисунок 26. Источник данных Метрики );

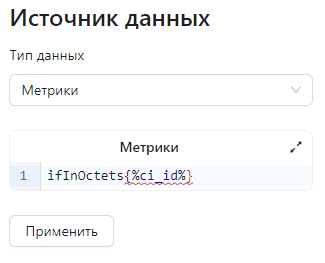


Рисунок 26. Источник данных Метрики

* обновленный тип метрики – при выборе нового типа метрик необходимо заполнить произвольное количество запросов, для корректной работы виджетов расчитанных на данный тип данных (см. Рисунок 27. Источник данных новые метрики ):

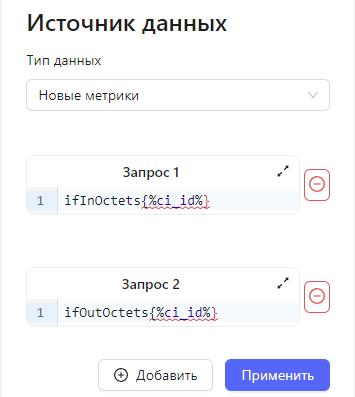


Рисунок 27. Источник данных новые метрики

* индикатор здоровья – при выборе индикатора здоровья необходимо уточнить, будем ли мы использовать статус КЕ или какого-то конкретного индикатора здоровья (см. Рисунок 28. Источник данных Индикаторы здоровья ):

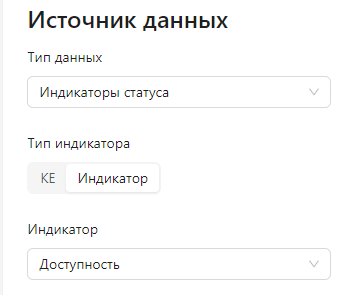


Рисунок 28. Источник данных Индикаторы здоровья

##### Виджет «Лист тревог»

Виджет «Лист тревог» используется для отображения состояния индикаторов здоровья КЕ, полученных в результате TQL, привязанного к текущему дашборду (см. Рисунок 29. Виджет «Лист тревог» ).

Особенности настройки виджета:

1. показывать историю – если переключатель выключен – то будут показаны текущие статусы по всем КЕ с настроенным мониторингом из TQL. Если переключатель включен – то будут отображаться последние события, связанные с изменениями статусов;
2. настройка сортировки по умолчанию – выбор сортировки, которая будет применена по умолчанию. После отрисовки виджета сортировки можно менять, используя возможности таблицы.

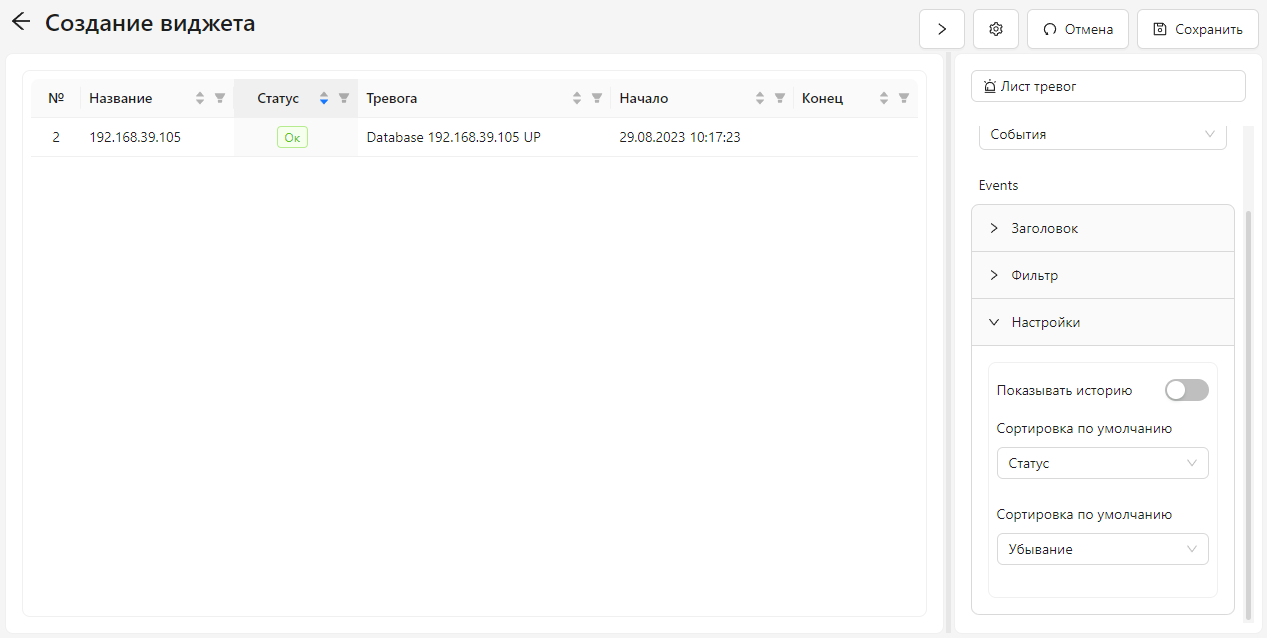


Рисунок 29. Виджет «Лист тревог»

##### Виджеты «Диаграмма с областями» и «График»

Для графического отображения данных предусмотрены два виджета: «Диаграмма с областями» и «График».

Особенности настройки виджета:

1. перезапись осей (см. Рисунок 30. Перезапись осей ) – переводит полученные данные в читаемый формат. Существует 2 перезаписываемые оси – ось времени (yAxisConfig) и ось значение (xAxisConfig);



Рисунок 30. Перезапись осей

1. Настройка событий (см. Рисунок 31. Настройка событий ) – настройка каких-либо действий при взаимодействии с элементами графика. Возможно добавить конечное множество событий;

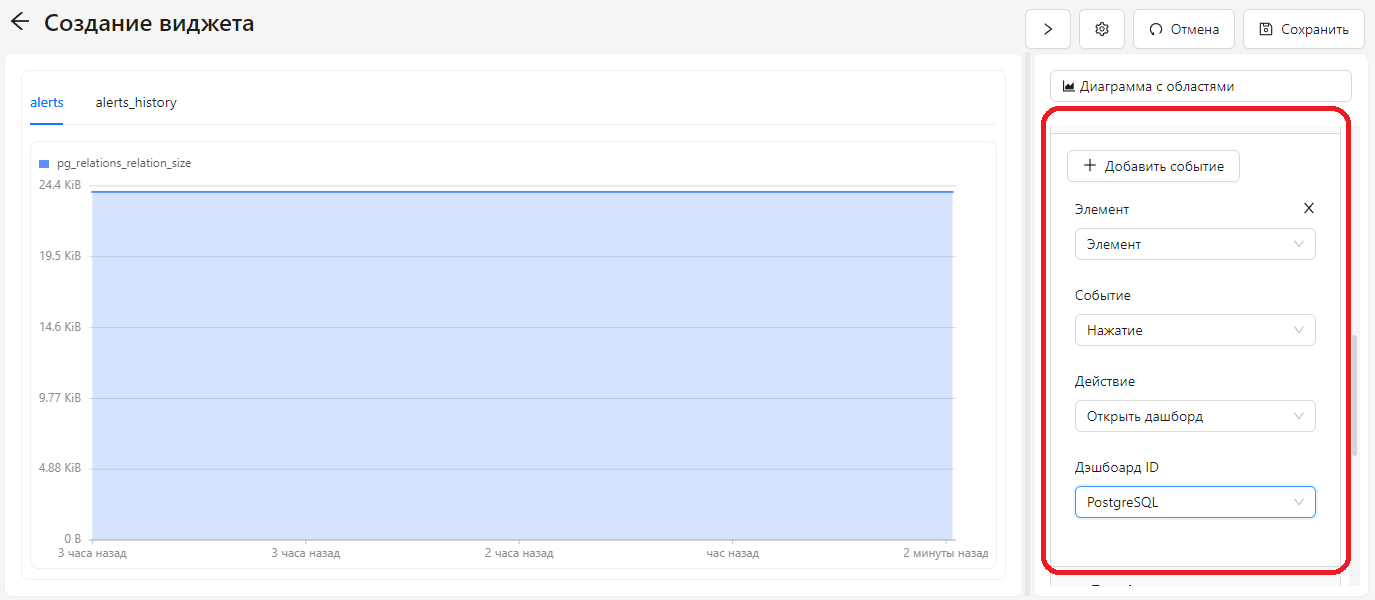


Рисунок 31. Настройка событий

1. преобразование данных (см. Рисунок 32. Преобразование данных и Рисунок 33. Преобразование данных, продолжение) – часто данных запроса не хватает, чтобы представить результат в виде читаемого графика;
   1. источник названия – какой параметр метрики использовать в качестве названия;
   2. группировка – поскольку запрос часто строится для группы КЕ, то данные желательно сгруппировать, например, по названию КЕ;
   3. ключ группировки – по какому ключу группировать данные;
   4. способ отображения – каким способом данные будут отображаться. При выборе опции «Табы» каждый график будет отображаться в отдельной вкладке. При выборе «Сетка» – для каждого сгруппированного набора данных будет создана отдельная ячейка;



Рисунок 32. Преобразование данных

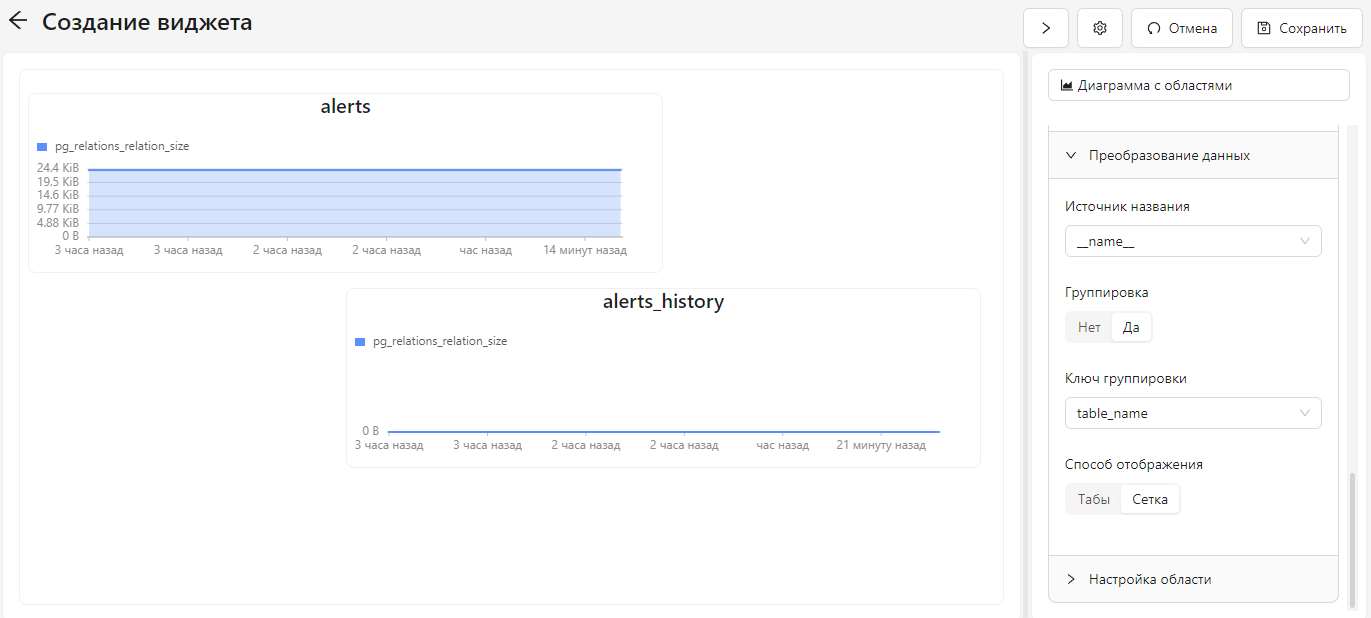


Рисунок 33. Преобразование данных, продолжение

* 1. настройка области (есть только у виджета «Диаграмма с областями») – если isPercent активна – то на каждой точке графика высчитывается процентное значение данных, а не абсолютное.

##### Виджет «Карточка с атрибутами»

Виджет Карточка с атрибутами предназначен для вывода данных атрибутов КЕ, полученных в результате выполнения TQL (см. Рисунок 34. Виджет Карточка с атрибутами ).

Из уникальных настроек применяется только Способ отображения – каким способом данные будут отображаться. При выборе опции «Табы» – каждый график будет отображаться в отдельной вкладке. При выборе «Сетка» – для каждого сгруппированного набора данных будет создана отдельная ячейка.

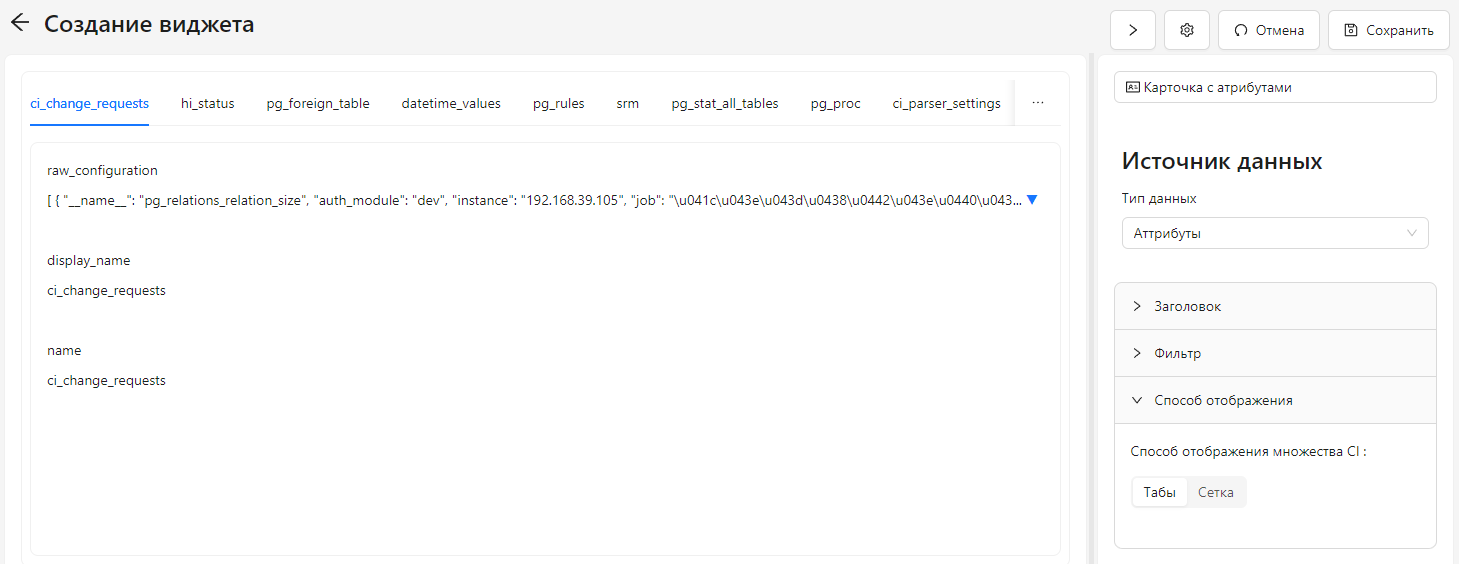


Рисунок 34. Виджет Карточка с атрибутами

##### Виджет «Дерево компонентов»

Виджет «Дерево компонентов» (см. Рисунок 35. Виджет Дерево компонентов ) предназначен для отображения информации из TQL в виде дерева. Для этого необходимо настроить «Конструктор дерева».

Конструктор дерева. При изменении дерева в «Конструктор дерева» – меняется дерево, отображаемое в виджете. Если у типа КЕ не стоит галочки – то при его отображении не будет создана групповая папка (на рисунке папками являются «База данных» и «Схема», в то время как таблицы просто встроены в схему public, см. Рисунок 35. Виджет Дерево компонентов ):

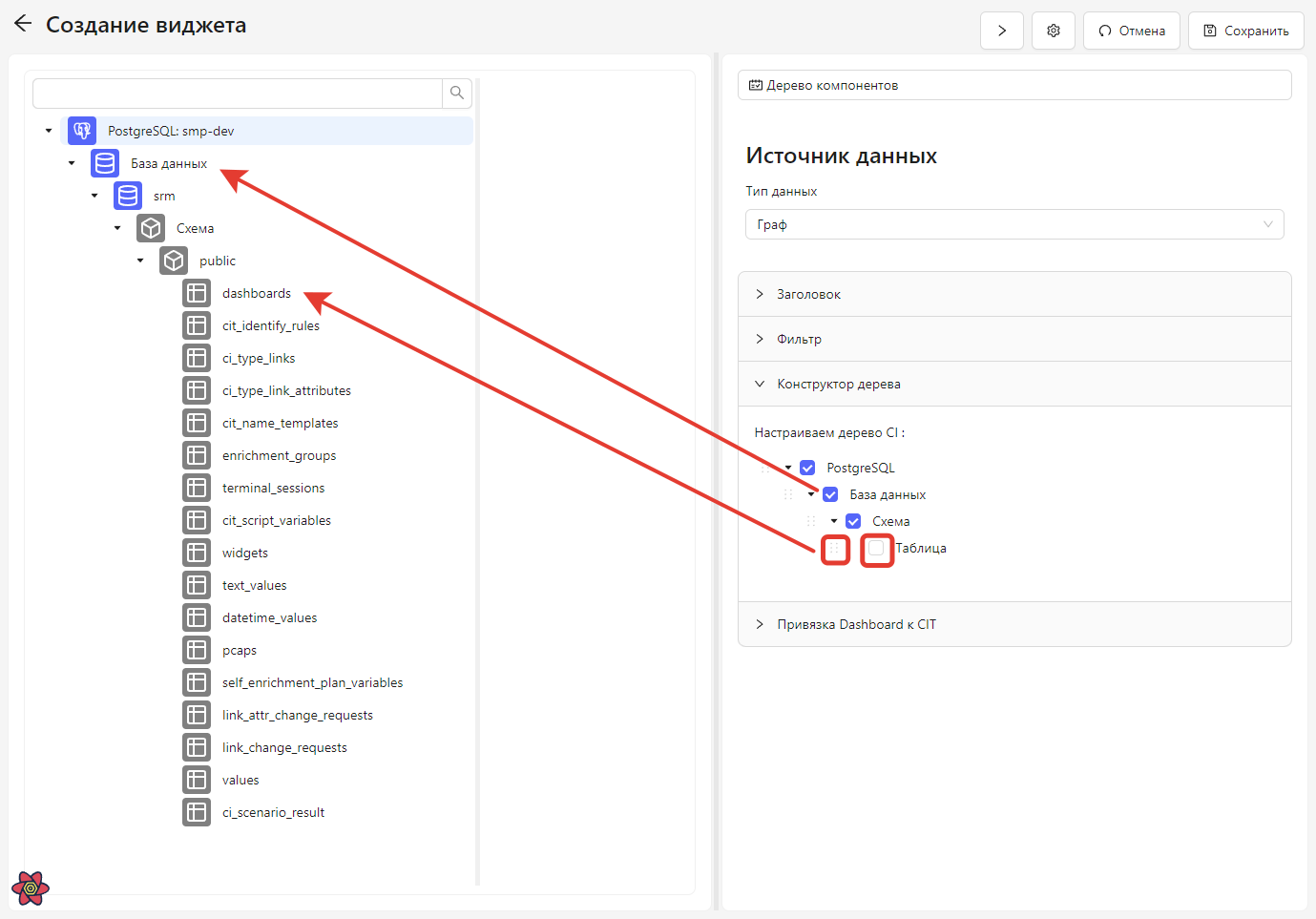


Рисунок 35. Виджет Дерево компонентов

Важно! Если неправильно составить дерево, то элементы, которые абсолютно никак не связаны, попадут в «bad schema» (см. Рисунок 36. Bad schema ) и часть данных будет скрыта при отображении после сохранения, т.к. bad links показываются только в режиме просмотра:

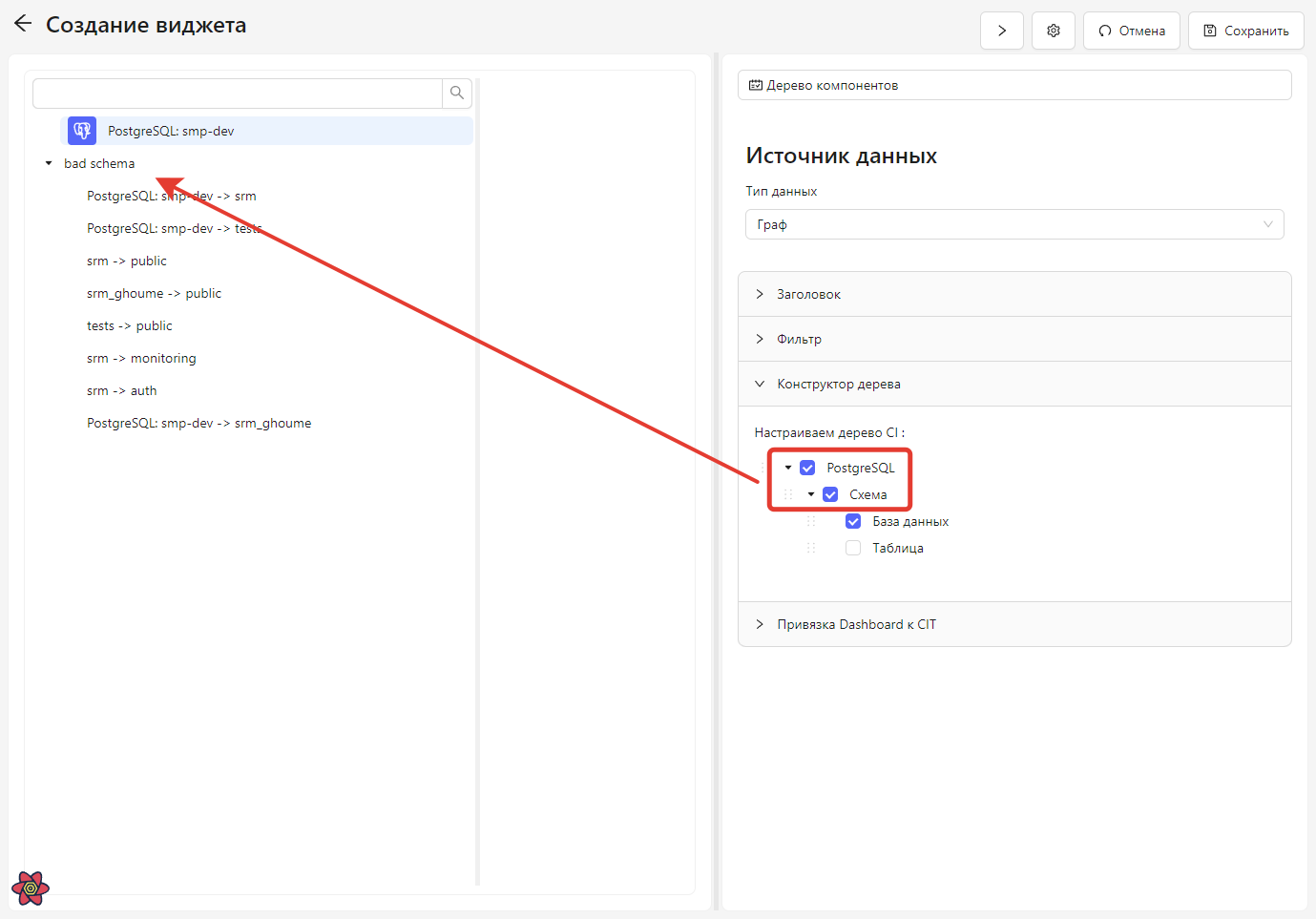


Рисунок 36. Bad schema

«Привязка Dashboard к CIT» - «Дерево компонентов» позволяет встраивать Дашборды внутрь себя. Возможно настроить дашборд для каждого элемента дерева, причем также и отдельно настраивать для групп КЕ и самого КЕ.

Если в TQL дашборда есть две ноды, удовлетворяющие КЕ, то при клике на иконку можно настроить, в какую именно ноду будет проставляться условие поиска по КЕ (см. Рисунок 37. Привязка вложенного виджета ):

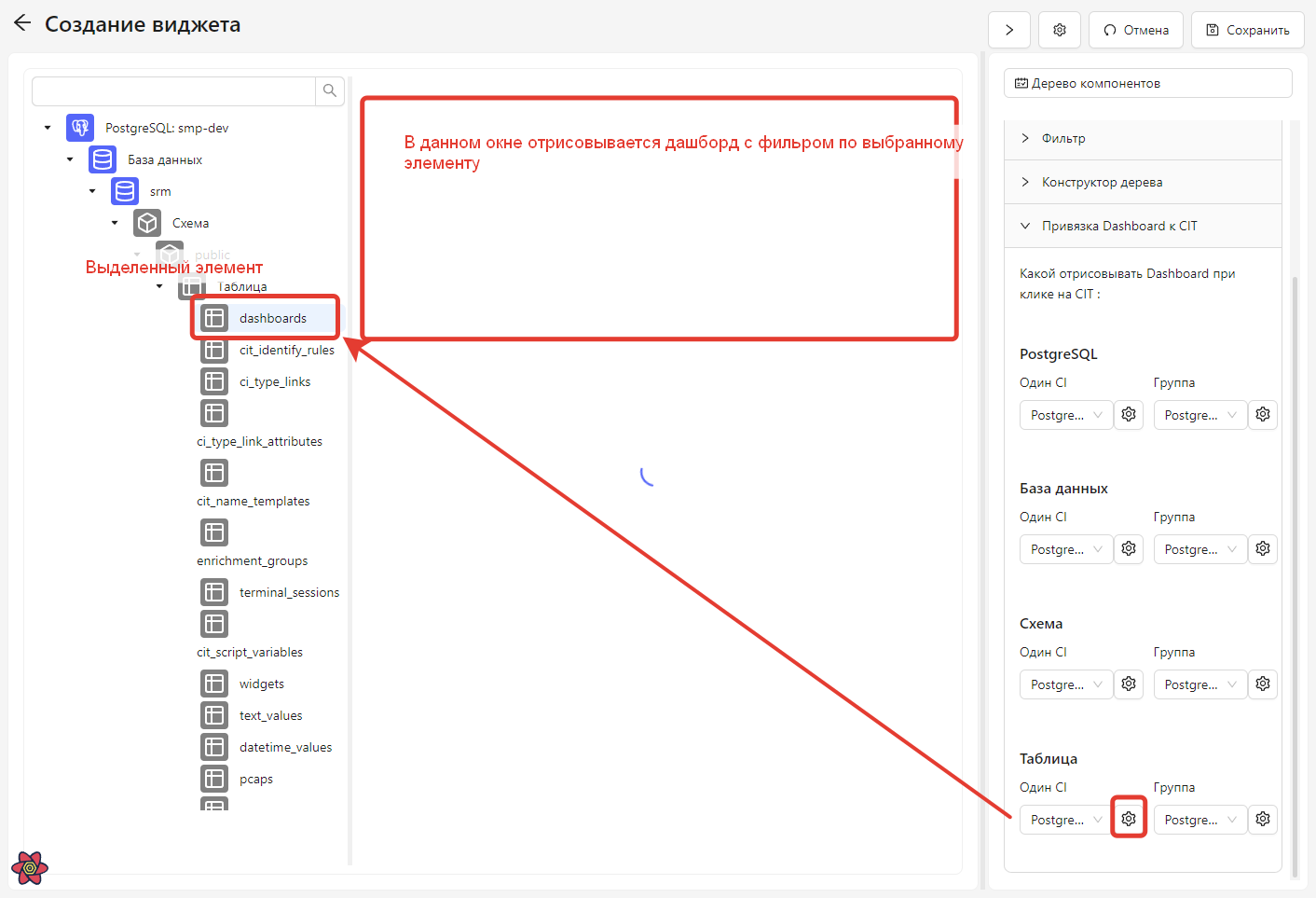


Рисунок 37. Привязка вложенного виджета

##### Виджет «Таблица»

Виджет Таблица выполняет функцию отображения табличных данных. Если в TQL присутствуют несколько типов КЕ, то для каждого из типов рисуется отдельная вкладка с таблицей. В связи с этой особенностью, настройка таблицы начинается с выбора ТКЕ, для которого осуществляется настройка. В приводимом примере показана настройка таблицыдля ТКЕ «Таблица» (см. Рисунок 38. Виджет Таблица):

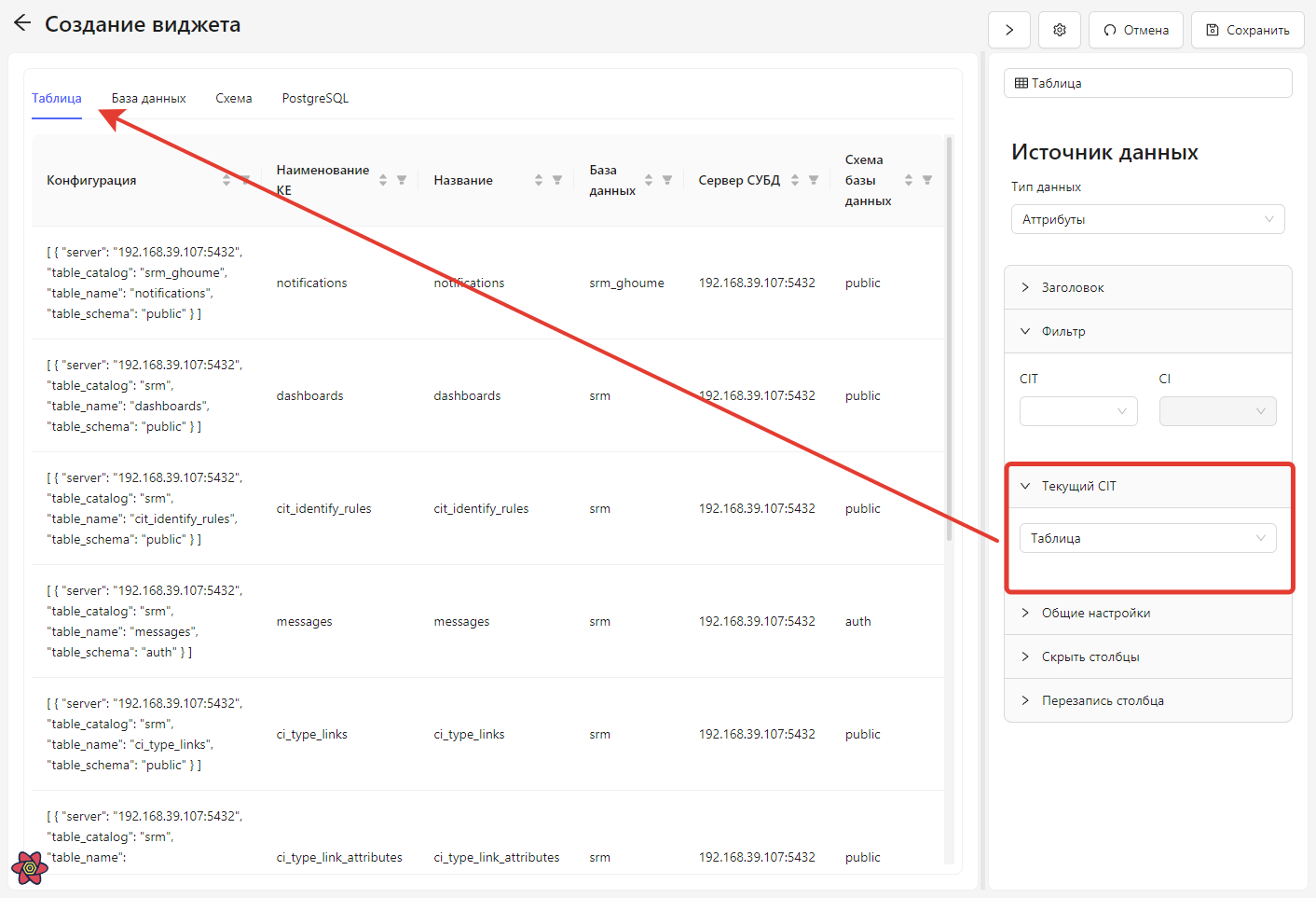


Рисунок 38. Виджет Таблица

После выбора ТКЕ для настройки возможно приступить к настройке остальных опций (см. Рисунок 39. Общие настройки ):

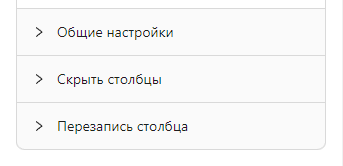


Рисунок 39. Общие настройки

1. настройки таблицы (см. Рисунок 40. Настройки виджета таблица ):
   1. обводка – добавляет/убирает обводку у ячеек;
   2. размер – размер отступов у ячеек;
   3. макет таблицы – есть два варианта: «Фиксированный», «Автоматический». Фиксированный сохраняет заданную ширину ячеек. Автоматический растягивает ячейки в зависимости от их наполнения;
2. общие настройки для столбцов:
   1. выравнивание – выравнивание текста внутри ячейки;
   2. многоточие – если включено – то урезает контент ячейки по ширине;
   3. ширина – имеется возможность задать ширину для всех колонок;
3. настройки расширений:
   1. показывать номера столбцов – дополнительно показывает № столбца;
   2. пагинация - включает/выключает пагинацию;

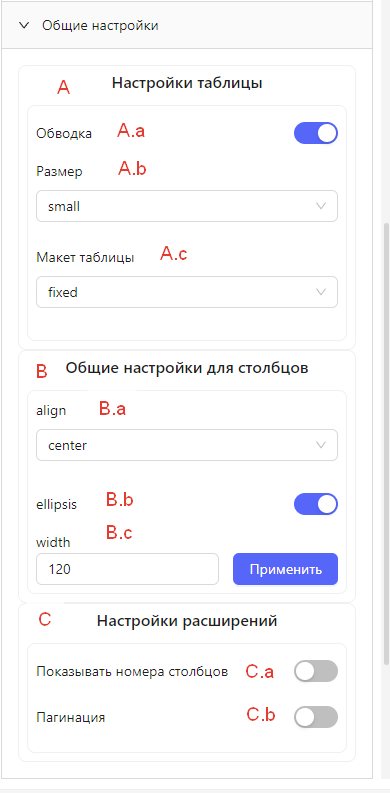


Рисунок 40. Настройки виджета таблица

Скрыть столбцы (см. Рисунок 41. Скрыть столбцы ) – позволяет менять порядок колонок и скрывать лишние. Для смены порядка необходимо перенести интересующую колонку за иконку . Для скрытия столбца необходимо кликнуть на чекбокс:

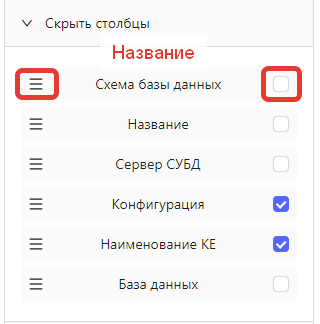


Рисунок 41. Скрыть столбцы

Перезапись столбцов (см. Рисунок 42. Перезапись столбцов ) – позволяет точечно настроить столбец из таблицы. Для того чтобы перейти к настройке, необходимо нажать кнопку «добавить» и выбрать Идентификатор соответствующего столбца. Далее необходимо заполнить требуемые параметры:

1. кнопка добавить – добавляет перезапись;
2. идентификатор – выбираем, какой столбец будем настраивать;
3. заголовок – меняет заголовок столбца;
4. единицы измерения – форматирует вывод таблицы на основе установленных единиц измерения;
5. выравнивание;
6. многоточие – обрезать ли контент ячейки;
7. ширина – ширина в пикселях для ячейки;

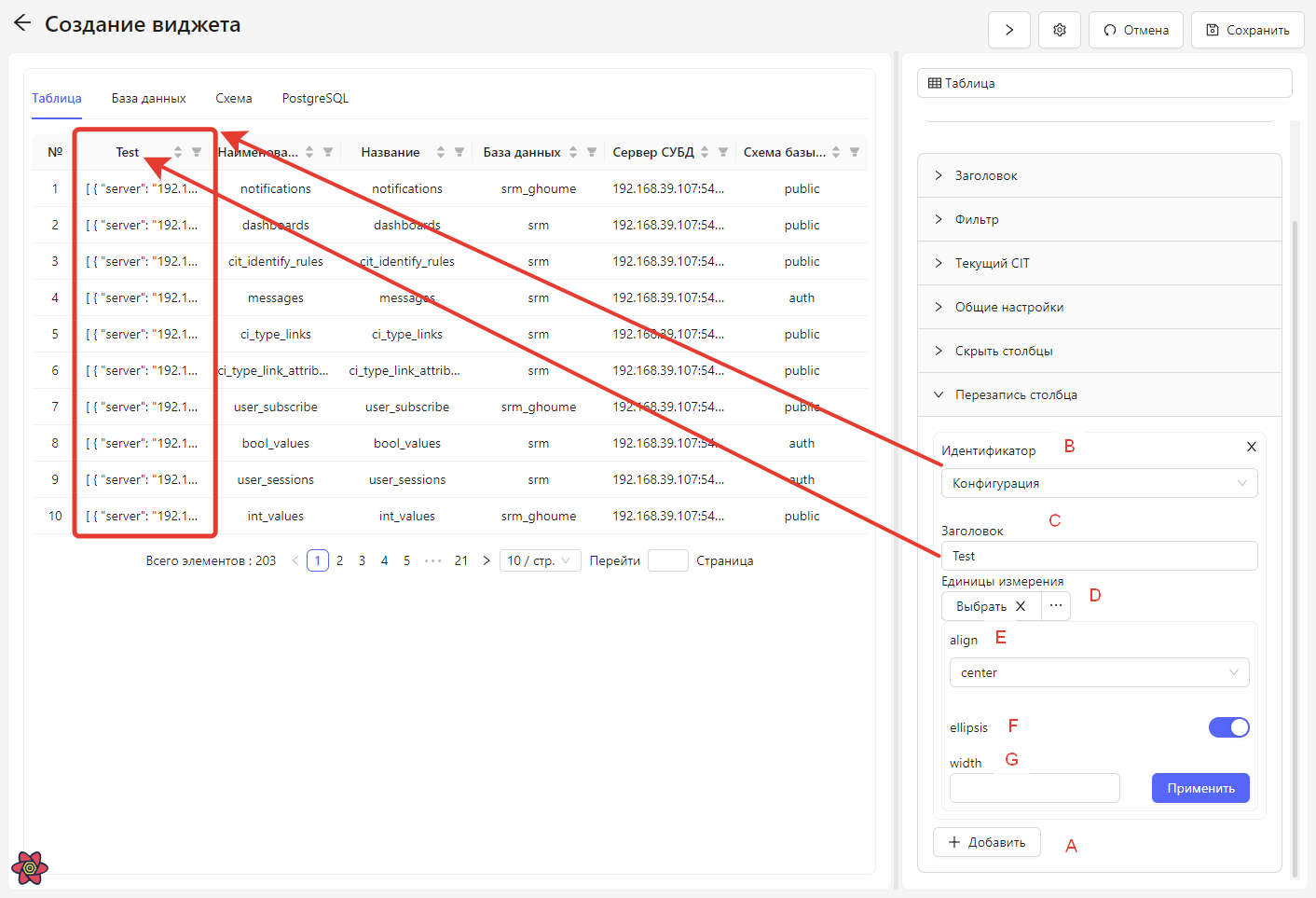


Рисунок 42. Перезапись столбцов

##### Виджет Спидометр

В отличие от виджетов отображения статуса виджеты точного отображения состояния (Измеритель, Статус значения) помимо качественной оценки приходящего значения (зеленый, желтый, красный) также отображают и само значение, процент (см. Рисунок 43. Виджет Спидометр ).

Разберем особенности настройки данных виджетов на примере «Измерителя».

Основные настройки: ь задаются единицы измерения, а также минимальное/максимальное значение.

Пороги – стандартный порог по умолчанию маркируется зеленым цветом и начинается с минимального значения. Далее каждый добавленный порог добавляет дополнительное деление на отрезок значений :

1. цвет порога;
2. до какого значения срабатывает – заполняем значение, с которого срабатывает порог; значение берется либо от максимального , либо от следующего по очереди порога;
3. удалить порог;
4. добавить порог;
5. значение в % либо абсолютное – расчитывается текущее значение в процентах от диапазона «от – до», либо порогом будет являться абсолютное значение. Пример: если от : 0, до : 200, то 50% = 100, а 50 в формате абсолютного значения = 50.

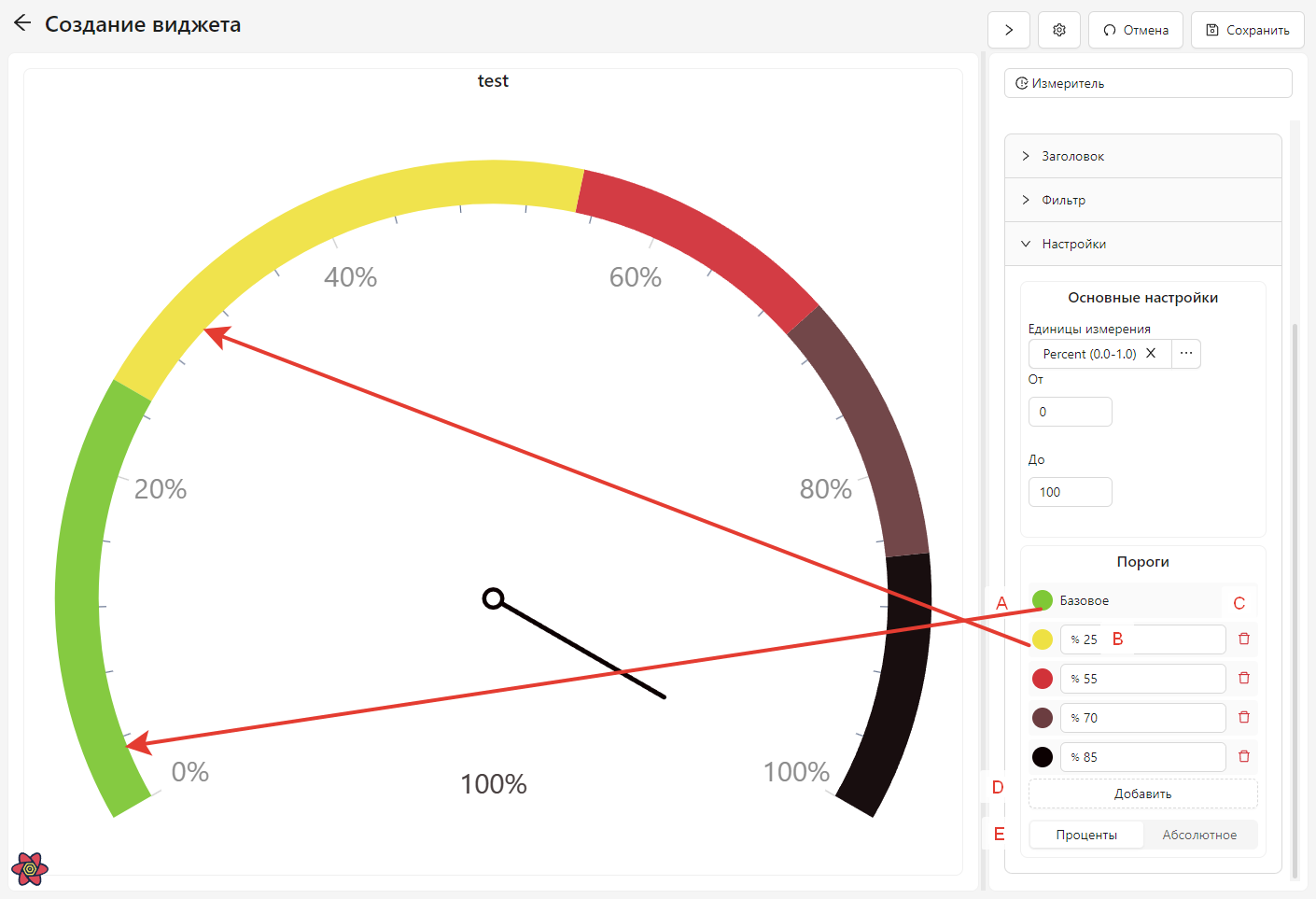


Рисунок 43. Виджет Спидометр

##### Виджет «Новая метрика»

Виджет «Новая метрика» (см. Рисунок 44. Виджет Новая метрика ) позволяет более гибко настраивать получение данных за счет привязки группы запросов к конкретным КЕ. Перед настройкой виджета необходимо убедиться в заполнении «Источника данных»:

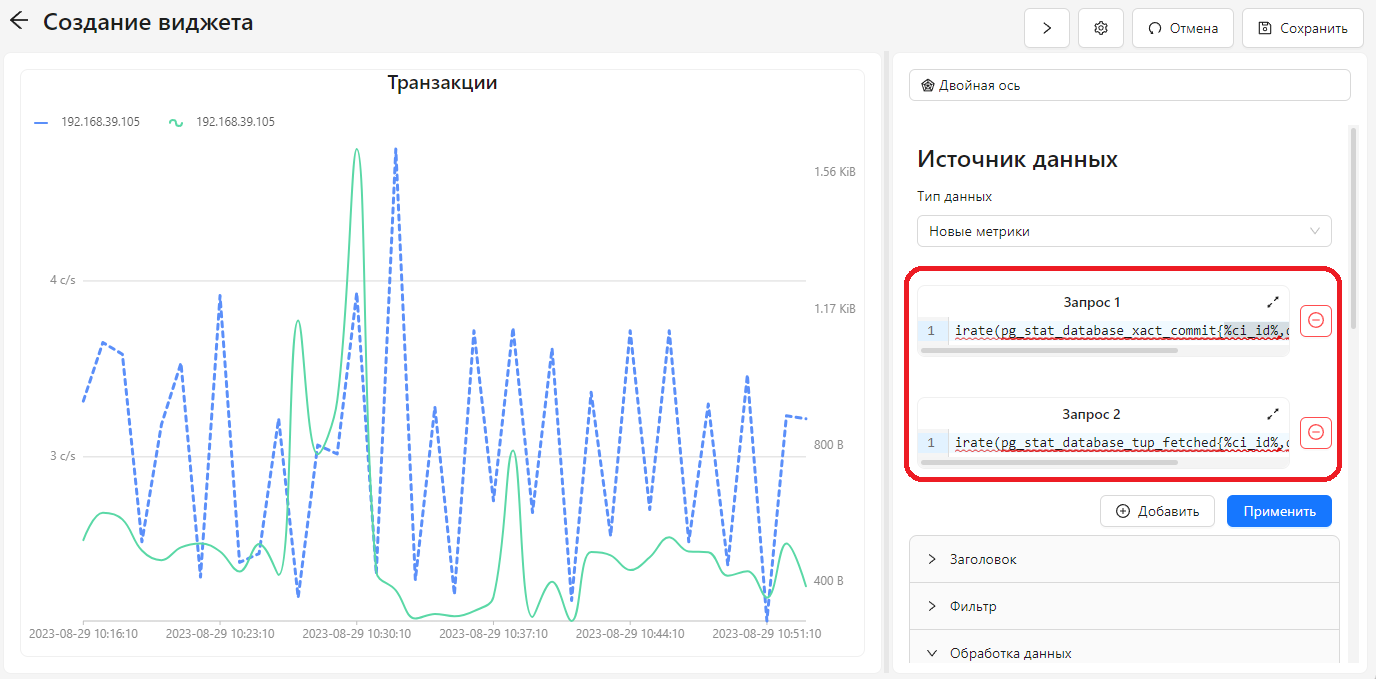


Рисунок 44. Виджет Новая метрика

Далее, если имеется несколько запросов, требуется начать заполнять «Источники» (см. Рисунок 45. Привязка запросов к осям значений ).

Источником являются данные для одной оси Y. На данный момент максимальное количество осей 2.

Источник состоит из группы парсеров и единиц измерения.

Парсером является связка из указания номера запроса и поля для уникального именования данных, т.к. в результате запроса может прийти несколько метрик:

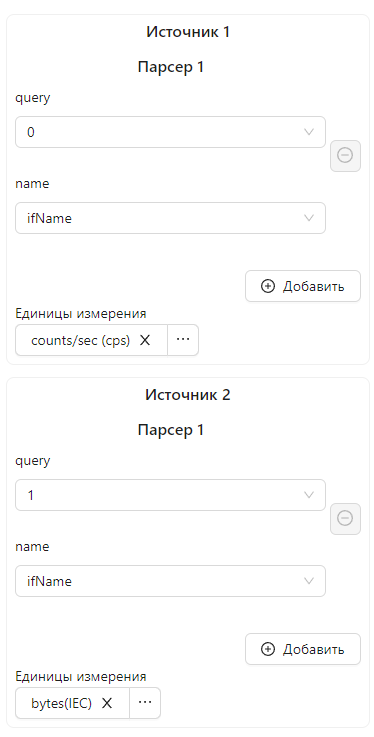


Рисунок 45. Привязка запросов к осям значений

Формат времени – формат, к которому необходимо привести время на оси Х.

#### Механика построения дашбордов

Для построения дашбордов необходимо:

* настроить в СРМ TQL для получения интересующей информации или, если есть удовлетворяющий TQL, выбрать его при создании дашборда;
* создать дашборд, привязав к нему выбранный TQL;
* при необходимости настроить параметры дашборда;
* добавить в него ячейки и заполнить их виджетами.

### СРМ

#### Фильтрация СРМ

СРМ может содержать огромное количество КЕ, десятки и сотни тысяч. Одновременно их все просмотреть невозможно. КЕ могут делиться по функциональному, организационному и иному признаку на группы. Эти признаки интересны разным пользователям. Каждый пользователь желает видеть только интересующую его группу КЕ. ПАМИР помогает в решении таких задач.

ПАМИР имеет инструмент фильтрации СРМ – Topology Query Language (TQL), язык описания топологии графа. TQL оперирует такими понятиями, как ТКЕ и связи. Он позволяет создавать маску из ТКЕ и накладывать ее на СРМ. Результатом выполнения TQL является ограниченный список КЕ.

TQL используется:

* как самостоятельный инструмент – для фильтрации СРМ;
* в правилах доступа – для ограничения видимости СРМ;
* в дашбордах – для ограничения КЕ, доступных для отображения в виджетах;
* в мониторинге – для привязки шаблонов и тревог к избранным КЕ.

#### Построение TQL

Topology Query Language (TQL) - это язык запросов для описания топологии графа. Он позволяет описывать различные свойства графов, такие как количество вершин и ребер, наличие циклов, связность и т.д. TQL является мощным инструментом для анализа и визуализации СРМ.

Вершина – точка на графе, которая имеет определенные свойства:

* соответствие ТКЕ;
* смежность с другими вершинами.

Вершины могут быть связаны друг с другом ребрами, которые образуют граф.

Ребра соединяют две вершины и имеют определенные свойства:

* тип (композиция, ассоциация, агрегация);
* мощность;
* направление мощности.

###### Фильтрация по ТКЕ

Построение TQL выполняется в модальном окне (см. Рисунок 46. Окно построения TQL ).

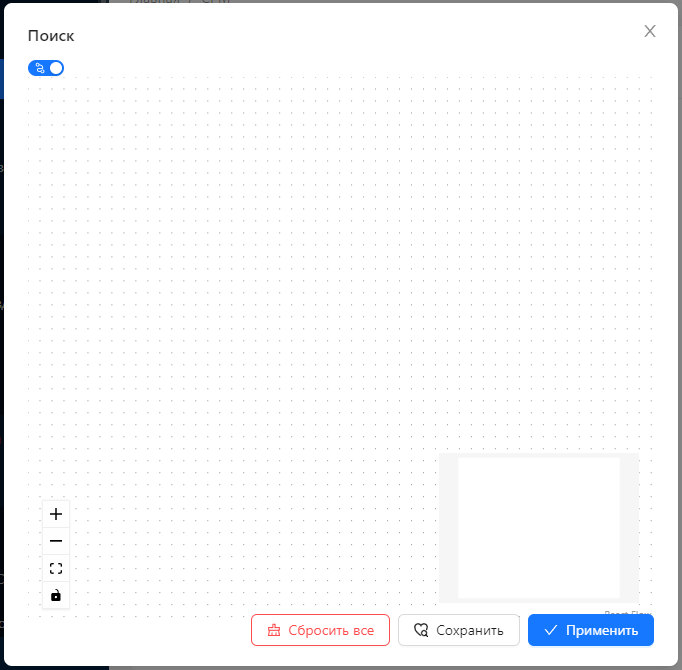


Рисунок 46. Окно построения TQL

Через контекстное меню (открывается по правой кнопке мыши) можно добавлять ТКЕ.

Для выбора определенного типа КЕ достаточно создать вершину данного типа и нажать кнопку «Применить» (см. Рисунок 47. Выбор всех КЕ типа Конфигурационная единица ).

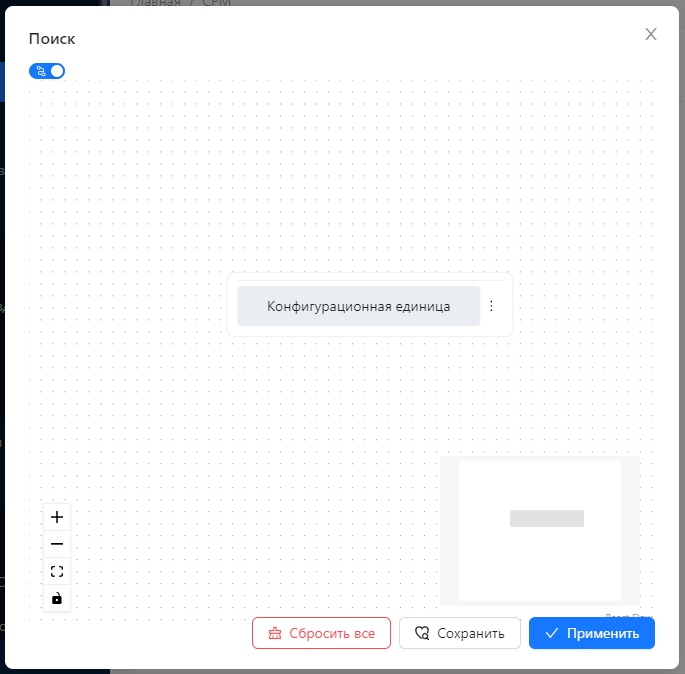


Рисунок 47. Выбор всех КЕ типа Конфигурационная единица

ТКЕ имеют иерархическую структуру. Для выбора КЕ, которые имеют общего родителя, нет необходимости добавлять все ТКЕ. Достаточно выбрать родительский тип. Будут выбраны все КЕ дочерних типов.

Пример.

Для выбора всех КЕ типа СУБД PostgreSQL, MySQL достаточно создать TQL с ТКЕ СУБД (см. Рисунок 48. Выбор всех КЕ типа СУБД ):

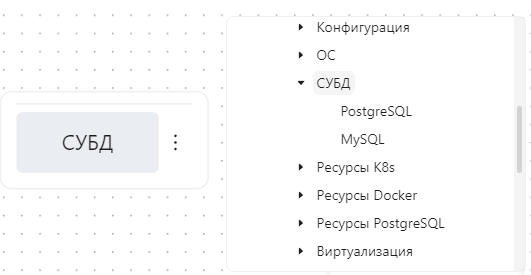


Рисунок 48. Выбор всех КЕ типа СУБД

###### Фильтрация по значению атрибутов

Фильтрация по значению аттрибутов выполняется в меню каждого ТКЕ. Доступ в меню ТКЕ открывается по наведению/клику по иконке [Три вертикальные точки] > Редактировать (см. Рисунок 49. Открыть окно фильтрации по значению атрибута ).

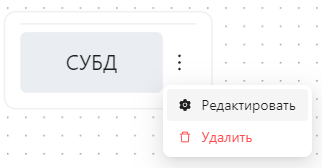


Рисунок 49. Открыть окно фильтрации по значению атрибута

Фильтр представляет собой логическое выражение, оперирующее атрибутами КЕ. Логическое выражение применяется для каждого КЕ соответствующего типа. Если логическое выражение разрешается в истину, то КЕ проходит фильтрацию и будет присутствовать в итоговом результате.

Выражение может оперировать:

* атрибутами КЕ;
* статусом КЕ.

Выражения могут объединяться в группы через логические операторы ( И / ИЛИ ). Группы могут быть вложенными.

Выражение состоит из названия атрибута, оператора сравнения и значения:

* название – название атрибута или другого свойства ТКЕ доступного для сравнения;
* оператор – оператор сравнения значения. Зависит от типа выбранного атрибута;
* значение – значение, которое TQL будет искать в СРМ;
* оператор выбора значения.

Оператор выбора значения позволяет указать способ заполнения значения выражения (см. Рисунок 50. Оператор выбора значения ):

* Ввод значения – требует указания литерала в самом выражении.
* Выбор из переменных – сопоставляет значение атрибута КЕ со значением атрибута пользователя.
* Ввод значения при выполнении – требует ввода значения в момент выполнения ФИР.



Рисунок 50. Оператор выбора значения

###### Связанные ТКЕ

TQL может состоять из нескольких несвязанных вершин. В таком случае будут выбраны КЕ все указанных типов. Между ними будут построены все существующие связи.

Связи можно скрыть, выключив переключатель .



TQL может фильтровать СРМ по связям между КЕ. Для этого нужно кликнуть на внутреннюю область вершины и, не отпуская левую кнопку мыши, связать с другой вершиной. При движении курсора TQL подсветит вершины, до которых может быть установлена связь. Эти настройки берутся из Дополнительных связей ТКЕ (см. Рисунок 51. Связывание ТКЕ ):

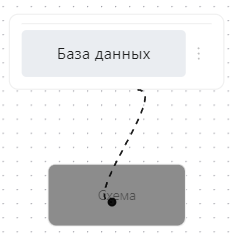


Рисунок 51. Связывание ТКЕ

В открывшемся диалоговом окне нужно выбрать тип связи, установить направление (см. Рисунок 52. Параметры связи ):

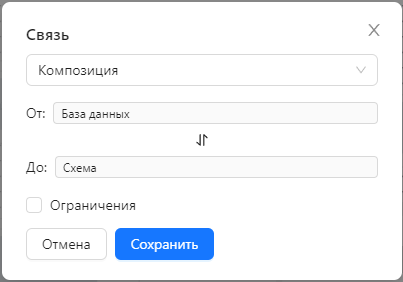


Рисунок 52. Параметры связи

Связанные вершины графа будут соединены линией. На линии выводится название связи и ее символ.

Редактировать связь можно, кликнув на ее названии левой кнопкой мыши.

Изображения типов связей:

* композиция – полилиния, начальный корректор изображен в виде черного закрашенного ромба. Ромб располагается со стороны композирующего КЕ;
* агрегация – полилиния, начальный корректор изображен в виде бесцветного ромба. Ромб располагается со стороны агрегирующего КЕ;
* ассоциация – полилиния, конечный корректор изображен в виде стрелки.

Итоговый TQL показан на примере КЕ PostgreSQL (см. Рисунок 53. TQL запрос ):

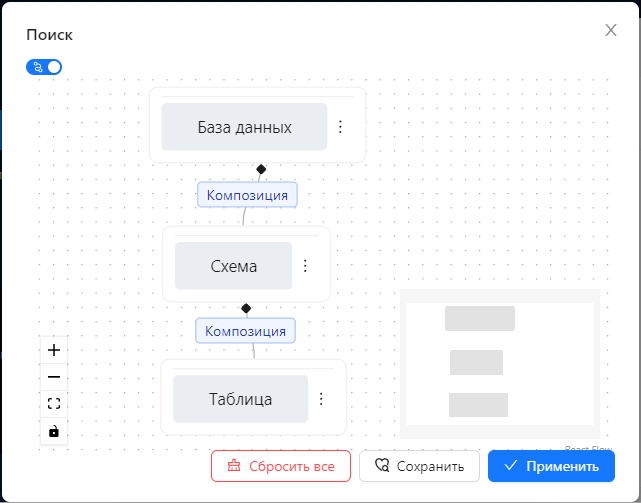


Рисунок 53. TQL запрос

###### Мощность связи

TQL позволяет накладывать количественные ограничения на связанные узлы. По умолчанию накладывается ограничение от 1 до \*, где \* означает бесконечность, со стороны конца связи (см. Рисунок 54. Мощность связи по умолчанию ):

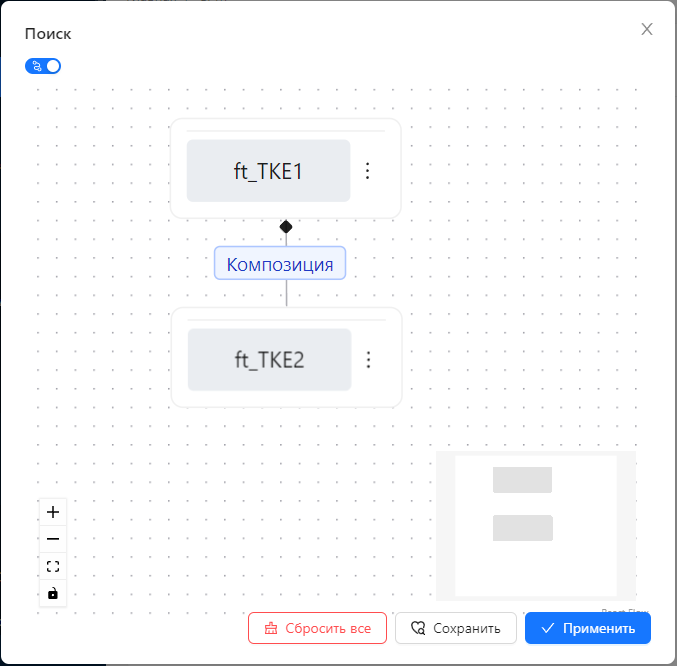


Рисунок 54. Мощность связи по умолчанию

Иными словами, это ограничение можно интерпретировать таким образом.

Отобразить все ТКЕ 1, которые связаны как минимум с одним ТКЕ 2.

Мощность связи можно корректировать, если включить пункт «Ограничения» в меню настройки связи.

Возможные значения:

* любое натуральное число;
* начальное значение может быть 0;
* конечное значение может быть бесконечностью \*;
* конечное значение не может быть меньше начального.

Пример (см. Рисунок 55. Настройка ограничения мощности связи ).

Отобразить все ТКЕ 1, которые связаны как минимум с двумя ТКЕ 2.

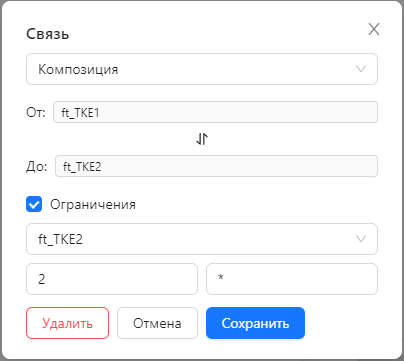


Рисунок 55. Настройка ограничения мощности связи

**Сохранение поиска**

Построенный TQL можно сохранить для дальнейшего использования по кнопке Сохранить (см. Рисунок 56. Сохранение TQL ).

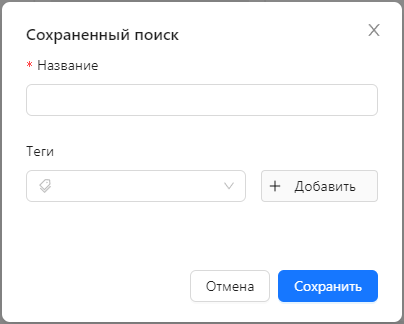


Рисунок 56. Сохранение TQL

#### Навигация по графу СРМ

Основной задачей страницы СРМ является поиск и отображение КЕ и их связей на графе.

Страница СРМ состоит из меню управления и графа отображения СРМ (см. Рисунок 57. Страница СРМ ).

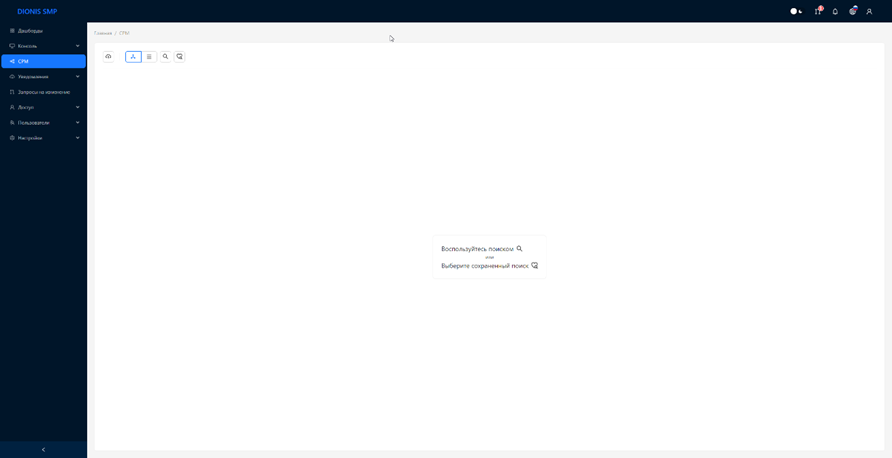


Рисунок 57. Страница СРМ

Меню управления

Основными функциями меню являются (см. Рисунок 58. Кнопка открытия окна импорта ):

* импорт конфигураций КЕ;
* регулировка отображения КЕ и их связей (граф и таблица);
* редактирование фильтра поиска для отображения графа;
* создание и редактирование шаблонов фильтров поиска.

**Импорт конфигураций** (Кнопка “Импорт”) – модальное окно, которое позволяет импортировать конфигурации КЕ и графов с помощью CSV файлов и JSON файлов соответственно. Для того, чтобы импортировать эти конфигурации, необходимо открыть модальное окно кнопкой “Импорт” и выбрать нужный тип конфигурации. Затем нужно выбрать файл, который вы хотите импортировать (см. Рисунок 59. Окно настроек импорта конфигурации ).

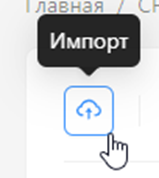


Рисунок 58. Кнопка открытия окна импорта

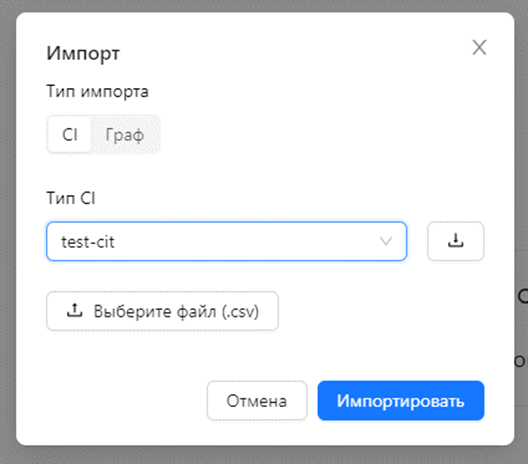


Рисунок 59. Окно настроек импорта конфигурации

Результатом выполнения импорта будет открытие модального окна с информацией об успешном импорте или окно с информацией о конфликте импорта (см. Рисунок 60. Окно успешного импортирования конфигурации и Рисунок 61. Конфликт создания КЕ .):

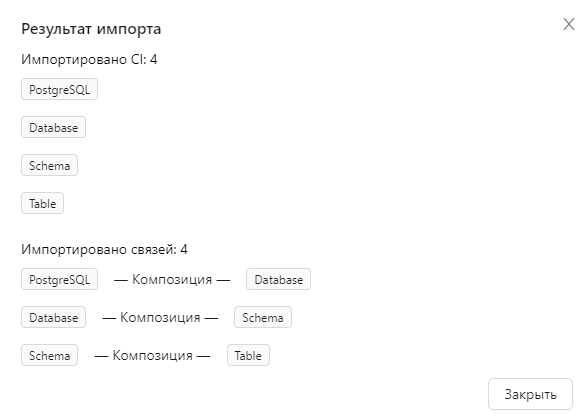


Рисунок 60. Окно успешного импортирования конфигурации

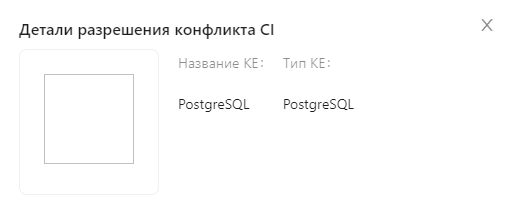


Рисунок 61. Конфликт создания КЕ

**Регулировка отображения КЕ и их связей**. За регулировку отвечает селектор видов – граф и таблица (см. Рисунок 62. Селектор видов ).

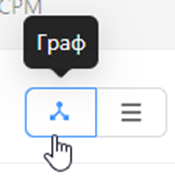


Рисунок 62. Селектор видов

Примеры отображения:

* Граф(см. Рисунок 63. Граф СРМ ):

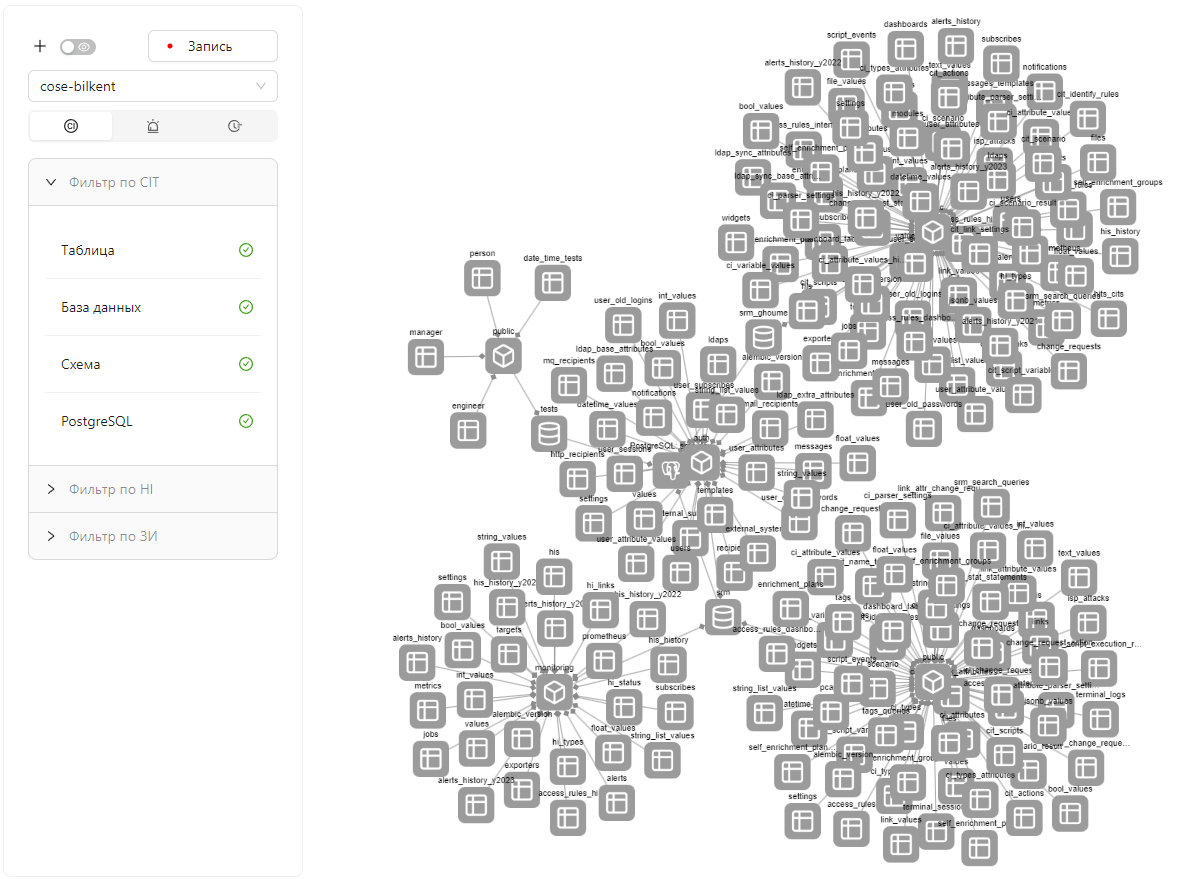


Рисунок 63. Граф СРМ

* Таблица (см. Рисунок 64. Представление СРМ в виде таблицы )

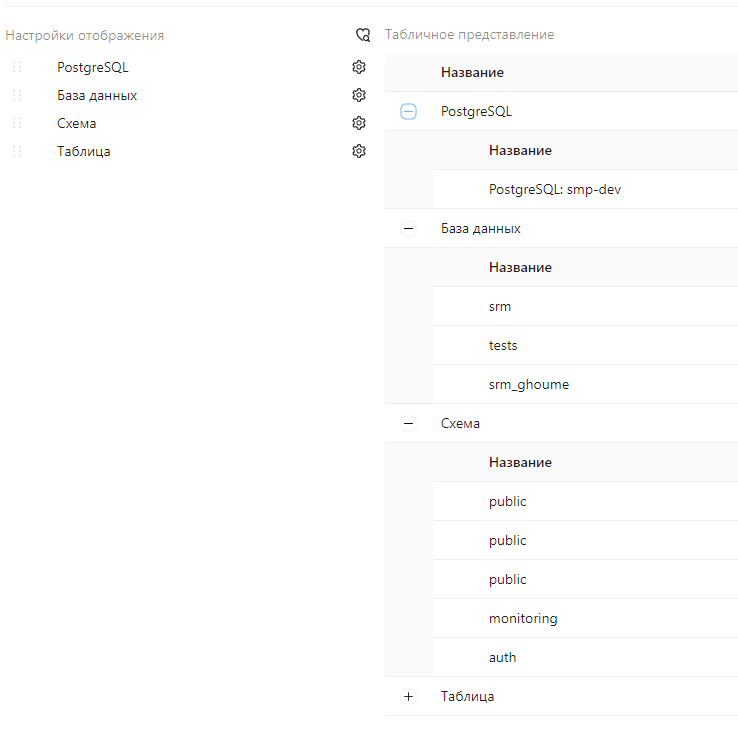


Рисунок 64. Представление СРМ в виде таблицы

**Редактирование фильтра поиска КЕ** (Кнопка “Поиск”) - модальное окно для редактирования TQL фильтра поиска (см. Рисунок 65. Кнопка окна фильтра поиска ).



Рисунок 65. Кнопка окна фильтра поиска

Использование на примере фильтра КЕ по БД (см. Рисунок 66. TQL ).

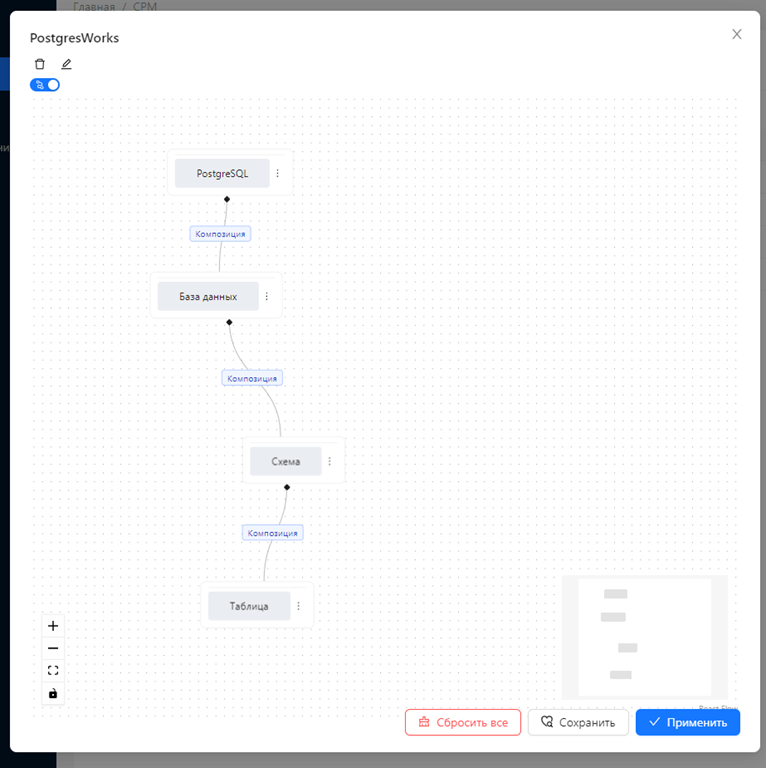


Рисунок 66. TQL

**Создание и редактирование шаблонов фильтра поиска** (Кнопка “Сохраненные поиски”) (см. Рисунок 67. Кнопка открытия окна поисков ):

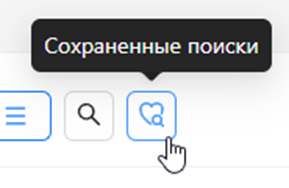


Рисунок 67. Кнопка открытия окна поисков

По нажатию данной кнопки открывается модальное окно со списком сохраненных поисковых TQL запросов. Поиск возможен как по названию запроса, так и по присвоенному ему тегу (см. Рисунок 68. Окно выбора поискового запроса ):

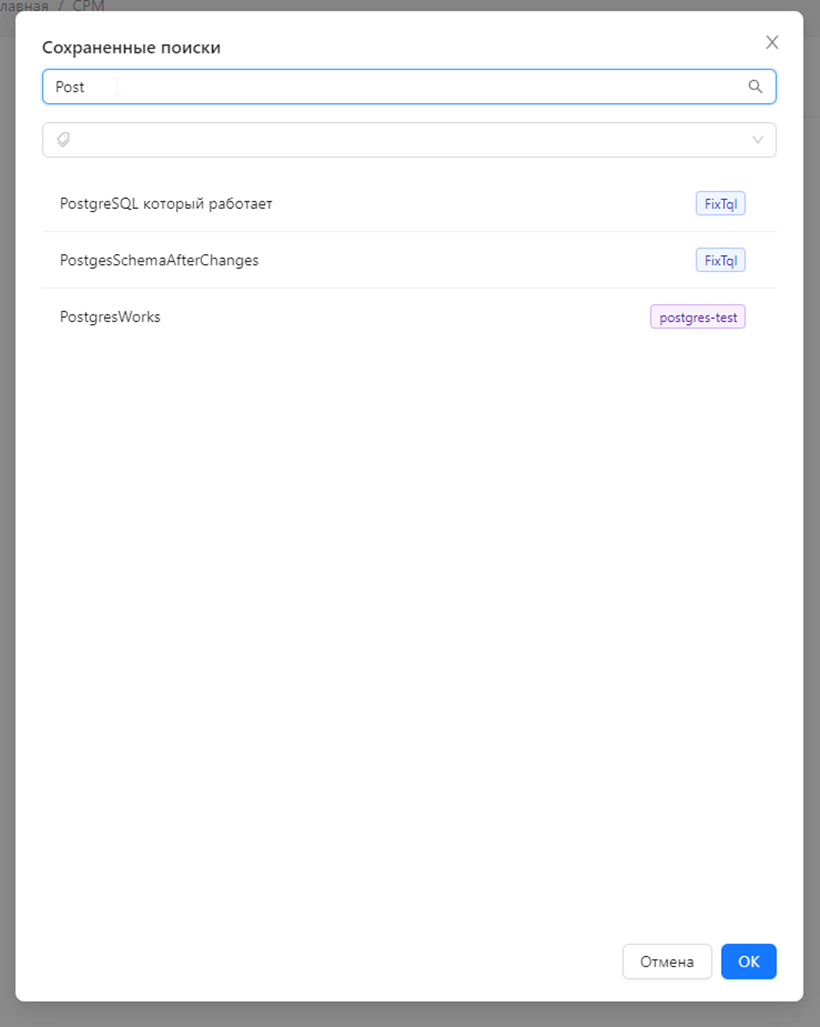


Рисунок 68. Окно выбора поискового запроса

**Граф СРМ**

Данный граф отображает КЕ и их связи на основе TQL запроса (см. Рисунок 69. Граф СРМ ). Это позволяет облегчить поиск и редактирование КЕ, а также упрощает понимание архитектуры связей КЕ за счет наглядного графического представления.

Пример:

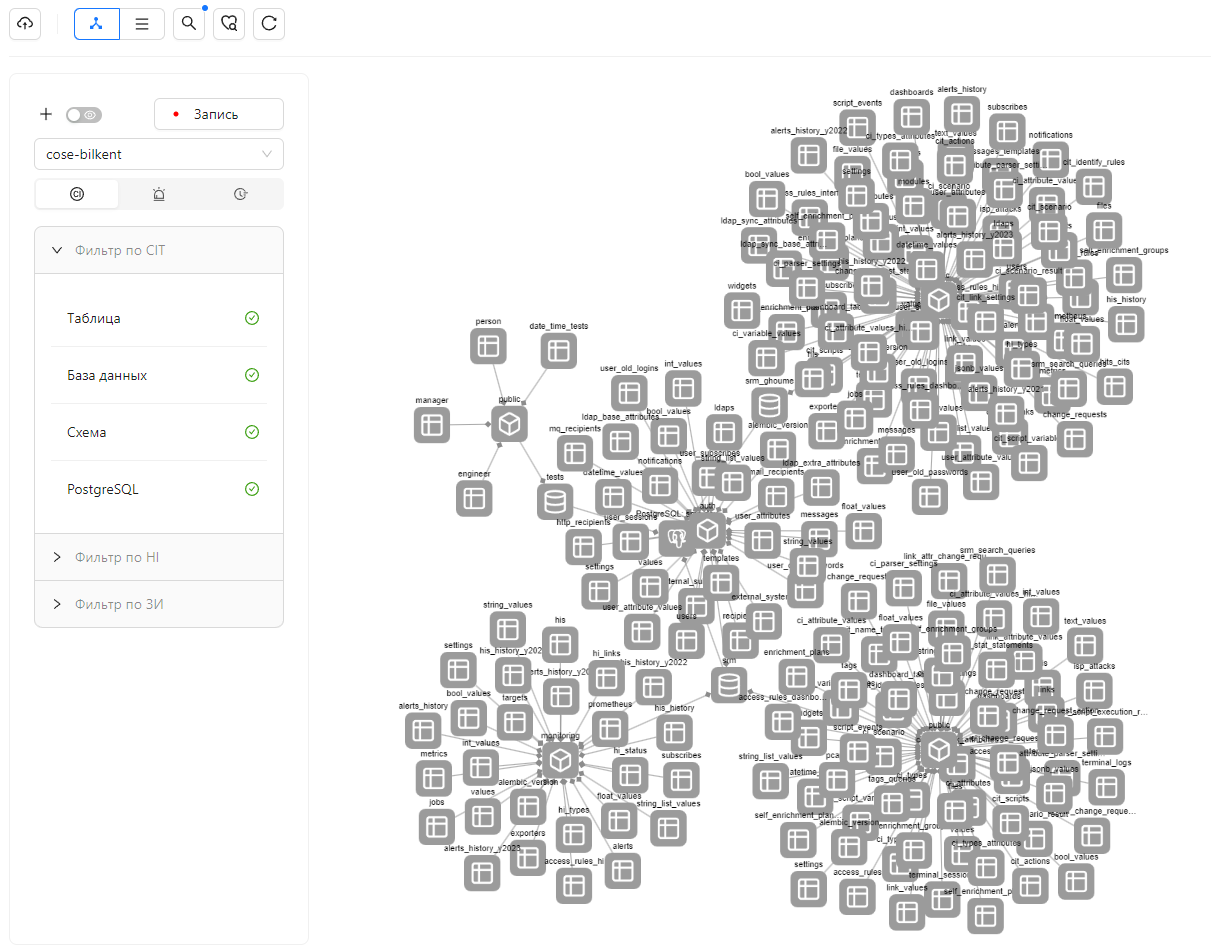


Рисунок 69. Граф СРМ

**Меню управления графом**

Основные функции меню (см. Рисунок 70. Меню управления графом ):

* создание КЕ;
* настройка представления графа;
* изменение схемы отображения КЕ;
* фильтрация КЕ в графе по типам;

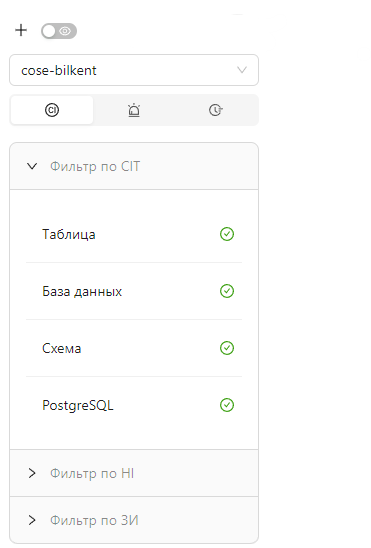


Рисунок 70. Меню управления графом

**Создание КЕ** – модальное окно, позволяющее создать элемент КЕ с заданным типом, композирующим КЕ и атрибутами типа (см. Рисунок 71. Окно создания КЕ ):

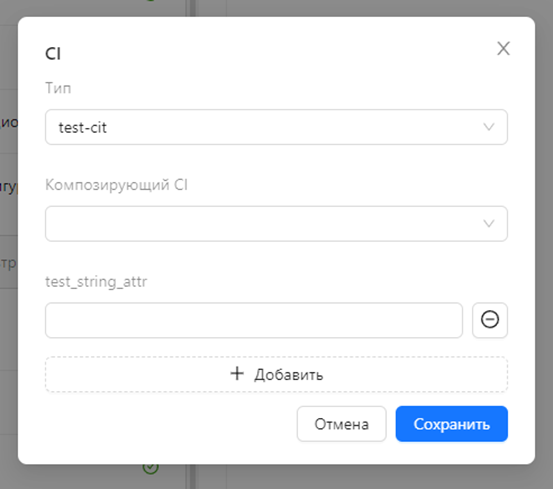


Рисунок 71. Окно создания КЕ

**Настройка представления графа** – селектор, позволяющий пользователю выбрать наиболее удобный вид графа с помощью одного из представленных алгоритмов построения (см. Рисунок 72. Настройка представления графа ).

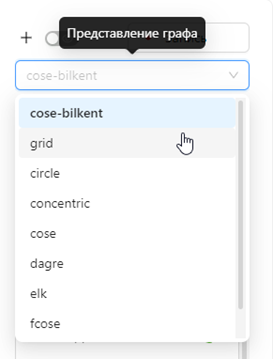


Рисунок 72. Настройка представления графа

**Изменение схемы отображения графа** – селектор схемы отображения. Всего доступно два вида отображения схем:

* схема СРМ (см. Рисунок 73. Схема СРМ );
* схема мониторинга (см. Рисунок 74. Схема мониторинга ).

Примеры:

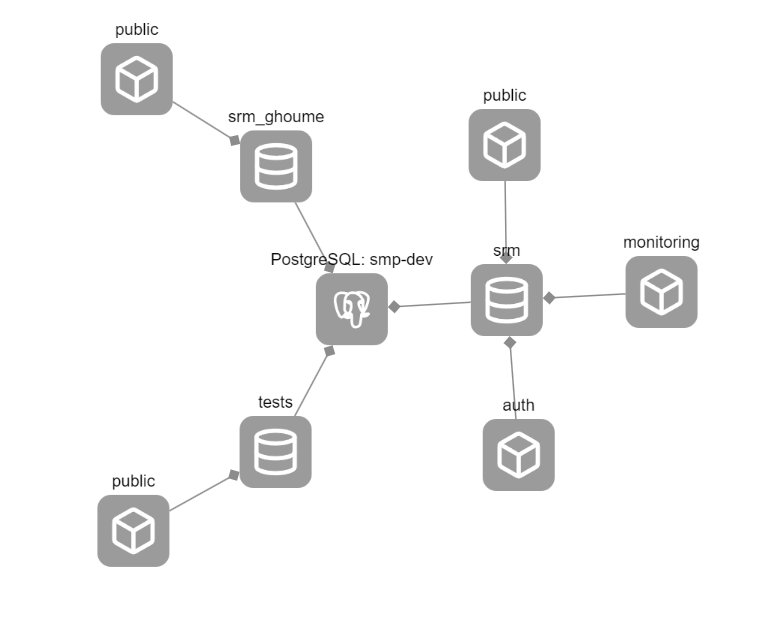


Рисунок 73. Схема СРМ

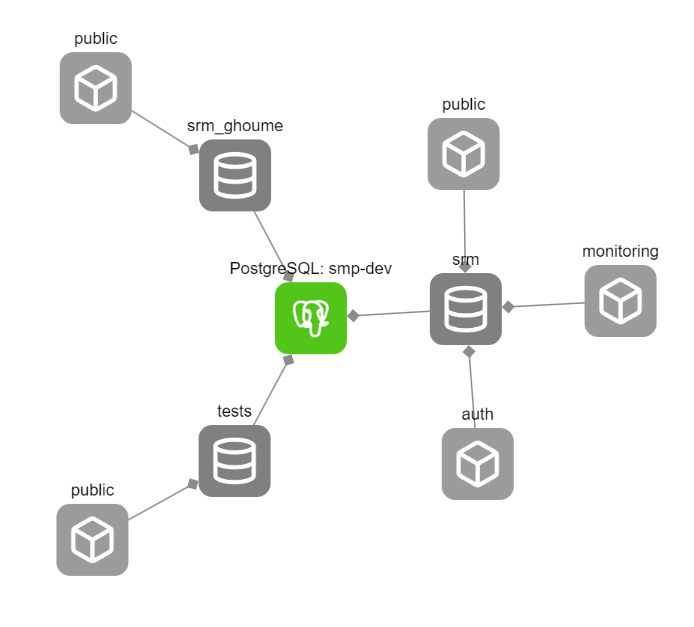


Рисунок 74. Схема мониторинга

**Фильтрация КЕ в графе по типам** – это блок меню, предназначенный для фильтрации КЕ по их типам, типам индикаторов мониторинга жизнеспособности и запросам на изменение (см. Рисунок 75. Зависимость фильтров от типа схемы графа ). Опции фильтров зависят от выбранной схемы отображения графа. Использование фильтров позволяет пользователям быстро находить и настраивать нужные КЕ, а также отслеживать их состояния.

Пример:

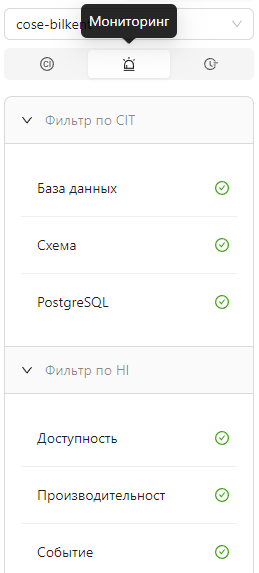
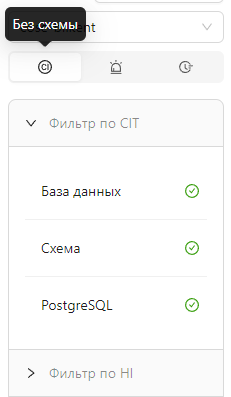


Рисунок 75. Зависимость фильтров от типа схемы графа

Граф СРМ также позволяет редактировать отдельные КЕ и их атрибуты. Для выбора КЕ на графе необходимо нажать по нему левой кнопкой мыши, после этого откроется боковое меню с опциями, такими как изменение названия, добавление или удаление атрибутов, удаление и запуск обогащения (см. Рисунок 76. Меню настройки КЕ ).

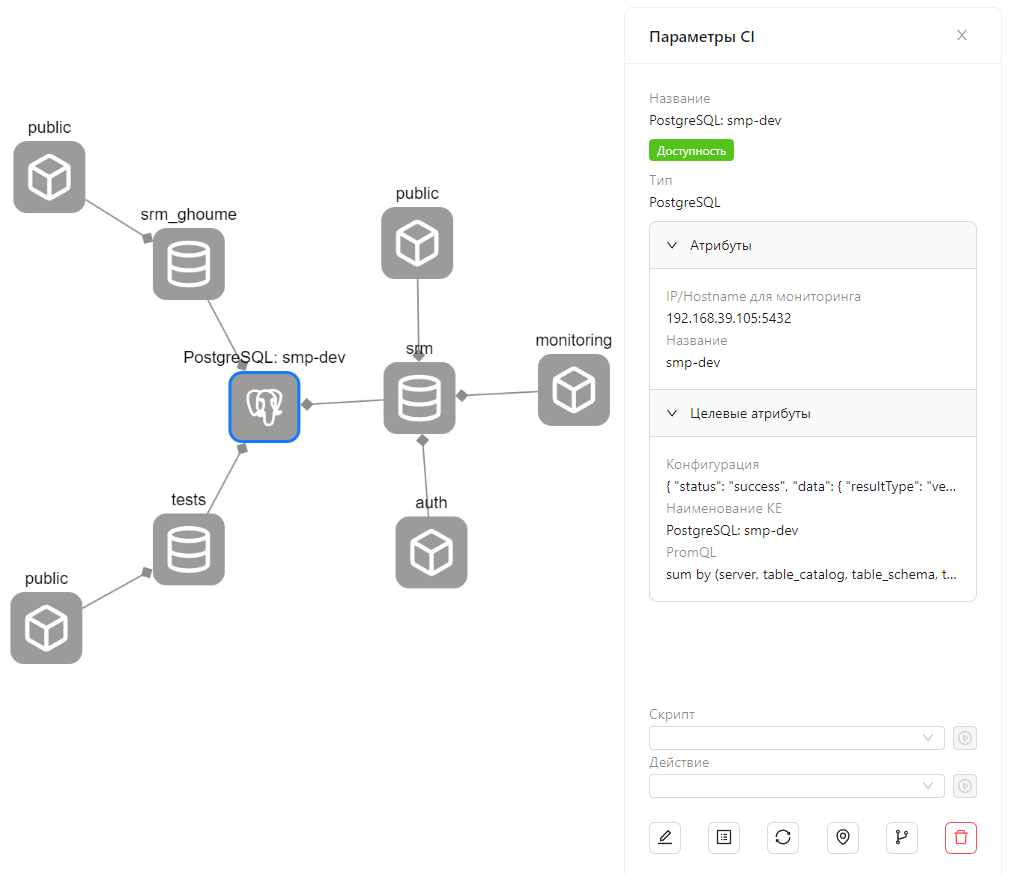


Рисунок 76. Меню настройки КЕ

### Моделирование СРМ

Под моделированием СРМ понимается процесс построения модели предметной области в нотации ТКЕ и доступных связей.

Моделирование делится на этапы:

1. исследование предметной области, выделение объектов контроля и их декомпозиция на значимые части (конфигурационные единицы). При этом перечень объектов контроля не должен быть избыточен;
2. типизация конфигурационных единиц;
3. построение дерева ТКЕ;
4. создание реестра атрибутов и привязка атрибутов к ТКЕ;
5. создание отношений между ТКЕ.

#### Создание атрибутов

При создании атрибута нужно заполнить:

* название;
* ключ;
* целевое использование (необязательно);
* тип;
* схема.

Название и ключ должны быть уникальными.

Ключ может содержать только латинские символы, цифры или символ подчеркивания.

Целевое использование может принимать следующие значения:

* Host for SSH connection to control device - IP адрес или hostname КЕ в сети для подключения системы по протоколу SSH;
* Port for SSH connection - порт КЕ для подключения системы по протоколу SSH;
* Login for SSH authorization - идентификатор учётной записи КЕ для подключения системы по протоколу SSH;
* Password for SSH authorization - пароль учётной записи КЕ для подключения системы по протоколу SSH;
* Fingerprint for SSH connection - цифровой отпечаток КЕ для подключения системы по протоколу SSH, если значение не задано, то оно будет получено при первом подключении системы к КЕ;
* Approved flag for fingerprint - флаг подтверждения цифрового отпечатка КЕ;
* Storage for configuration from device - хранилище текущей (running) конфигурации КЕ;
* Storage for startup configuration from device - хранилище стартовой (startup) конфигурации КЕ;
* Determinate if device is busy – флаг определяет, что на КЕ выполняются задачи поставленные системой;
* Determinate name of CI - значение будет использоваться системой как отображаемое наименование КЕ в СРМ.

Тип может принимать следующие значения:

* строка;
* число;
* дата и время;
* логический;
* список строк;
* JSONB;
* файл.

Схема выбирается из доступных для соответствующего типа, определяет форму редактирования значения атрибута и используется для проверки соответствия значения типу при записи значения в БД.

#### Настройка ТКЕ

##### Создание ТКЕ

Типы конфигурационных единиц – это древовидная структура. Корнем дерева является наиболее общий тип. Листья – наиболее частный и детализированный ТКЕ.

Дочерние ТКЕ наследуют свойства и поведение родительского. По наследству передаются:

* атрибуты (включая настройки);
* связи (включая настройки).

При создании ТКЕ нужно заполнить (см. Рисунок 77. Параметры ТКЕ ):

* название;
* родительский ТКЕ;
* иконка;
* цвет;
* размер иконки;

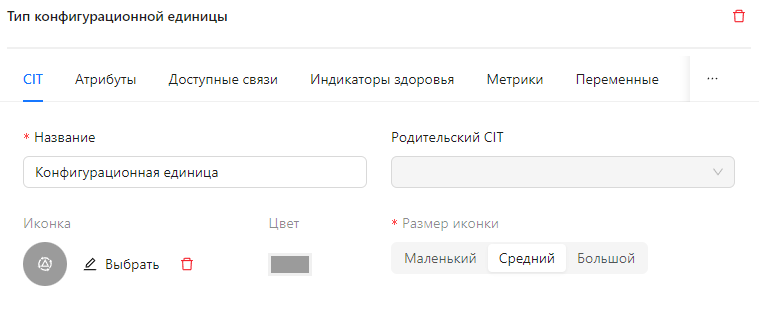


Рисунок 77. Параметры ТКЕ

Родительский ТКЕ нельзя менять после создания.

Иконка и цвет влияют на отображение на графе СРМ.

##### Привязка атрибутов

Атрибуты привязываются к ТКЕ на вкладке «Атрибуты» (см. Рисунок 78. Атрибуты ТКЕ ).

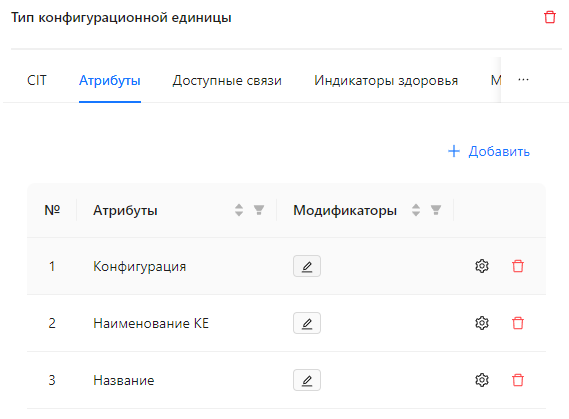


Рисунок 78. Атрибуты ТКЕ

Атрибуты могут быть привязаны к ТКЕ из реестра атрибутов. Атрибут может быть привязан к ТКЕ только один раз. Привязка Атрибута распространяется на всех наследников.

**Модификаторы**

Атрибуты могут имеют модификаторы, которые изменяют поведение атрибута.

* обязательный:

КЕ не может быть создан без заполнения этого атрибута;

* редактируемый:

значение атрибута может редактироваться пользователем;

* скрытый:

атрибут не будет показан пользователю;

* сохранять историю:

ПАМИР будет хранить историю изменения значения атрибута;

* изменить схему атрибута:

позволяет переопределить json схему атрибута.

#### Реестр индикаторов здоровья

Для работы с реестром индикаторов здоровья необходимо выбрать в основном меню: Главная > Настройки > Мониторинг > Индикаторы здоровья (см. Рисунок 79. Меню работы с Индикаторами здоровья ):

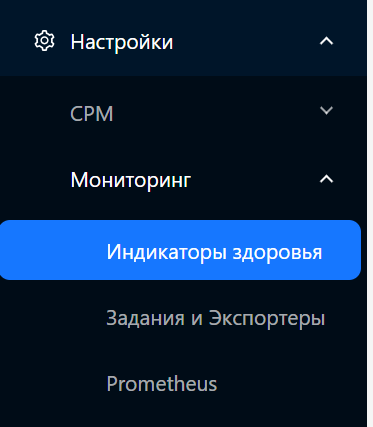


Рисунок 79. Меню работы с Индикаторами здоровья

Реестр индикаторов (см. Рисунок 80. Реестр индикаторов здоровья ) представляет собой список типов индикаторов, которые используются для обобщения качественных характеристик состояния объекта контроля и, при необходимости, указания шаблона конфигурирования alert prometheus.

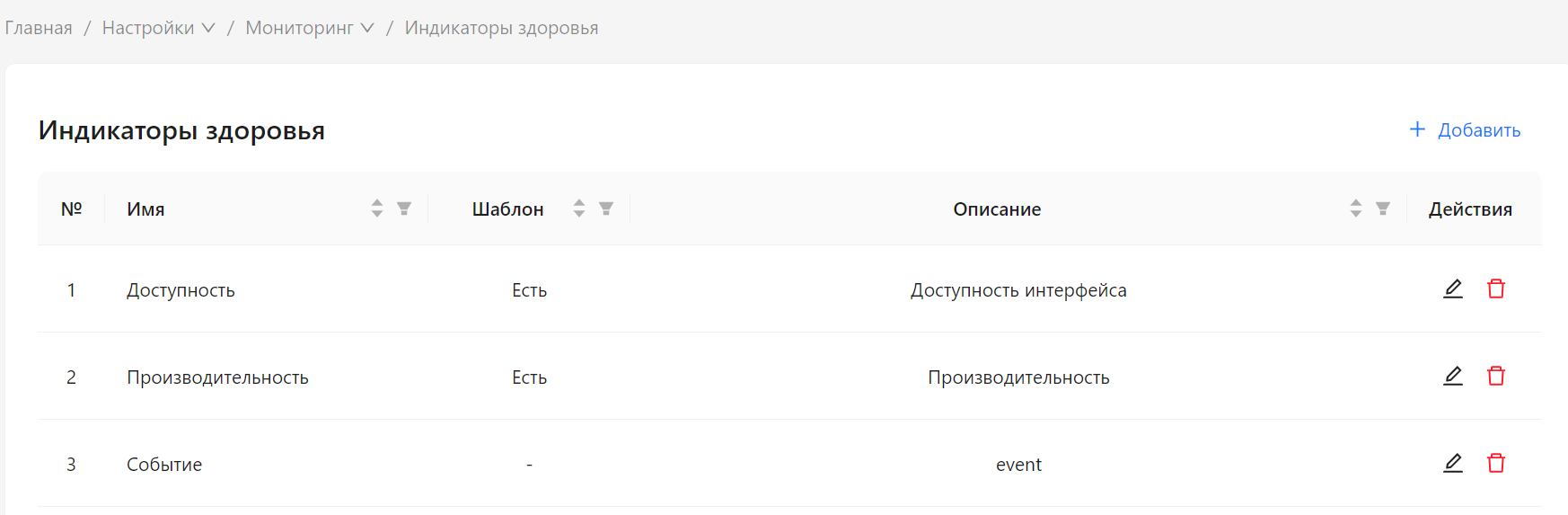


Рисунок 80. Реестр индикаторов здоровья

Описание представленных полей:

* имя – краткое наименование качественной характеристики;
* шаблон – флаг, указывающий на наличие шаблона конфигурации для контролирования необходимого качественного состояния (шаблон может быть переопределен при настройке конкретного объекта контроля);
* описание – пояснение к качественной характеристике;
* действия – кнопки для изменения полей или удаления индикатора.

К индикаторам здоровья применимы действия: добавить, изменить и удалить.

При добавлении индикаторов (см. Рисунок 81. Добавление индикатора здоровья ) необходимо заполнить обязательные поля Имя и Описание. Поле шаблон не обязательно для заполнения в реестре, но может быть использовано для указания конфигурации, которую при необходимости можно использовать в неизменном виде, уточнить или заменить при назначении индикатора конкретному объекту контроля. Содержимое шаблона, указываемое в реестре, валидируется только на соответствие формату yaml.



Рисунок 81. Добавление индикатора здоровья

При изменении возможно указание любых новых значений для полей индикатора. Значения из реестра индикаторов здоровья не используются для конфигурирования prometheus (только для отображения в интерфейсе) и могут быть изменены в любой момент времени без последствий для мониторинга.

При удалении индикатор будет исключен из реестра. Удаление индикатора здоровья повлечет за собой удаление всех настроенных индикаторов на объектах контроля, связанных с удаляемым типом. Предупреждение об использовании индикатора здоровья в каких-либо объектах контроля не предусмотрено.

#### Привязка метрик

Для привязки метрик необходимо выбрать в меню раздел с настройкой типов КЕ: Главная > Настройки > СРМ > Типы КЕ CIT (Рисунок 23. Меню выбора типа КЕ), выбрать требуемый объект контроля (см. Рисунок 82. Меню выбора типа КЕ и Рисунок 83. Выбор объекта контроля ) , перейти к вкладке Метрика (см. Рисунок 84. Вкладка с настройками метрики )

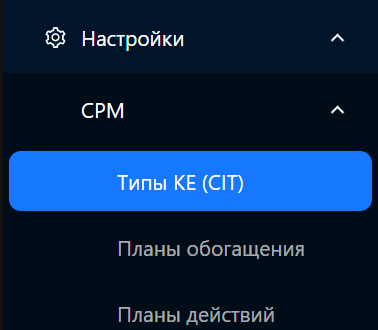


Рисунок 82. Меню выбора типа КЕ

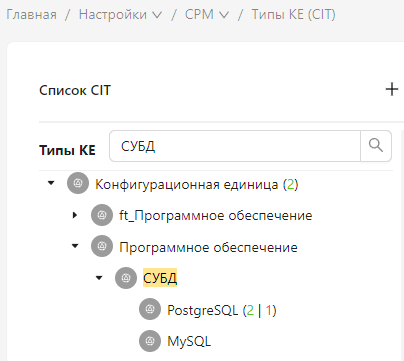


Рисунок 83. Выбор объекта контроля

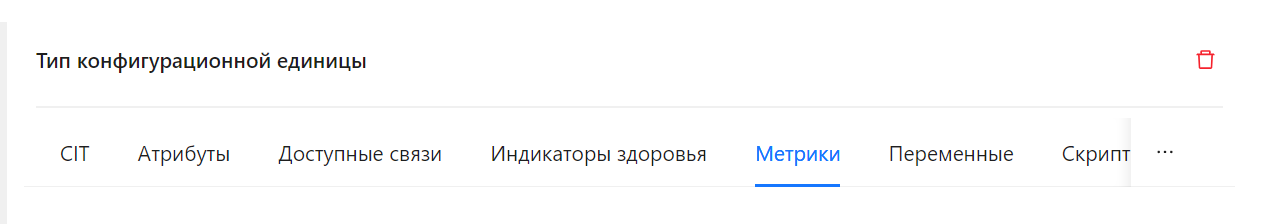


Рисунок 84. Вкладка с настройками метрики

Привязка метрик (см. Рисунок 85. Настройка метрик ) представляет собой процесс конфигурирования, в результате которого для объекта контроля назначается задание (способ сбора метрик) и шаблон (идентификаторы, применяемые к метрике).

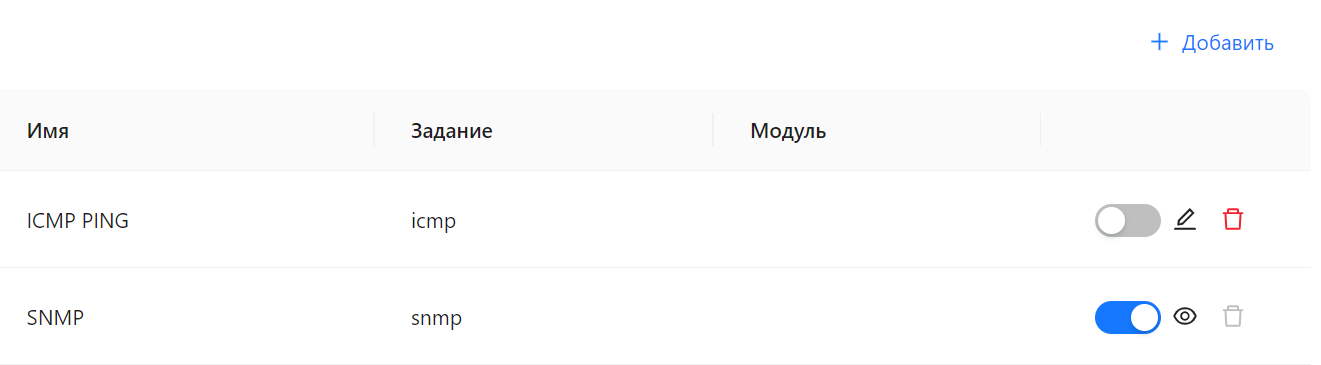


Рисунок 85. Настройка метрик

Описание представленных полей:

* имя – краткое наименование метрики;
* задание – выбранный способ сбора метрик;

К метрике применимы действия: добавить, включить/выключить, просмотреть/изменить и удалить.

При добавлении метрики (см. Рисунок 86. Добавление метрики ) необходимо заполнить обязательные поля Имя, Задание, Шаблон и при необходимости фильтр.

Имя – произвольное наименование для идентификации метрики, должно быть уникально для типа КЕ.

Задание представляет собой выпадающий список (см. Рисунок 87. Выбор задания ) из которого выбирается необходимый способ сбора метрик.

В поле шаблон представлен образец в формате yaml для конфигурирования метрики в prometheus. Шаблон корректируется под конкретные потребности, указываются необходимые идентификаторы метрики (label) и откуда ее собирать (target). В качестве значений могут быть использованы переменные (см. 4.1.1.10.1) или конкретные значения (будут общими для всех КЕ текущего типа). Содержимое шаблона, валидируется на соответствие формату ожидаемому prometheus (наличие ключей label и targe обязательно) и наличие у типа КЕ указанных переменных.

Фильтр (см. Рисунок 88. Фильтр метрики ) используется для уточнения КЕ, к которым будут применяться указанные настройки метрики. Если фильтр не выбран, метрика будет привязана ко всем КЕ текущего типа. При включении фильтра метрика будет привязана только к КЕ (текущего типа), полученным в результате выполнения построенного в окне TQL.

После добавления (изменения) метрики текущая конфигурация сохраняется в базе данных приложения, метрики не собираются. При включении переключателя (), конфигурация указанная в шаблоне, заполняется конкретными значениями, передается в prometheus и начинается сбор метрик. Несмотря на наличие переменных в шаблоне, конкретные значения у КЕ могут отсутствовать (быть не заполненными), такие КЕ будут пропускаться, и сбор метрик по ним осуществляться не будет. Если при включенной метрике будет добавлена КЕ для текущего типа (и она попадает в фильтр, если он включен) то она автоматически будет добавлена в конфигурацию prometheus. При выключении переключателя конфигурация удаляется из prometheus и сбор метрик прекращается.



После включения метрики возможно просмотреть () используемые настройки. Внесение изменений () в конфигурацию допускается только при выключенной метрике. Изменять можно любые параметры метрики (см. Рисунок 86. Добавление метрики ).



Удаление метрики () возможно только у выключенной метрики.

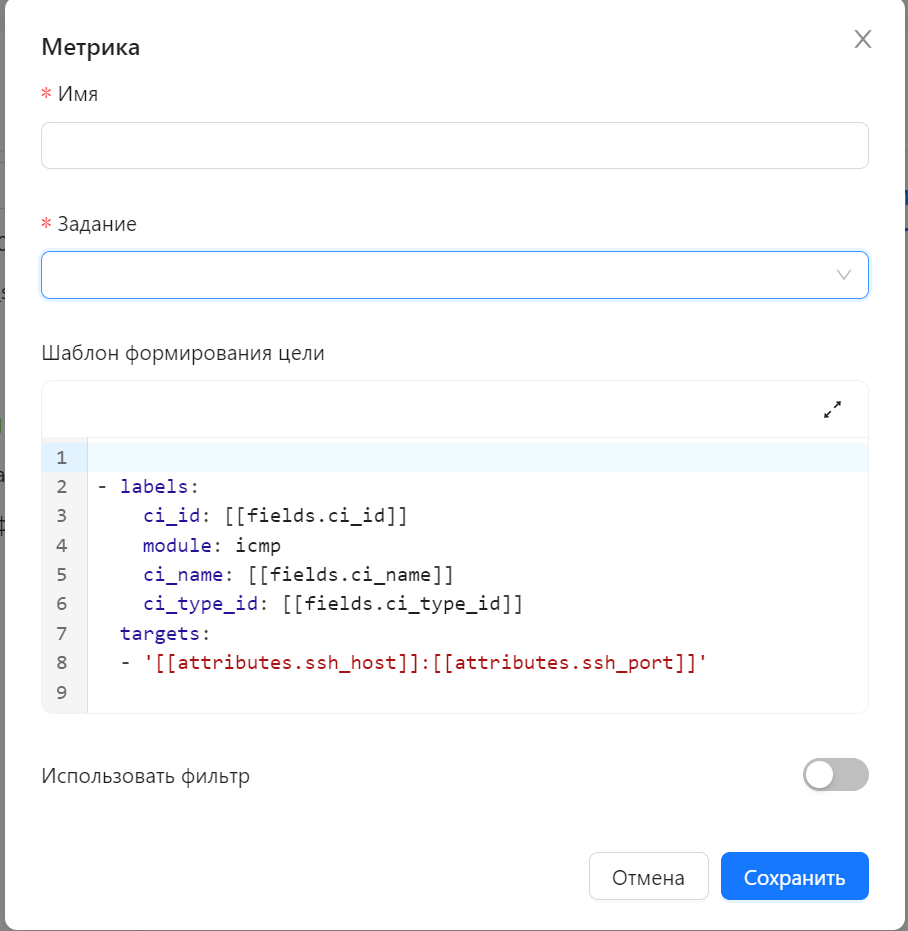


Рисунок 86. Добавление метрики

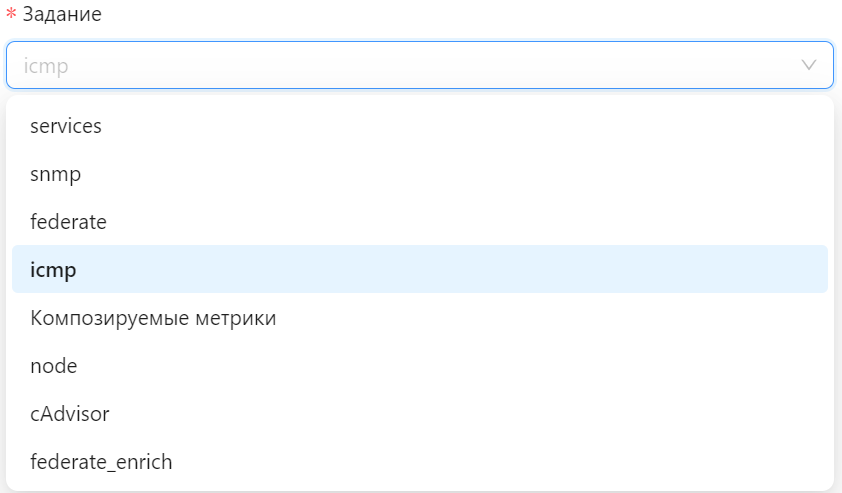


Рисунок 87. Выбор задания

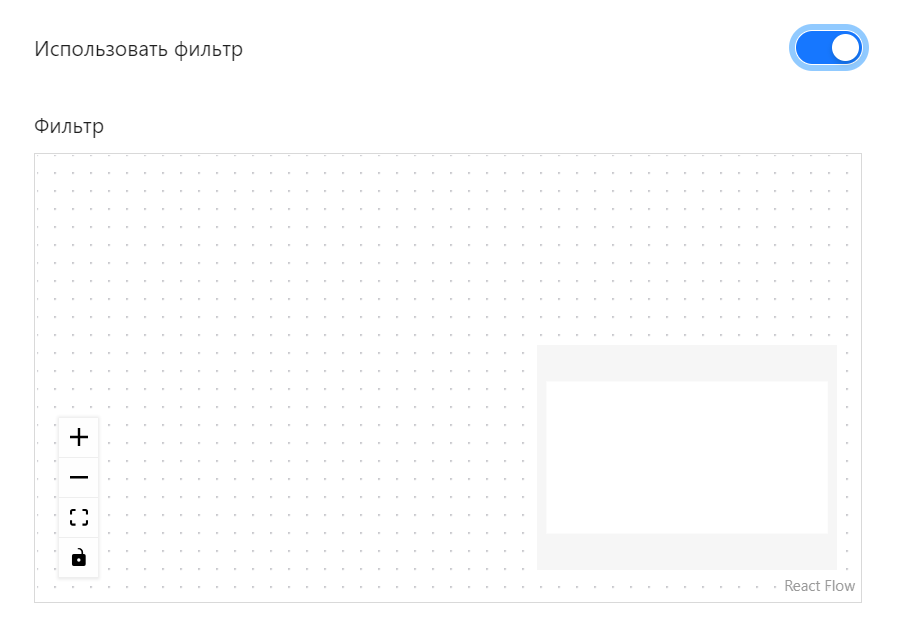


Рисунок 88. Фильтр метрики

#### Привязка индикаторов здоровья

Для привязки индикаторов необходимо выбрать в меню раздел с настройкой типов КЕ: Главная > Настройки > СРМ > Типы КЕ CIT (см. Рисунок 82. Меню выбора типа КЕ ), выбрать требуемый объект контроля (см. Рисунок 83. Выбор объекта контроля ), перейти к вкладке индикаторы здоровья (см. Рисунок 89. Вкладка с настройками индикаторов здоровья ):

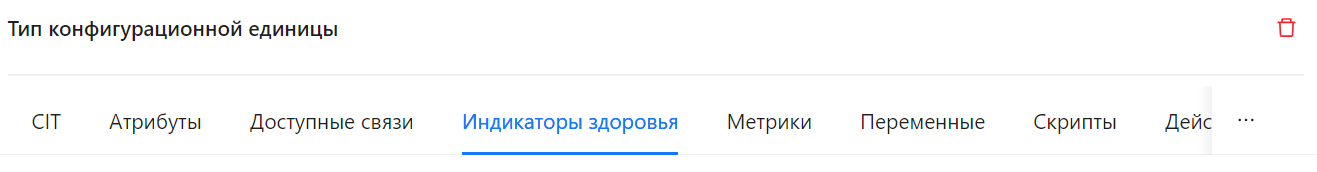


Рисунок 89. Вкладка с настройками индикаторов здоровья

Привязка индикаторов здоровья (см. Рисунок 90. Настройка индикаторов здоровья ) представляет собой процесс конфигурирования, в результате которого для объекта контроля назначается тип индикатора (обобщенное название, идентифицирующее характеристику, по которой оценивается состояние КЕ), тип (способ получения данных характеризующих состояние) и дополнительные данные, уточняющие характер оценки.

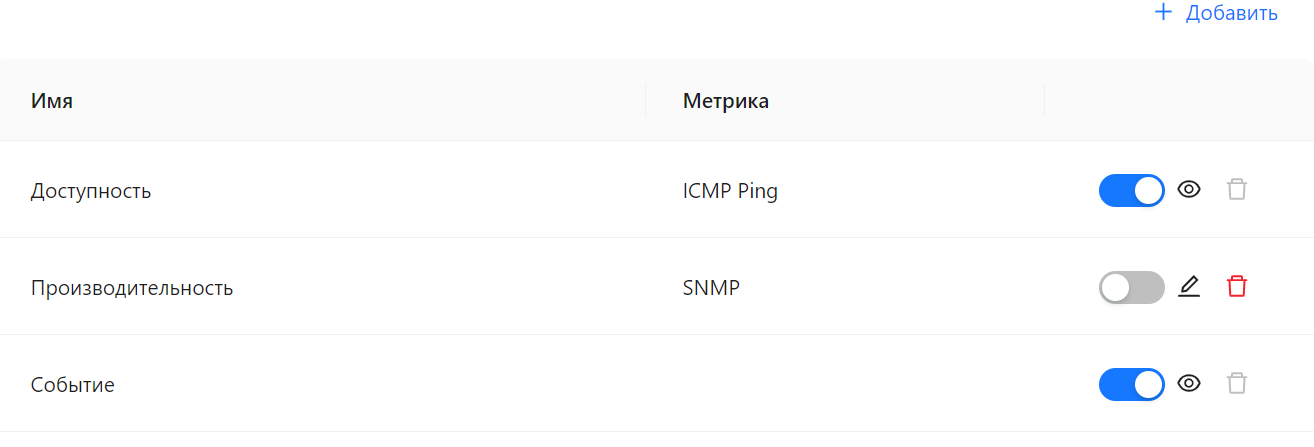


Рисунок 90. Настройка индикаторов здоровья

Описание представленных полей:

* имя – выбранный тип из реестра индикаторов здоровья;
* метрика – источник данных (если данные получаются от внешних источников).

К индикаторам здоровья применимы действия: добавить, включить/выключить, просмотреть/изменить и удалить.

При добавлении индикатора здоровья к объекту контроля (см. Рисунок 91. Добавление индикатора здоровья ) необходимо выбрать индикатор здоровья из реестра индикаторов (см. Рисунок 92. Выбор индикатора здоровья из реестра индикаторов ) и способ получения данных для индикатора (см. Рисунок 93. Выбор типа индикатора здоровья ) . Возможны три типа индикаторов: Индикатор метрики (получение данных на основе собранных метрик), Расчетный (получение данных на основе состояния других КЕ), Событие (получение данных на основе логов из fluent-bit).

Для настройки типа «Индикатор метрики» (см. Рисунок 94. Настройка индикатора метрики ) необходимо заполнить правило, по которому будет оцениваться состояние индикатора и метрику (источник данных) (см. Рисунок 95. Выбор правила для индикатора здоровья ). Базовый шаблон будет подставлен из шаблона, указанного в реестре для выбранного индикатора. Поле правило заполняется в формате yaml для конфигурирования alert в prometheus, указывается условие при котором для индикатора устанавливается значение severity. В шаблоне, в нотации prometheus могут использоваться label, указанные при настройки выбранной метрики. Содержимое шаблона валидируется на соответствие формату yaml и наличие ключа expr с условием. Можно указать несколько правил с различным severity.

Для настройки типа «Расчетный» (см. Рисунок 96. Настройка расчетного индикатора ) необходимо выбрать тип(ы) КЕ, по которым будет оцениваться состояние индикатора. Типы КЕ, состояние которых можно использовать в текущем индикаторе, будут предложены из имеющихся в СРМ связей для текущего типа КЕ и наличия у связанных типов КЕ такого же типа индикатора здоровья (см. Рисунок 93. Выбор типа индикатора здоровья ) . Итоговый результат будет зависеть от значения выбранного в поле «Выражение» (см. Рисунок 96. Настройка расчетного индикатора ).

Для настройки типа «Событие» (см. Рисунок 97. Настройка индикатора события ) необходимо указать путь (url), по которому будут получаться события, наименование поля, в котором содержится интересующее значение логирования и регулярные выражения для существующих уровней критичности (могут быть заполнены на выбор). При нахождении совпадения регулярного выражения со значением из поля критичности индикатор установится в соответствующее значение. Парсинг происходит до первого совпадения от низшего уровня к высшему. URL представляет собой последнюю часть пути в ручке POST /event/{url} куда fluent-bit будет отправлять события.

Фильтр работает по такому же принципу, как и у фильтра метрики (см. Рисунок 88. Фильтр метрики ) и используется для уточнения КЕ, к которым будут применяться указанные настройки индикаторов. Если фильтр не выбран, индикатор будет привязан ко всем КЕ текущего типа. При включении фильтра индикатор будет привязан только к КЕ (текущего типа), полученным в результате выполнения построенного в окне TQL.

После добавления (изменения) индикатора здоровья текущая конфигурация сохраняется в базе данных приложения, индикаторы состояния отображаться не будут. При включении переключателя (), всем КЕ текущего типа (или по фильтру) присваивается индикатор настроенного типа; конфигурация, указанная в правиле, передается в prometheus, запускается процесс оценки состояния в соответствии с настроенными правилами.



Далее значение расчетного индикатор будет установлено в соответствии с получаемыми значениями (если получаемых значений нет, индикатор на КЕ не отобразится, например, индикатор настроен на получение значений с не настроенного индикатора). Индикатор события отобразится на КЕ только после получения первого значения. Индикатор метрики, при наличии нескольких правил, отобразится на КЕ со значением 1, при наличии одного правила отобразится со значением 3, если в severity правила указано 2, и значением 3 при любом другом значении в severity. Если при включенном индикаторе будет добавлена КЕ для текущего типа (и она попадает в фильтр, если он включен) то ему автоматически будет назначен индикатор и установлено значение в соответствии с описанной выше логикой. При выключении переключателя конфигурация удаляется из prometheus, индикаторы у КЕ удаляются.

После включения индикатора здоровья возможно просмотреть () используемые настройки. Внесение изменений () в конфигурацию допускается только при выключенном индикаторе. Изменять можно любые параметры индикатора (см. добавление индикатора).



Удаление индикатора() возможно только в выключенном состоянии.

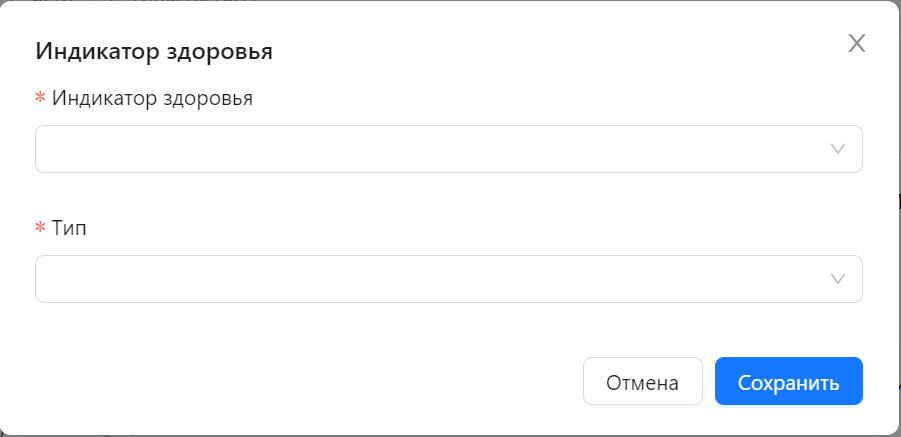


Рисунок 91. Добавление индикатора здоровья

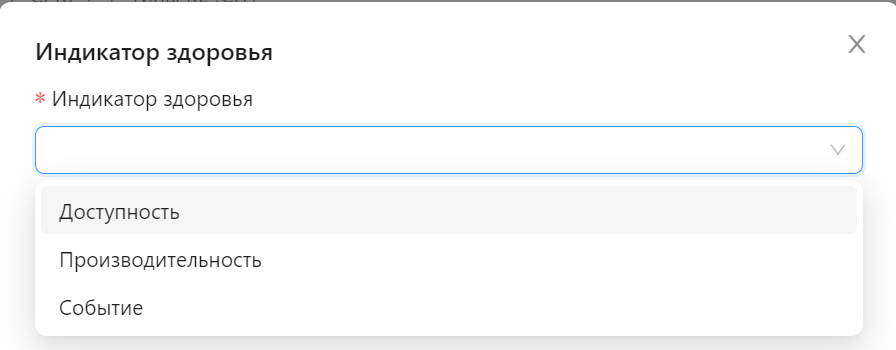


Рисунок 92. Выбор индикатора здоровья из реестра индикаторов

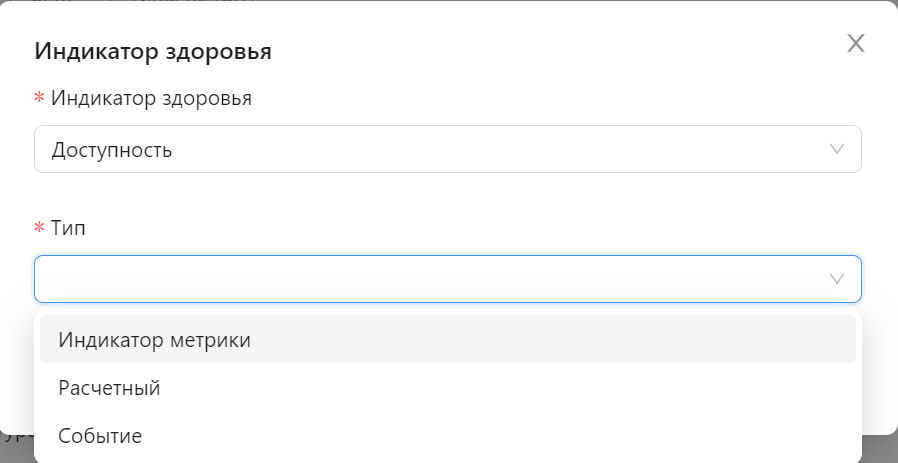


Рисунок 93. Выбор типа индикатора здоровья

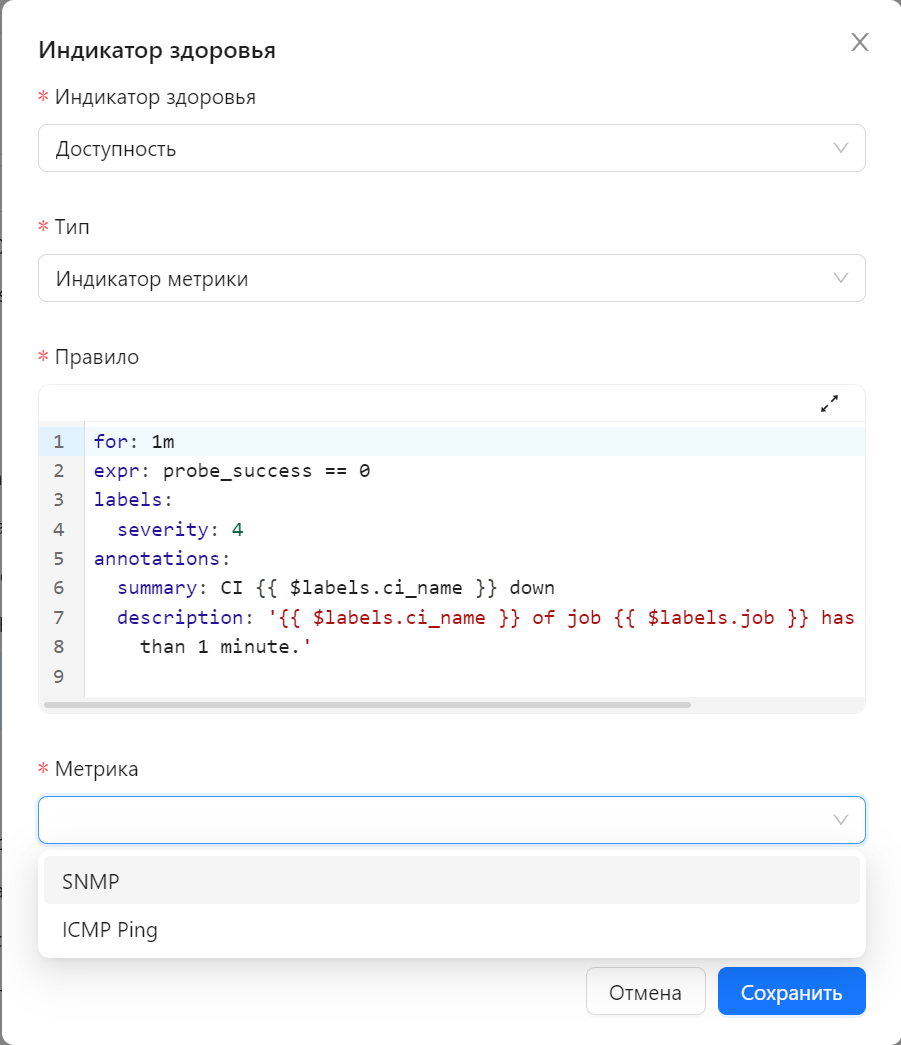


Рисунок 94. Настройка индикатора метрики

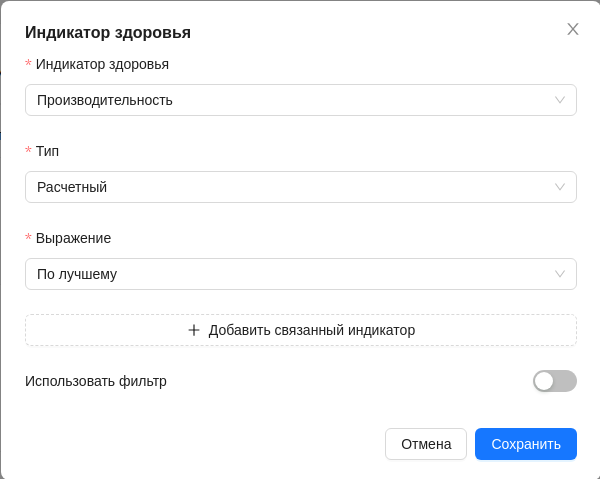


Рисунок 95. Выбор правила для индикатора здоровья

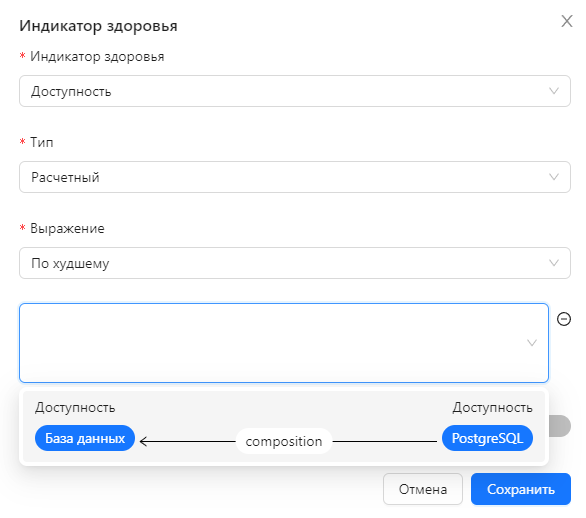


Рисунок 96. Настройка расчетного индикатора

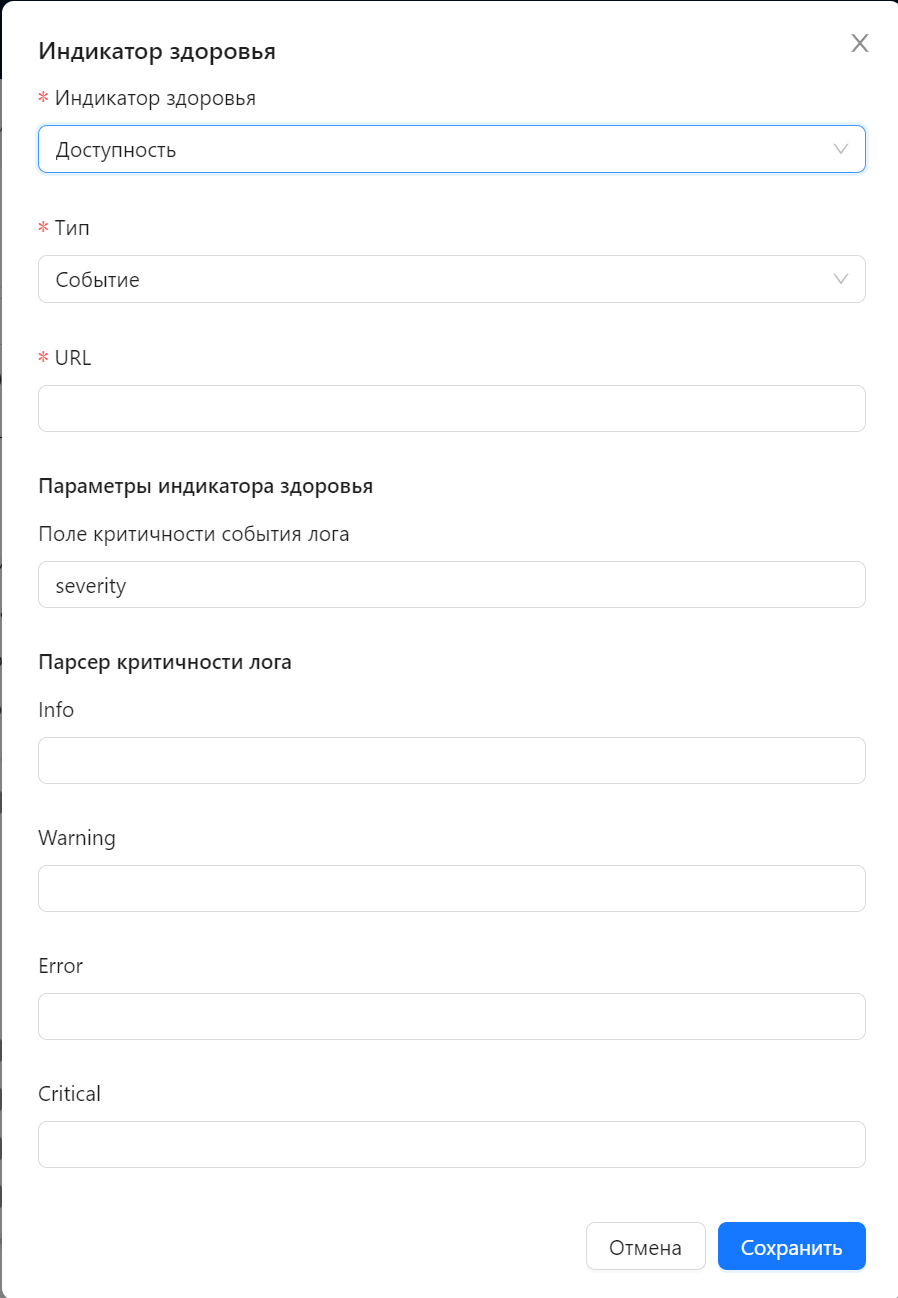


Рисунок 97. Настройка индикатора события

### Наполнение СРМ

#### Мануальное создание КЕ

Для создания КЕ нужно на графе СРМ вызвать контекстное меню и выбрать пункт «Добавить». Выбрать тип добавляемой КЕ; если это композируемая КЕ, то выбрать композирующую КЕ. Добавить атрибуты, значения которых необходимо задать при создании КЕ. Обязательно нужно задать значение атрибута, который используется для идентификации КЕ в СРМ.

#### Импорт КЕ

Для импорта КЕ в систему нужно на графе СРМ выбрать элемент «Импорт». Возможен импорт КЕ в систему из файла в формате csv и графа из файла в формате json.

###### Импорт КЕ из файла в формате csv

Из файла в формате csv можно импортировать КЕ одного типа. Для импорта нужно выбрать тип импортируемого КЕ, скачать шаблон файла. Каждая строка в файле соответствует одному КЕ. В строке через запятую перечисляются значения атрибутов КЕ. Нужно задать значения необходимых атрибутов. Обязательно нужно задать значение атрибута, который используется для идентификации КЕ в СРМ. Отредактированный файл нужно импортировать в СРМ. В случае успеха появится сообщение со списком импортируемых КЕ.

###### Импорт графа из файла в формате json

Для импорта графа из файла в формате json нужно подготовить файл, в котором описать КЕ с атрибутами и связи между КЕ. В файле должно быть два списка объектов: «nodes» и «links».

Импортируемые КЕ в списке «nodes» необходимо представить в виде словарей с ключами:

* relative\_id - относительный идентификатор, однозначно определяет КЕ в рамках файла;
* ci\_type\_id - идентификатор типа КЕ в СРМ;
* attribute\_values - список значений атрибутов КЕ.

Значения атрибутов КЕ в списке «attribute\_values» необходимо представить в виде словарей с ключами:

* attribute\_name - ключ атрибута;
* value - значение атрибута.

Для каждого КЕ обязательно нужно задать значение атрибута, который выбран для идентификации КЕ в СРМ.

Связи между КЕ в списке «links» необходимо необходимо представить в виде словарей с ключами:

* relative\_src\_id - относительный идентификатор КЕ источника связи;
* relative\_dst\_id - относительный идентификатор КЕ назначения связи;
* link\_name - наименование связи;
* attribute\_values - список значений атрибутов связи, заполняется аналогично соответствующему списку КЕ.

Наименование связи определяет тип связи и может принимать следующие значения:

* aggregation - агрегация;
* association - ассоциация;
* composition - композиция.

Если импортируемый КЕ в СРМ описан как композируемый, то в графе обязательно должен быть представлен композирующий КЕ.

Например ,граф может быть описан следующим образом:

{

"nodes": [

{

"relative\_id": "1",

"ci\_type\_id": "23080722-dfba-4bd8-b32a-f59c05c584d4",

"attribute\_values": [

{

"attribute\_name": "name",

"value": "PostgreSQL"

}

]

},

{

"relative\_id": "2",

"ci\_type\_id": "af412929-e153-4746-a273-5d26ea160065",

"attribute\_values": [

{

"attribute\_name": "name",

"value": "Database"

}

]

},

{

"relative\_id": "3",

"ci\_type\_id": "cc79aff1-2806-429d-bd80-b2d567e0498b",

"attribute\_values": [

{

"attribute\_name": "name",

"value": "Schema"

}

]

},

{

"relative\_id": "4",

"ci\_type\_id": "87ddc434-5d7f-4938-8a92-ea8c49ee56dc",

"attribute\_values": [

{

"attribute\_name": "name",

"value": "Table"

}

]

}

],

"links": [

{

"relative\_src\_id": "1",

"relative\_dst\_id": "2",

"link\_name": "composition"

},

{

"relative\_src\_id": "2",

"relative\_dst\_id": "3",

"link\_name": "composition"

},

{

"relative\_src\_id": "3",

"relative\_dst\_id": "4",

"link\_name": "composition"

}

]

}

После его импорта появится окно с результатами (см. Рисунок 98. Результат импорта ):

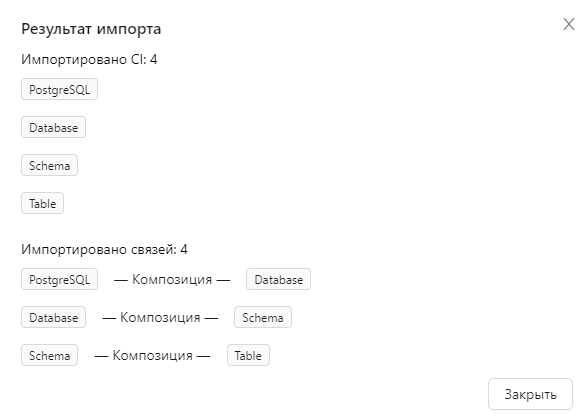


Рисунок 98. Результат импорта

#### Добавление КЕ через обнаружение

Обнаружение КЕ выполняется во время работы механизма обогащения. Механизм обогащения получает конфигурацию (фетчеры), разбирает ее на КЕ (парсеры) с заполнением значений атрибутов и созданием необходимых связей. Далее, полученные в результате разбора КЕ сравниваются с имеющимися в базе данных и отсутствующие в базе данных КЕ (и связи) создаются, имеющиеся обновляются актуальными значениями, не обнаруженные механизмом обогащения, но существующие в базе данных, удаляются.

Для получения конфигурации используются фетчеры, назначаемые атрибутам и связям. Фетчеры в соответствии со своим предназначением и настройками, выполняют различные действия (подключение по ssh, запрос по http и т.д.) и возвращают результат в виде строки. Полученная строка может быть обработана парсером или сохранена в атрибуте в неизменном виде.

Парсеры используются для извлечения из результатов работы фетчеров полезной информации. Результаты работы парсера сохраняется в атрибуте в качестве значения.

Механизм обогащения может быть запущен на КЕ вручную или с использованием планировщика.

Для выполнения обогащения необходимо произвести настройку ТКЕ (см. Рисунок 99. Необходимая настройка на ТКЕ для выполнения обогащения ):

* определить для атрибутов источник получения значения (фетчер) и, при необходимости, обработчик (парсер);
* определить связи, по которым КЕ связывается со своими детьми, установить источник получения конфигурации для связи и, при необходимости, обработчик;
* при построении связей, отличных от композиции, дополнительно можно указать TQL запрос для выбора связанных КЕ.

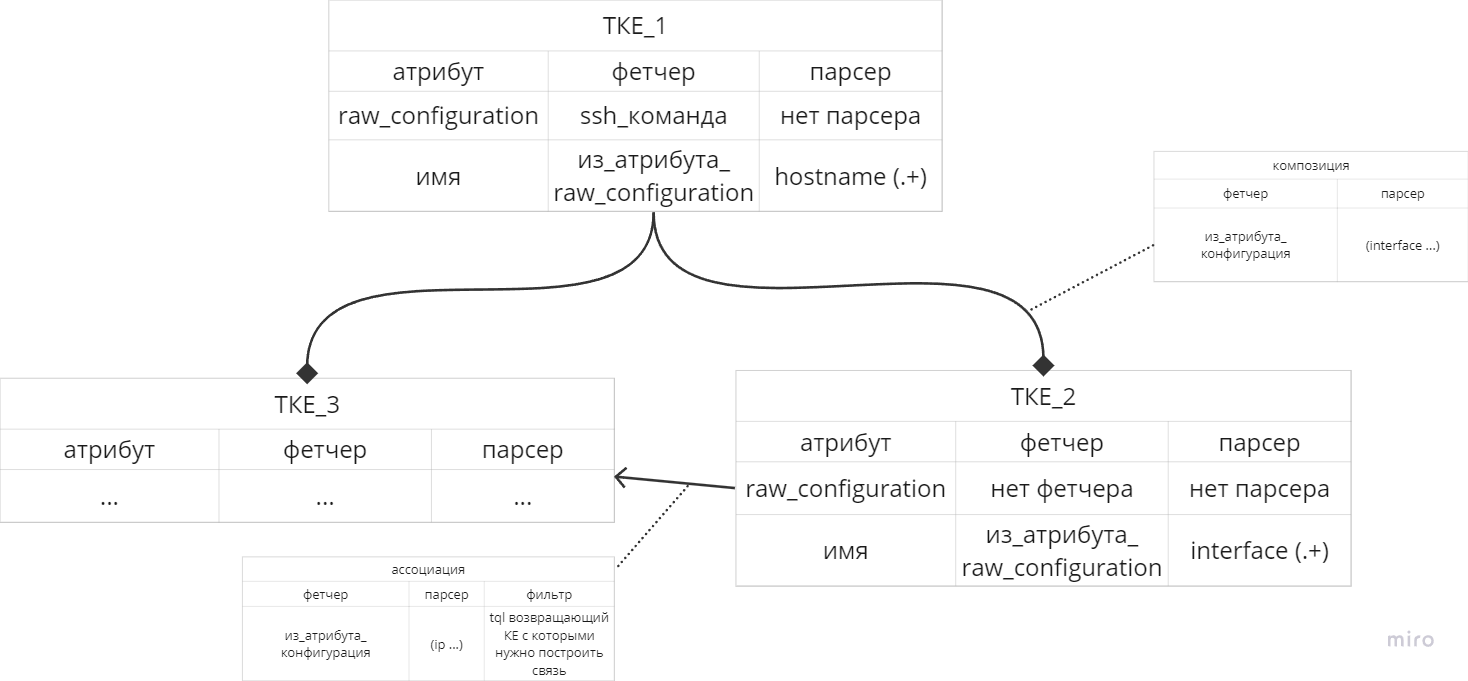


Рисунок 99. Необходимая настройка на ТКЕ для выполнения обогащения

После настройки КЕ и запуска пользователем (механизмом планировщика) механизма обогащения, происходит следующее (см. Рисунок 100. Заполнение атрибутов у КЕ 2 ):

1. На запущенной КЕ (КЕ\_1) заполняются атрибуты:
   1. для каждого настроенного атрибута запускается фетчер;
   2. результат, возвращенный фетчером, парсится (при наличии настроенного парсера);
   3. итоговый результат сохраняется в атрибуте.

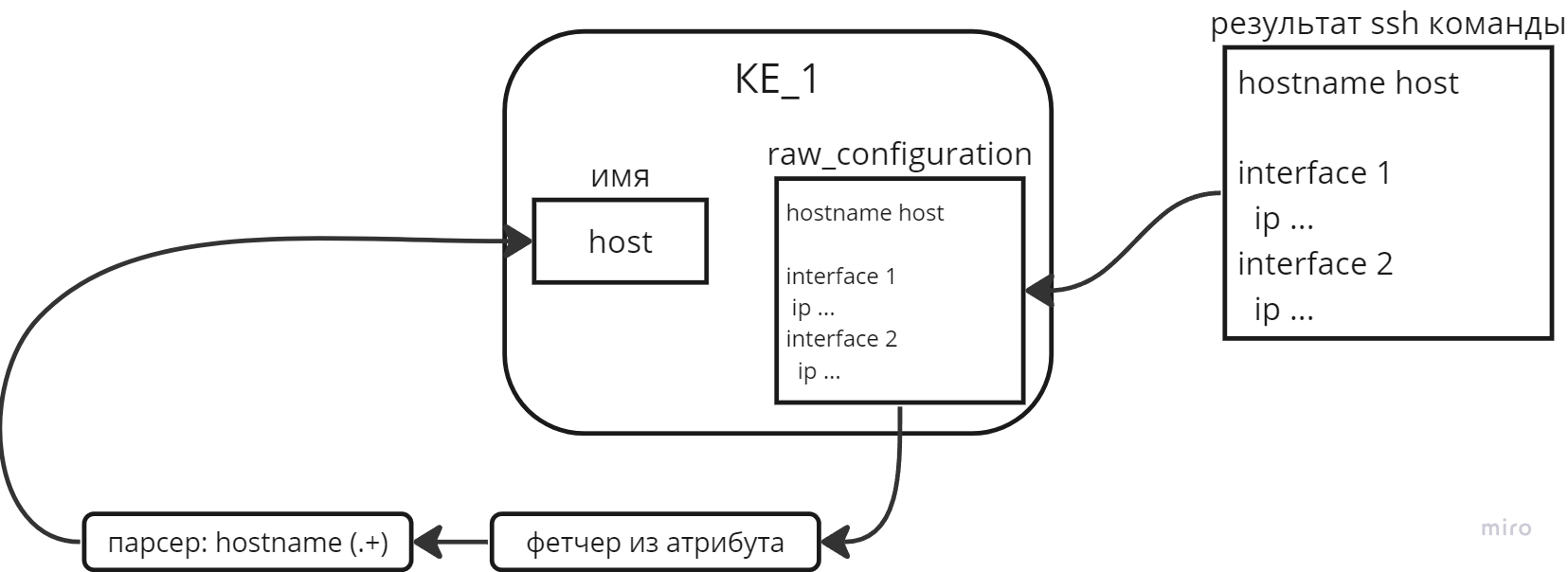


Рисунок 100. Заполнение атрибутов у КЕ 2

1. Проверяется отсутствие дубликата КЕ\_1 (см. Рисунок 101. Получение конфигурации для дочерних элементов КЕ ):
   1. в базе данных ищутся КЕ, которые совпадают с полученным КЕ\_1, в соответствии с имеющимися правилами идентификации;
   2. если похожих КЕ не найдено, происходит переход к следующему этапу;
   3. для найденных похожих КЕ запускается механизм разрешения конфликтов, который вернет одну актуальную КЕ в соответствии с работой механизма разрешения конфликтов.
2. КЕ\_1, заполненная актуальными значениями, становится видна всем остальным пользователям;
3. Для каждой, настроенной на КЕ\_1, связи производится поиск дочерних КЕ, связанных по связи типа композиция:
   1. фетчером, указанным в настройке связи, для получения конфигурации;
   2. результат, возвращенный фетчером, парсится (при наличии настроенного парсера), результатом работы парсера (фетчера) является набор (единственное значение) конфигураций для дочерних КЕ, свзанных с родителем связью типа композиция;

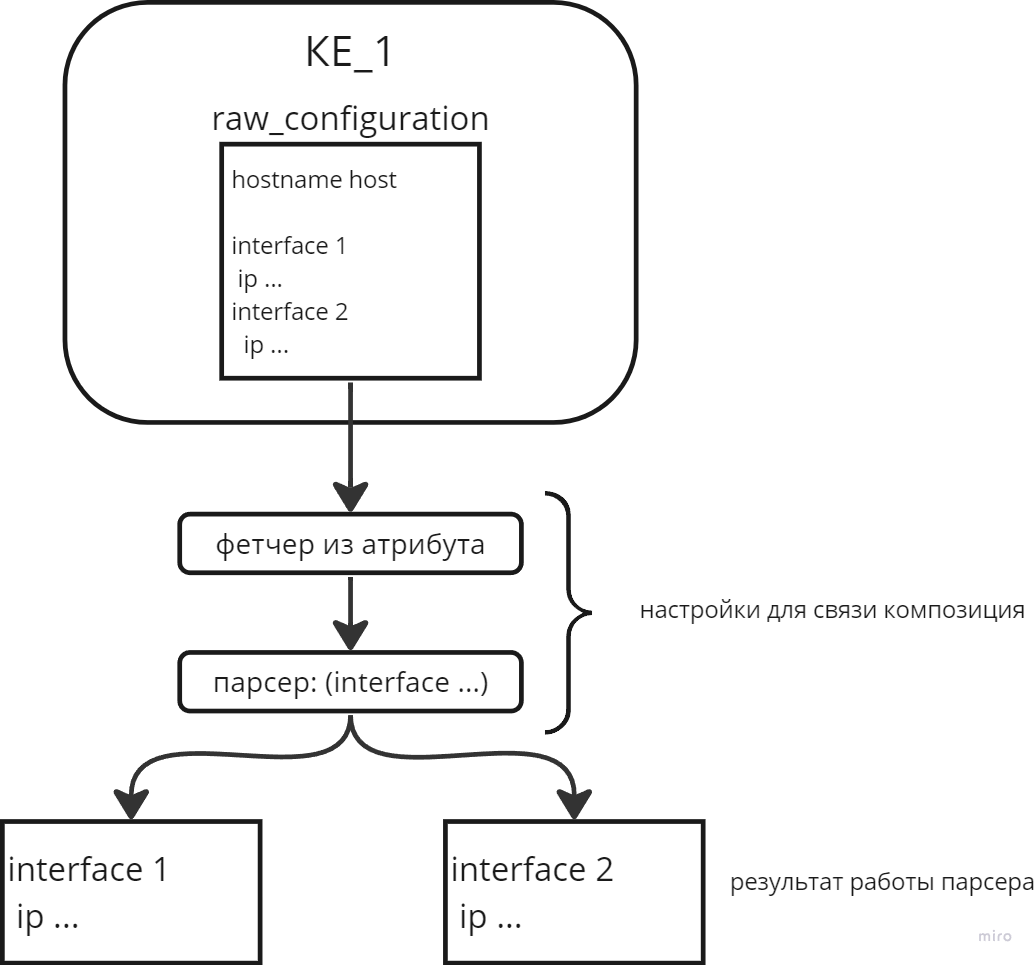


Рисунок 101. Получение конфигурации для дочерних элементов КЕ

* 1. создаются временные КЕ (КЕ\_2 и КЕ\_3);
  2. итоговый результат, для каждого полученного КЕ, сохраняется в атрибуте raw\_configuration (при наличии у КЕ такого атрибута) (см. Рисунок 102. Сохранение конфигурации );

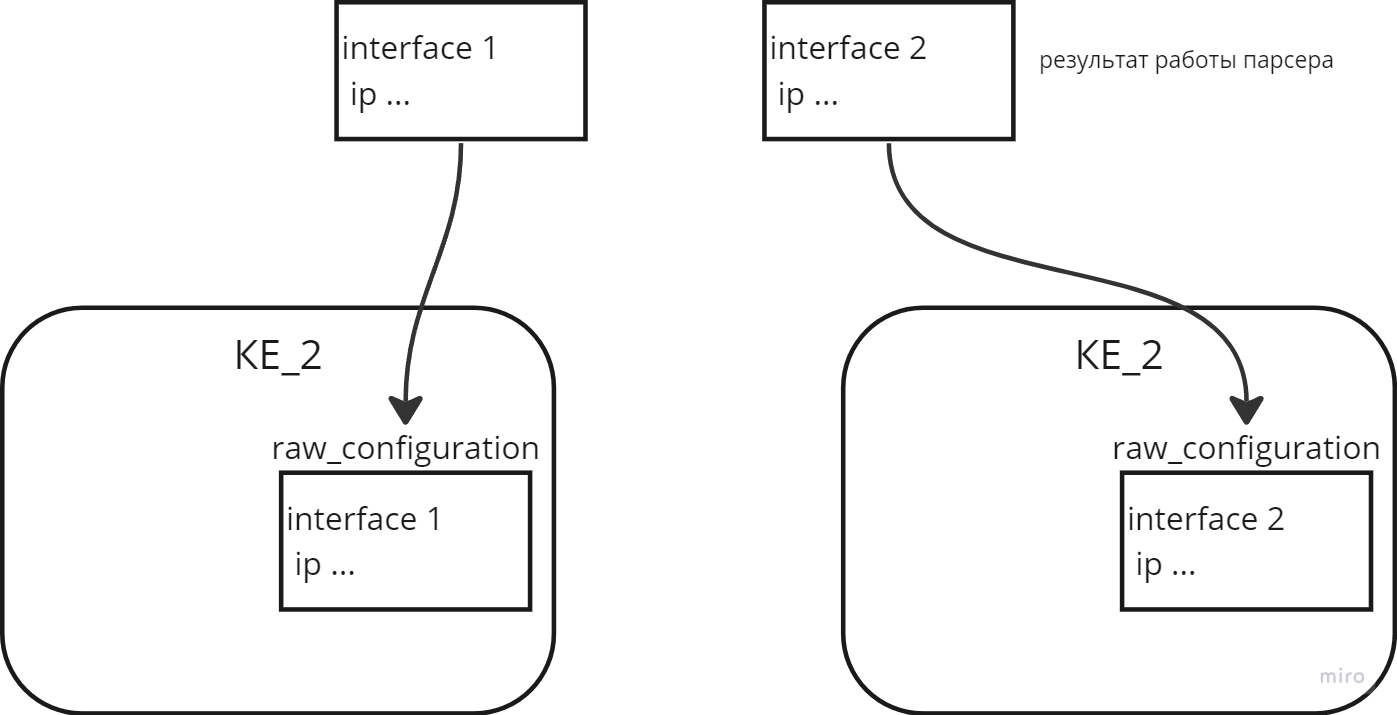


Рисунок 102. Сохранение конфигурации

* 1. для полученных КЕ выполняются п.п. 1 и 2 (см. Рисунок 103. Заполнение атрибутов КЕ );

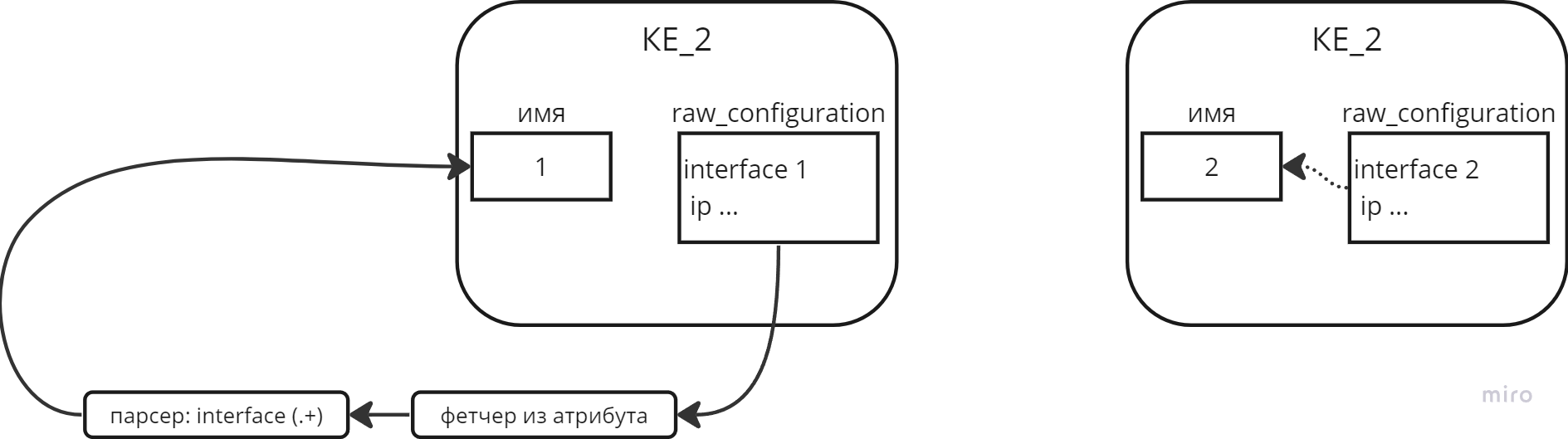


Рисунок 103. Заполнение атрибутов КЕ

* 1. итоговая КЕ добавляется в базу (обновляет имеющиеся значения) (см. Рисунок 104. Добавление (обновление) КЕ );

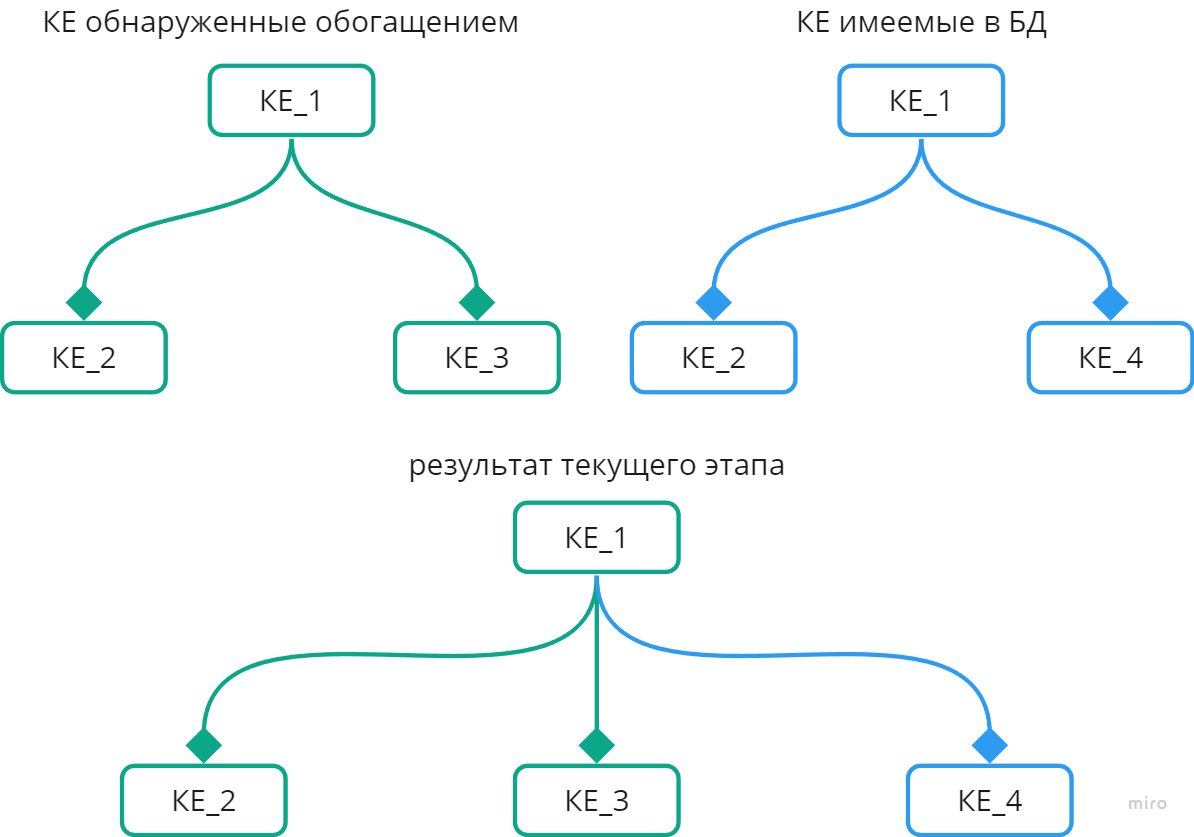


Рисунок 104. Добавление (обновление) КЕ

* 1. если у связи настроены атрибуты, то значения атрибутов парсятся в соответствии с настройкой из конфигурации, полученной в п. с (см. Рисунок 105. Заполнение атрибутов связи типа Композиция );

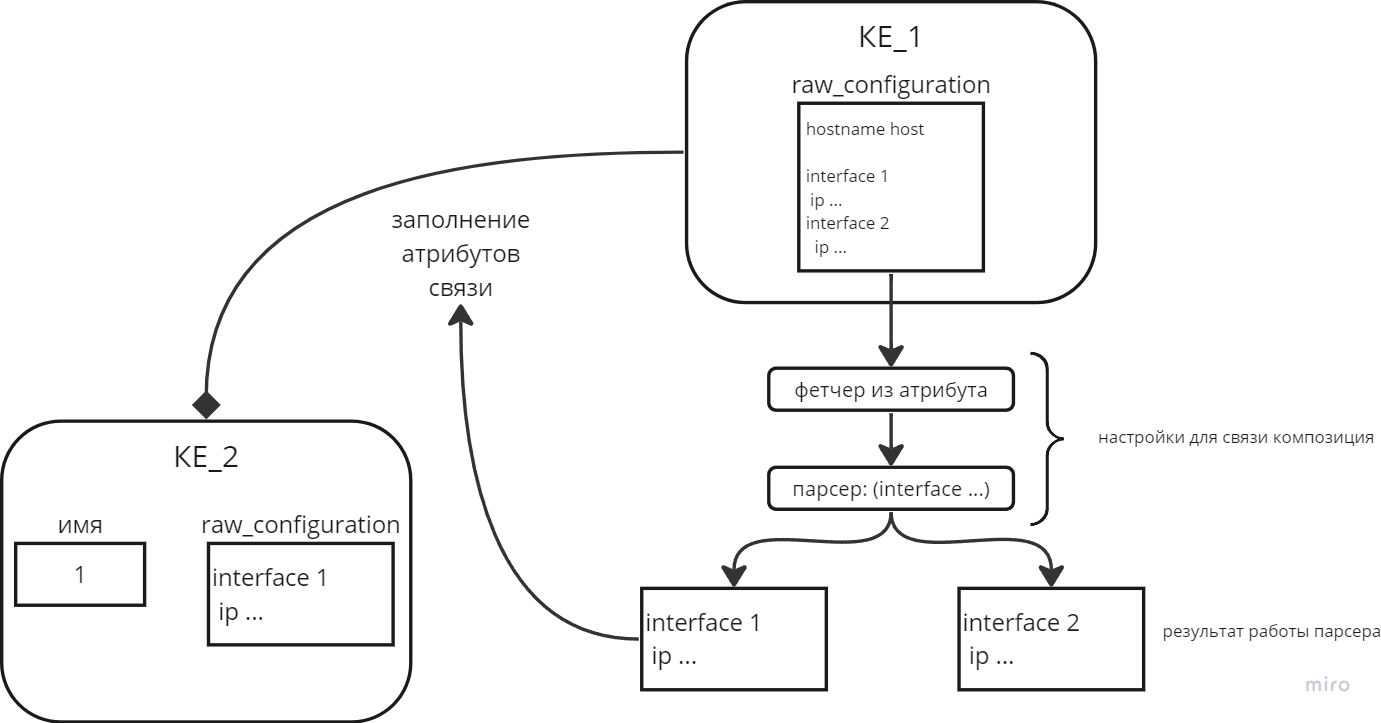


Рисунок 105. Заполнение атрибутов связи типа Композиция

* 1. из базы данных удаляются все КЕ, связанные связью композиция, но не найденные в результате выполнения обогащения (см. Рисунок 106. Результат обогащения, видимый пользователем );

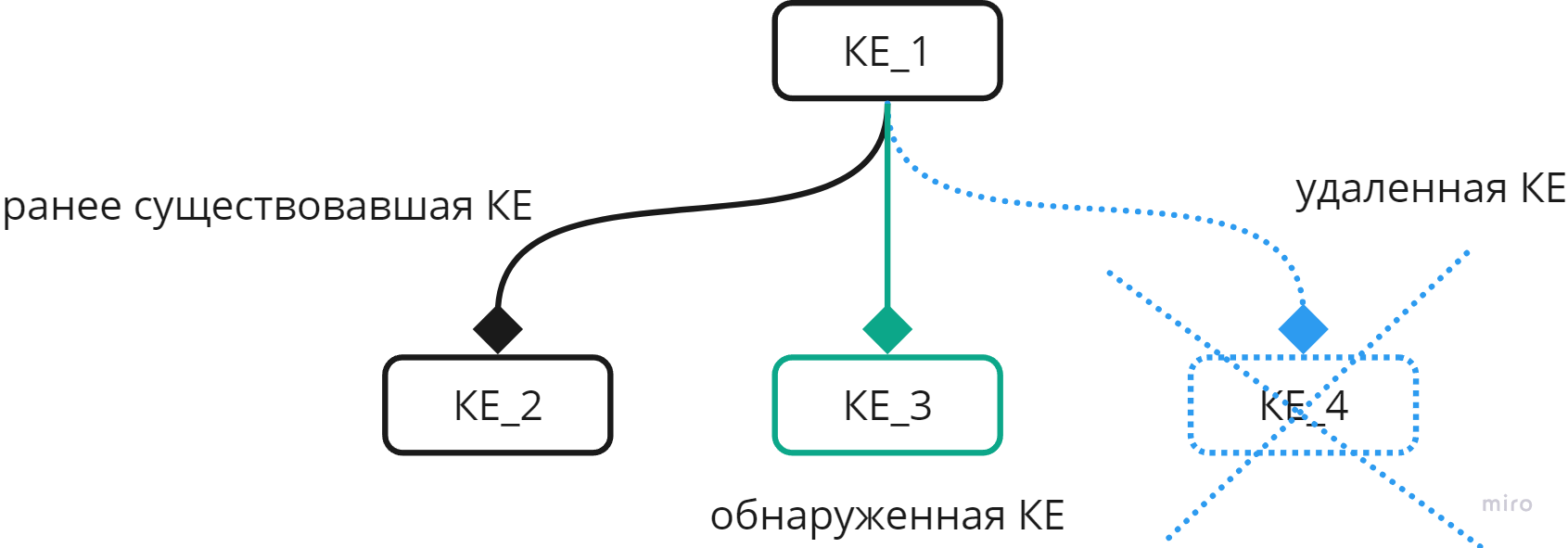


Рисунок 106. Результат обогащения, видимый пользователем

1. Пункт 4 повторяется для каждого найденного дочернего элемента до исчерпания настроенных связей типа композиция (см. Рисунок 107. Порядок обогащения ).

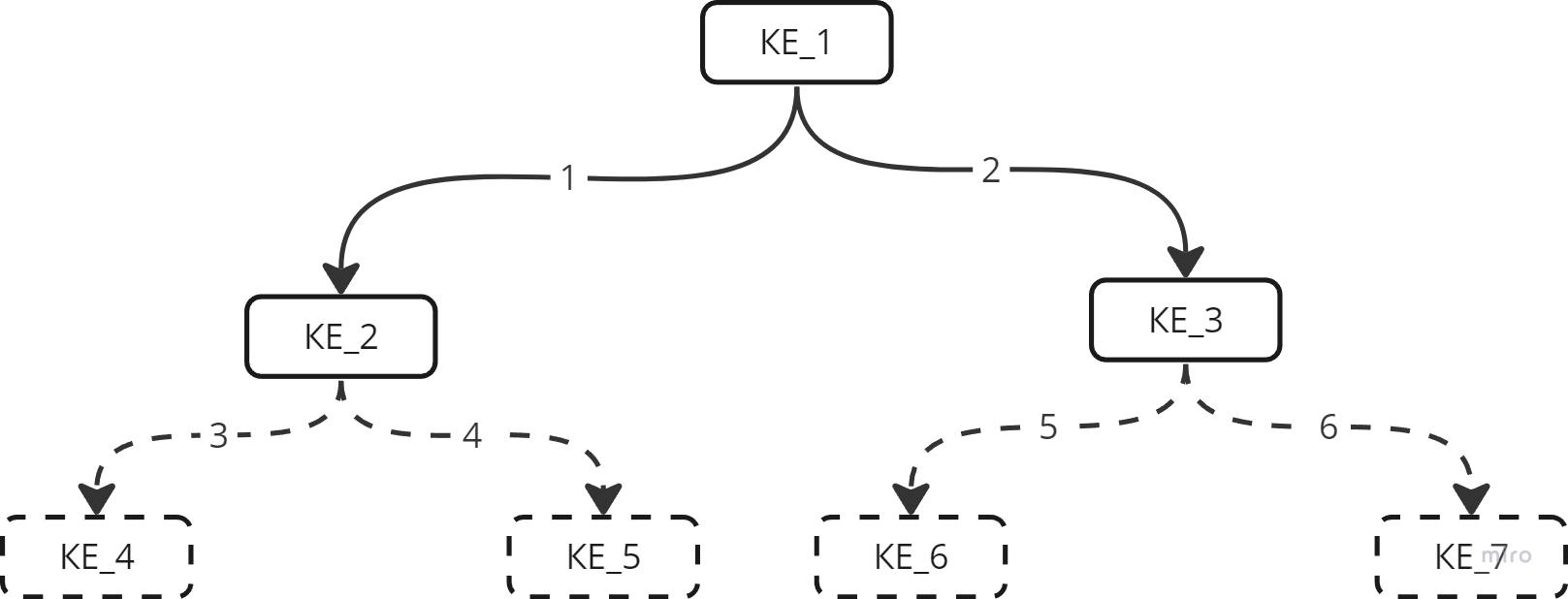


Рисунок 107. Порядок обогащения

1. Для всех КЕ, обнаруженных выше, в результате работы механизма обогащения строятся дополнительные связи (настроенные связи за исключением связи композиция) (см. Рисунок 108. Получение конфигурации для дополнительных связей ):
   1. фетчером, указанным в настройке связи, откуда получается конфигурация;
   2. при наличии настроенного парсера, обрабатывается парсером (итоговый результат, для каждого полученного КЕ, используется для извлечения атрибутов или переменных в условии поиска для tql запроса);

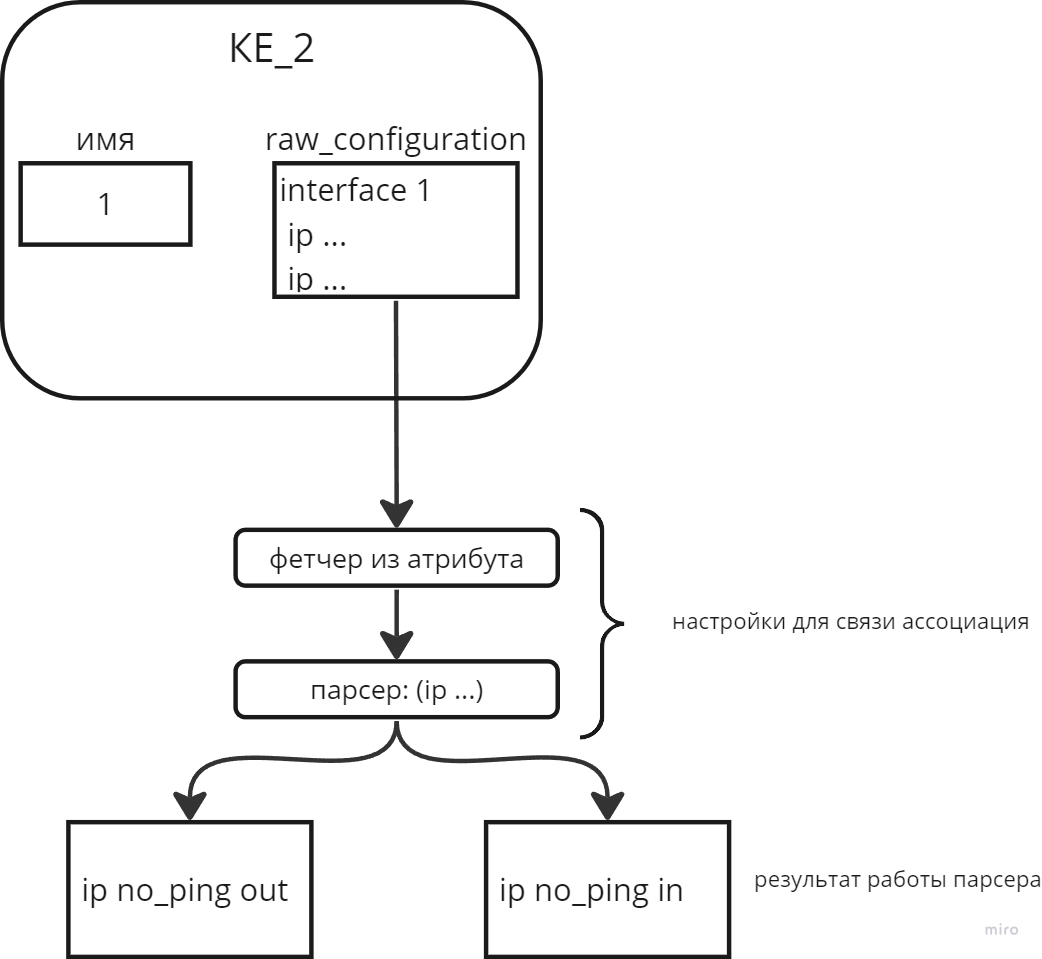


Рисунок 108. Получение конфигурации для дополнительных связей

* 1. заполняется фильтр (tql), назначенный для связи, и осуществляется поиск связанных КЕ (см. Рисунок 109. Подготовленный фильтр для поиска КЕ, с которыми необходимо построить связь);

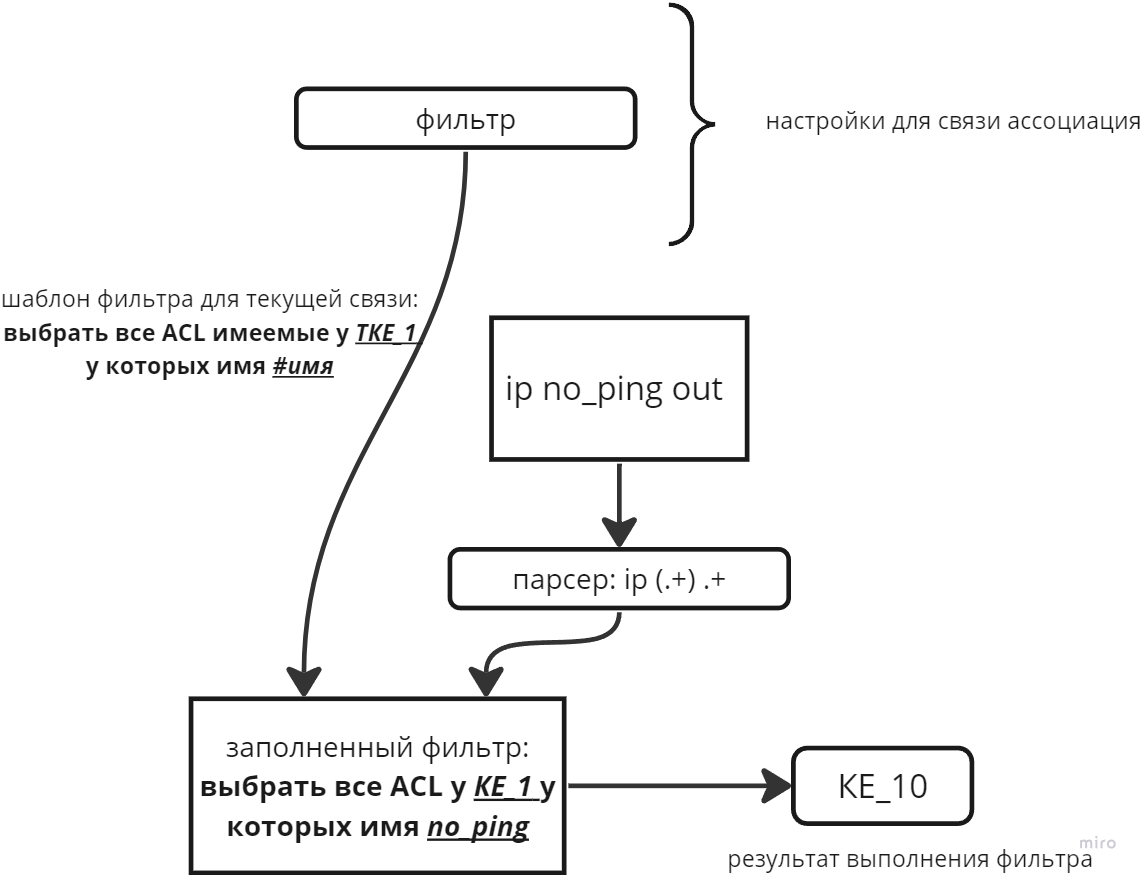


Рисунок 109. Подготовленный фильтр для поиска КЕ, с которыми необходимо построить связь

* 1. строится связь со всеми найденными КЕ (заполняются атрибуты связей);
  2. для обогащаемой связи из базы данных удаляются все имеющиеся связи, но не найденные в результате работы фильтра (см. Рисунок 110. Результат работы обогащения связей );

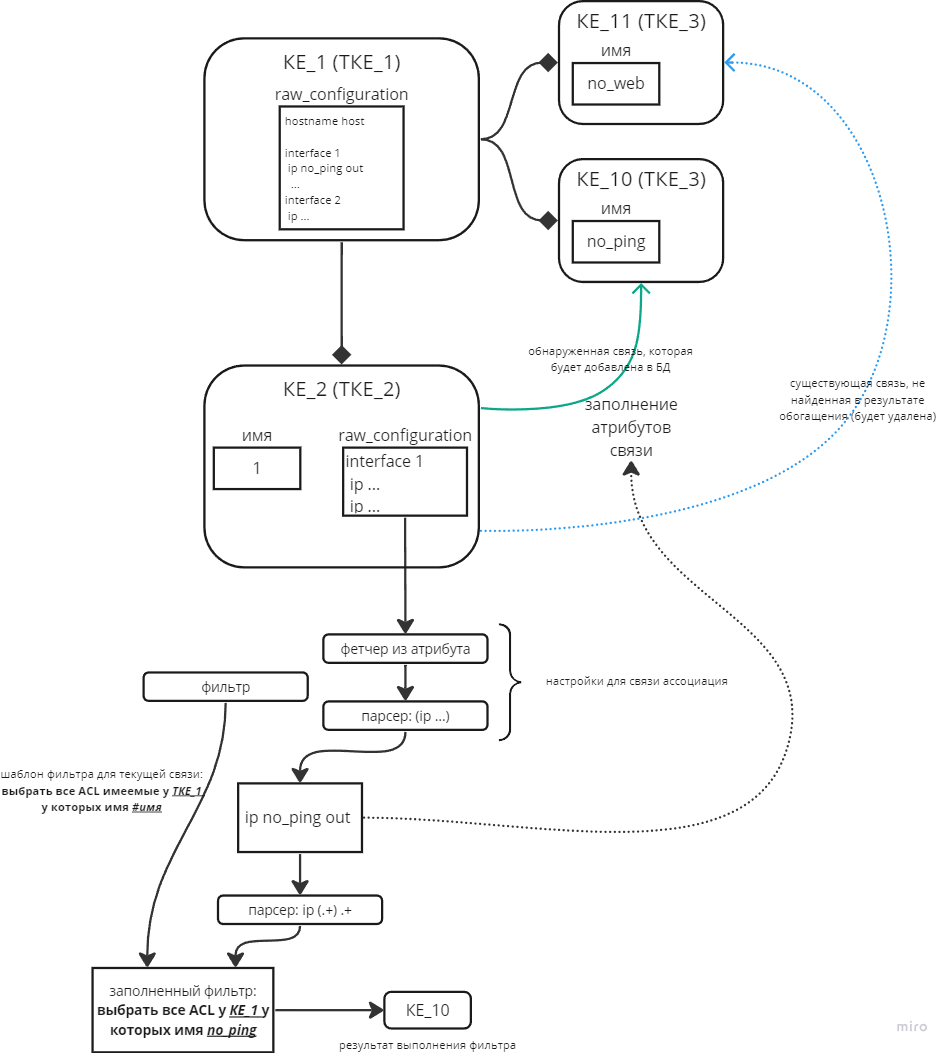


Рисунок 110. Результат работы обогащения связей

##### Настройка атрибутов

###### Фетчеры

Для атрибутов доступны следующие фетчеры:

* взять конфигурацию из атрибута (attribute\_config\_fetcher) - предназначен для получения данных из значения атрибута КЕ, при настройке необходимо выбрать атрибут, значение которого будет использовано для получения данных;
* сохранить конфигурацию в атрибут (filling\_config\_fetcher) - предназначен для заполнения атрибута КЕ определенным значением;
* выполнить PromQL запрос (prometheus\_config\_fetcher);
* выполнить SSH команду (ssh\_config\_fetcher) - предназначен для получения данных по ssh, при настройке необходимо задать команду, которая будет выполнена на КЕ, а результат ее выполнения будет сохранен в значение атрибута.

###### Парсеры

Для атрибутов доступны следующие парсеры:

* JsonPathParser – парсер для извлечения данных из JSON конфигурации, при помощи библиотеки Objectpath;
* RegexpChainParser - парсер по цепочке регулярных выражений;
* RegexpGroupParser - парсер по списку регулярных выражений;
* RegexpParser - парсер находит все или первое вхождения регулярного выражения;

JsonPathParser.

Парсер предназначен для получения конфигурации КЕ из JSON строки с использованием библиотеки Objectpath.

Параметры:

* выражение - строка, описывающая Objectpath выражение;
* группировка - поле, по которому происходит группировка, группа представляет из себя массив, задается в виде фильтра для json\_path, который можно применить после выполнения выражения.

RegexpChainParser.

Парсер по цепочке регулярных выражений. По очереди выполняет регулярные выражения; результаты, помещенные в группы, по одному передаются на вход следующего регулярного выражения. Возвращает список строк или логический тип, если используются постпроцессоры.

Параметры:

* список регулярных выражений – список регулярных выражений, поддерживаемых python;
* постпроцессор – постпроцессоры «Истина» (TruePostProcessor) или «Ложь» (FalsePostProcessor), применяются к результату, полученному после цепочки регулярных выражений, аргументом является ожидаемое значение в виде строки.

RegexpGroupParser.

Парсер по списку регулярных выражений с именованными группами. В регулярных выражениях можно задать имена групп, результаты разбора вернутся в виде списка словарей, ключами которых будут заданные имена групп. Если имена групп не заданы, вернется список строк.

Параметры:

* список регулярных выражений – список регулярных выражений с именованными группами, поддерживаемых python, которые по очереди выполняют поиск значений в многострочной текстовой конфигурации.

RegexpParser.

Парсер находит все или первое вхождения регулярного выражения. Возвращает список строк.

Параметры:

* регулярное выражение – регулярное выражение, поддерживаемое python, которое выполняет поиск значения в многострочной текстовой конфигурации.
* метод поиска – может принимать значения «все совпадения» или «первое совпадение»

##### Настройка дополнительных связей

###### Фетчеры

Для связей доступны фетчеры, аналогичные атрибутам.

###### Парсеры

Для связей доступны парсеры, аналогичные атрибутам.

##### Запуск обнаружения

###### Через планировщик

Для запуска обогащения по расписанию необходимо создать и настроить «План обогащения СРМ» (см. Рисунок 111. Настройка планировщика запуска обнаружения ).

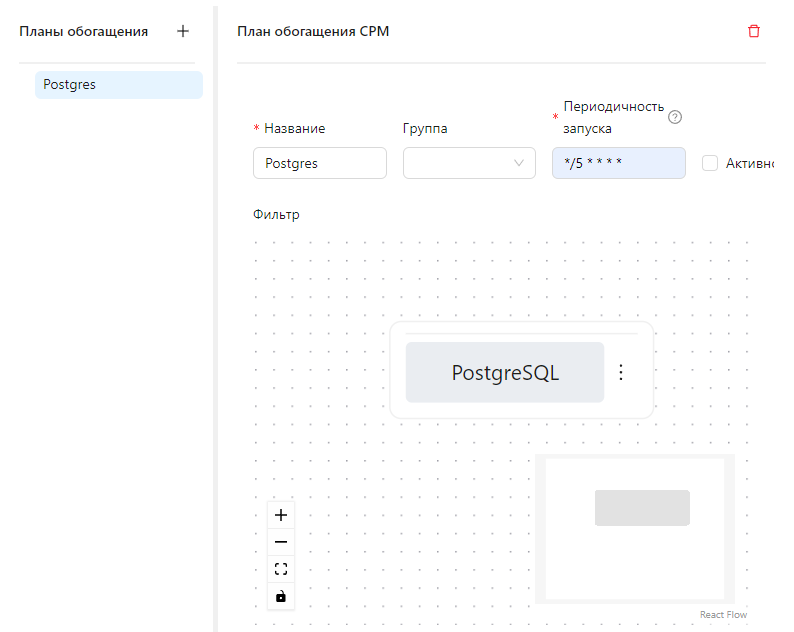


Рисунок 111. Настройка планировщика запуска обнаружения

При настройке необходимо задать название плана, периодичность запуска в формате CRON выражения, выбрать при необходимости группу, настроить фильтр ТКЕ и отметить флаг «Активно».

CRON выражение – это строка, которая состоит из пяти ячеек, разделённых пробелами. Ячейки задают время выполнения (минута, час, день, месяц, день недели), в них может находиться число, список чисел через запятую, диапазон чисел через дефис, символы «\*» или «/».

Планы обогащения для упорядочивания можно объединять в группы.

###### Мануально из СРМ

Для запуска обнаружения необходимо на графе СРМ выбрать КЕ, внизу окна «Параметры КЕ» нажать кнопку «Выполнить обогащение» (см. Рисунок 112. Запуск обнаружения на выбранном КЕ ).

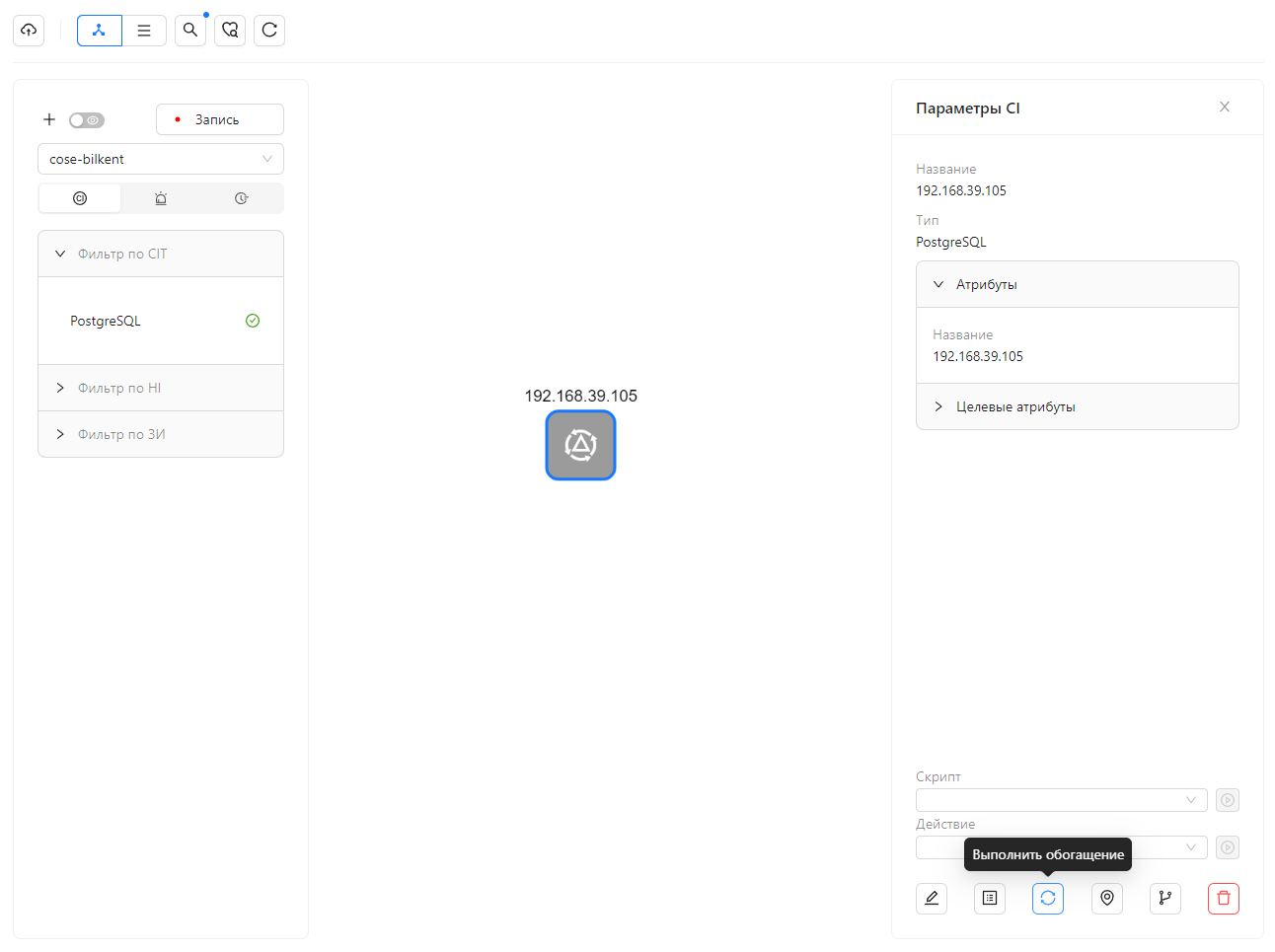


Рисунок 112. Запуск обнаружения на выбранном КЕ

### Управление доступом

ПАМИР позволяет управлять доступом пользователей в части:

* функциональных возможностей системы;
* доступа к СРМ и информации мониторинга.

Управление доступом к функциональным возможностям (см. Рисунок 113. Меню управление ) выполняется в разделе меню Доступ > Функции.

Управление доступом к КЕ выполняется в разделе меню Доступ > СРМ.

Управление доступом к дашбордам выполняется в разделе меню Доступ > Дашборды.

Управление доступом к индикаторам здоровья выполняется в разделе меню Доступ > Индикаторы здоровья.

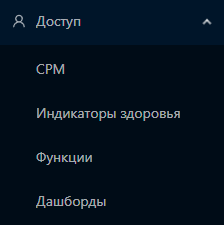


Рисунок 113. Меню управление

Рисунок. Меню управление

###### Алгоритм применения правил

Любое правило либо разрешает, либо запрещает доступ. Соответствующее действие выбирается в списке Доступ (см. Рисунок 114. Действия правил доступа ). Правило с доступом Разрешить является разрешающим. Правило с доступом Запретить является запрещающим.

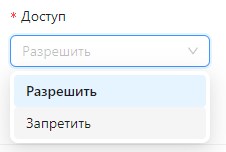


Рисунок 114. Действия правил доступа

Запрещающее правило имеет больший приоритет, чем разрешающее. При отсутствии разрешающих правил доступ запрещается, то есть все, что не разрешено, то запрещено.

Правило направлено на:

* объект: дашборд, индикатор здоровья, КЕ, функция системы;
* действие над объектом:
  + дашборд: просматривать, изменять;
  + индикатор здоровья: просматривать, изменять;
  + КЕ: редактирование, чтение;
  + функции : использовать.

Алгоритм:

1. доступ ко всем объектам по умолчанию запрещен;
2. выполняется поиск всех разрешающих правил, в которые попадает пользователь, прошедший аутентификацию;
3. найденные правила разрешают пользователю действия над объектами, указанными в правилах;
4. выполняется поиск всех запрещающих правил, в которые попадает пользователь, прошедший аутентификацию;
5. найденные правила запрещают пользователю действия над объектами, указанными в правилах.

Выбранные действия в разрешающих правилах интерпретируются как разрешить пользователю выполнять указанные действия над объектами (см. Рисунок 115. Действия в разрешающих правилах ).

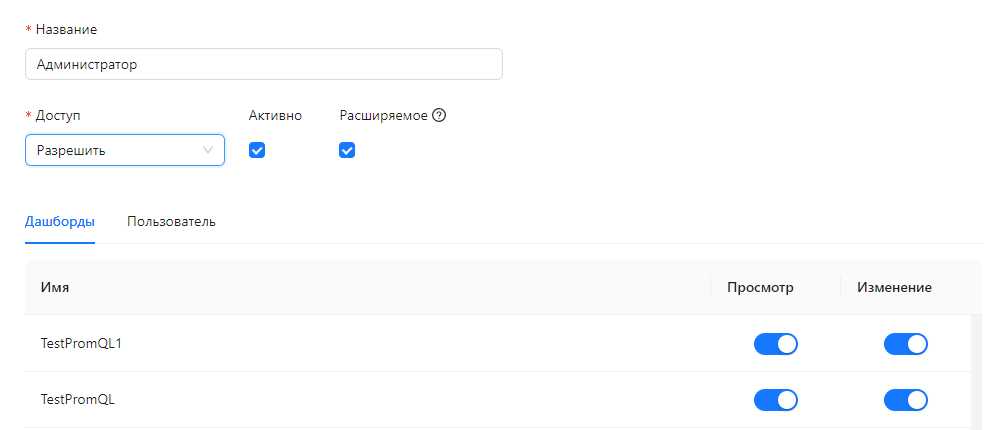


Рисунок 115. Действия в разрешающих правилах

Выбранные действия в запрещающих правилах интерпретируются как – запретить пользователю выполнять указанные действия над объектами (см. Рисунок 116. Действия в запрещающих правилах ).

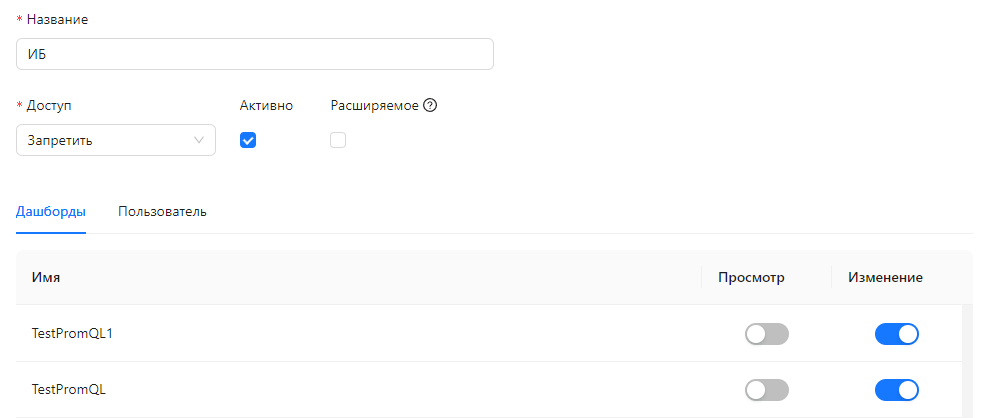


Рисунок 116. Действия в запрещающих правилах

Пример.

Рисунки выше (см. Рисунок 115. Действия в разрешающих правилах и Рисунок 116. Действия в запрещающих правилах ) показывают, как можно разрешить просмотр и редактирование дашбордов TestPromQL1 и TestPromQL для всех типов администраторов. Но запретить изменять их для администраторов ИБ.

Если правило является расширяемым, то в него будут добавляться вновь создаваемые объекты.

###### Фильтрация пользователей

Фильтрация пользователей, попадающих под правила, выполняется на вкладке Пользователь.

Фильтр представляет собой логическое выражение, оперирующее атрибутами пользователя. Если логическое выражение разрешается в истину, то правило применяется для пользователя, если в ложь, то не применяется.

Выражение оперирует атрибутами пользователя и группами атрибутов. В группе атрибутов можно указать оператор ( И / ИЛИ ), который будет применен между элементами группы. Группы могут быть вложенными.

### Управление пользователями

#### Алгоритм аутентификации и авторизации

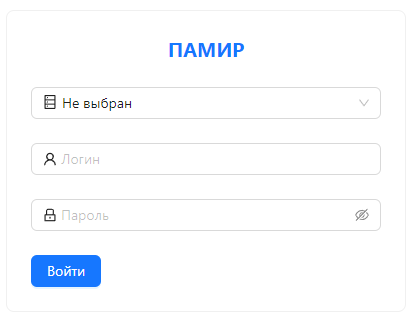


Рисунок 117. Окно аутентификации

**Окно аутентификации представлено на примере (см.** Рисунок 117. Окно аутентификации **)**

###### **Аутентификация с указанным доменом**

Если пользователь при аутентификации указывает домен, то аутентификация осуществляется сервером LDAP, обслуживающим соответствующий домен:

* в случае успешной аутентификации, происходит поиск пользователя в локальном хранилище на основе его login и LDAP (домен);
* если пользователь найден, то для него выполняется синхронизация атрибутов и авторизация;
* если пользователь не найден, то он создается и сохраняется в локальном хранилище, ему присваивается LDAP соответствующего сервера и в качестве пароля устанавливается пустое значение (null). Пароль пользователя от LDAP сервера **не сохраняется;**
* в случае ошибки - аутентификация прекращается, пользователю возвращается сообщение об ошибке.

###### **Аутентификация без указания домена**

Если пользователь при входе в систему не указывает домен, то приоритетно осуществляется аутентификация с использованием локального хранилища пользователей:

* если найден локальный пользователь с указанным login, то для него выполняется процедура аутентификации;
* в случае успеха для пользователя выполняется авторизация;
* в случае ошибки - аутентификация прекращается, пользователю возвращается сообщение об ошибке;
* если локальный пользователь с указанным login **не** найден, и в системе настроена активная интеграция с **одним** сервером LDAP, то выполняется попытка аутентификации через LDAP в полном соответствии с пунктом "Аутентификация с указанным доменом" с единственным настроенным сервером LDAP;
* если локальный пользователь с указанным login и LDAP **не** найден, и в системе настроена активная интеграция с **более чем одним (или нулем)** сервером LDAP, то пользователю возвращается сообщение об ошибке.

Вне зависимости от домена пользователь имеет ограниченное количество попыток указания верных данных для аутентификации, в случае неудачи - пользователь будет заблокирован.

#### Управление локальными пользователями

###### **Реестр пользователей**

Данные локальных пользователей хранятся в базе данных на сервере. Список существующих пользователей можно увидеть в соответствующей вкладке (см. Рисунок 118. Реестр пользователей).

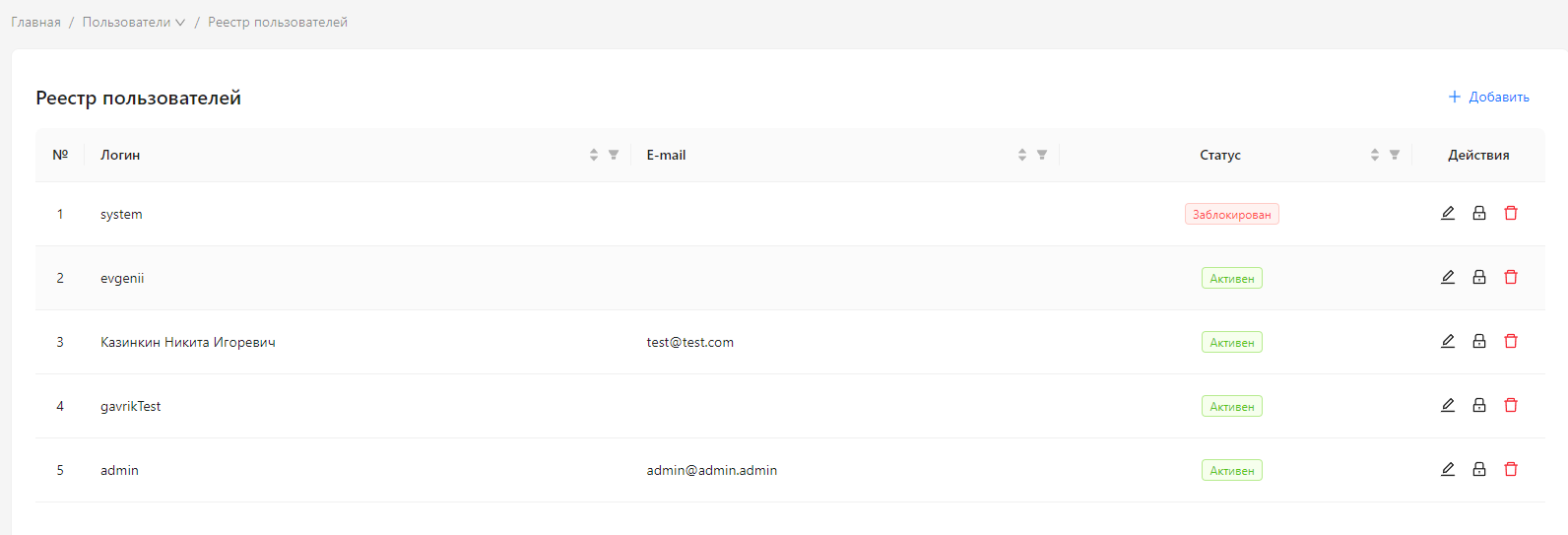


Рисунок 118. Реестр пользователей

Каждый пользователь обладает рядом обязательных атрибутов (логин, пароль, статус) и набором необязательных, таких как Роль, Регион, Аватар и прочие. На странице (см. Рисунок 118. Реестр пользователей) видна часть этих атрибутов в качестве полей таблицы пользователей. Здесь можно открыть меню редактирования атрибутов пользователей, изменить пароль, удалить и добавить пользователя.

###### Добавление пользователя

Добавить нового пользователя можно как через успешную аутентификацию (см. п. [4.2.6.1](#_Алгоритм_а)), так и через кнопку «Добавить» на странице Реестра пользователей (см. Рисунок 118. Реестр пользователей). Для успешного создания нового пользователя (см. Рисунок 119. Добавление пользователя ) необходимо заполнить все обязательные поля (логин, статус, пароль и подтверждение), оставшиеся поля заполняются по желанию. Таким образом можно добавить только пользователя **не** из LDAP сервера (для добавления пользователя с аутентификацией через домен LDAP смотри п. [4.2.6.1 ч.1](#_Аутентифик)). В случае корректного заполнения всех полей и наличия у текущего пользователя необходимых прав доступа при нажатии кнопки «Сохранить» произойдет добавление нового пользователя, в ином случае будет выведено сообщение об ошибке с указанием подробностей.

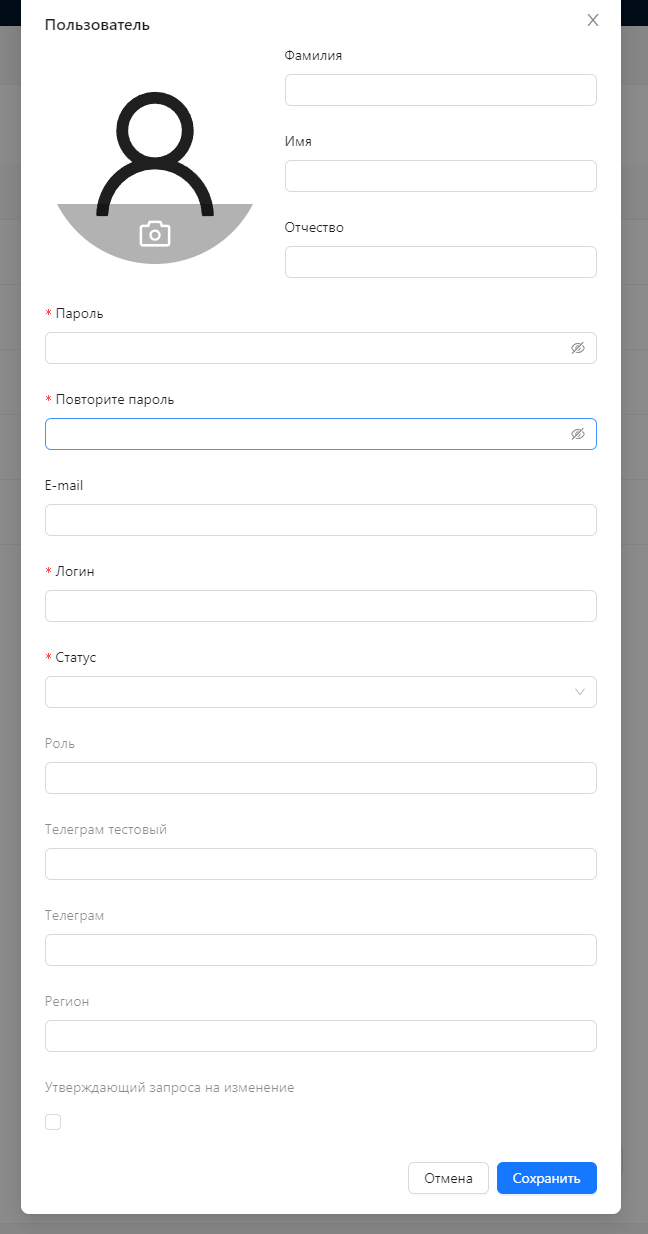


Рисунок 119. Добавление пользователя

###### **Изменение пользователя**

Изменение данных пользователя происходит с помощью окна, открывающегося при нажатии кнопки «Редактировать» на странице Реестр пользователей (см. Рисунок 120. Изменение пользователя ).

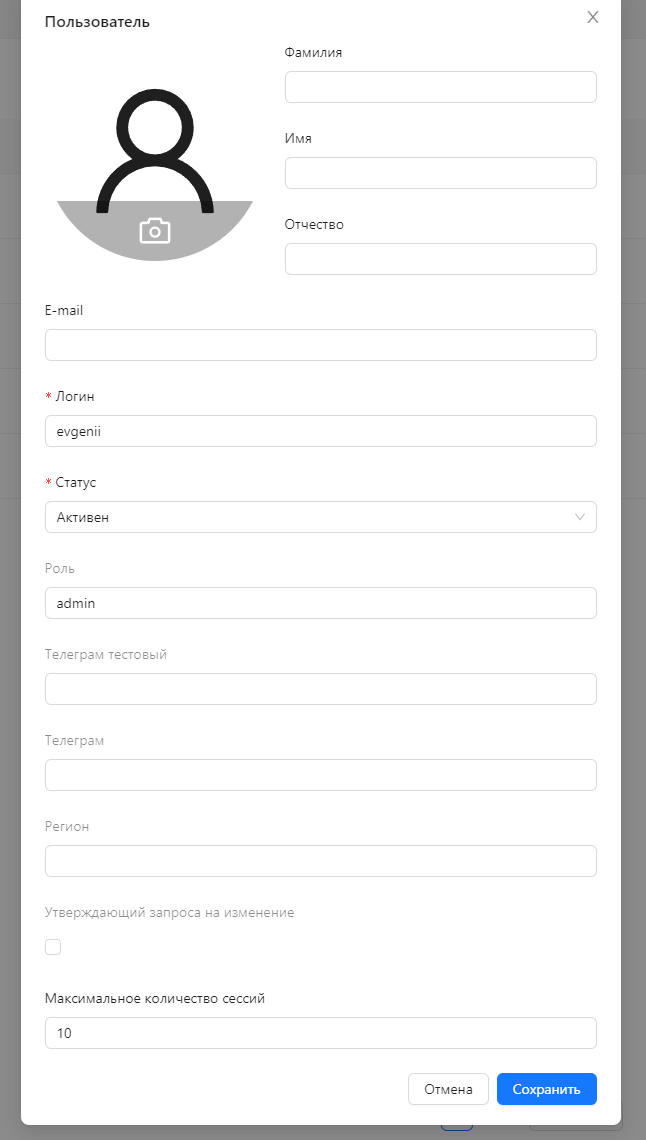


Рисунок 120. Изменение пользователя

Для изменения доступны все существующие атрибуты пользователя кроме его пароля и домена. В случае корректного заполнения всех полей и наличия у текущего пользователя необходимых прав доступа при нажатии кнопки «Сохранить» произойдет изменение атрибутов выбранного пользователя, в ином случае будет выведено сообщение об ошибке с указанием подробностей.

###### **Создание дополнительных полей пользователя**

Базовыми атрибутами пользователя считаются имя, фамилия, отчество, логин, статус, почтовый адрес, аватар и максимальное количество одновременных сессий. Условно скрытыми являются пароль и домен LDAP. Помимо базовых атрибутов пользователям могут быть присвоены дополнительные атрибуты, созданные вручную. Для этого необходимо перейти на пункт меню «Настройки», далее «Пользователь» и «Атрибуты». На открывшейся странице будет выведен список существующих дополнительных атрибутов пользователя, который можно расширять своими добавляемыми атрибутами (см. Рисунок 121. Дополнительные поля пользователя ).

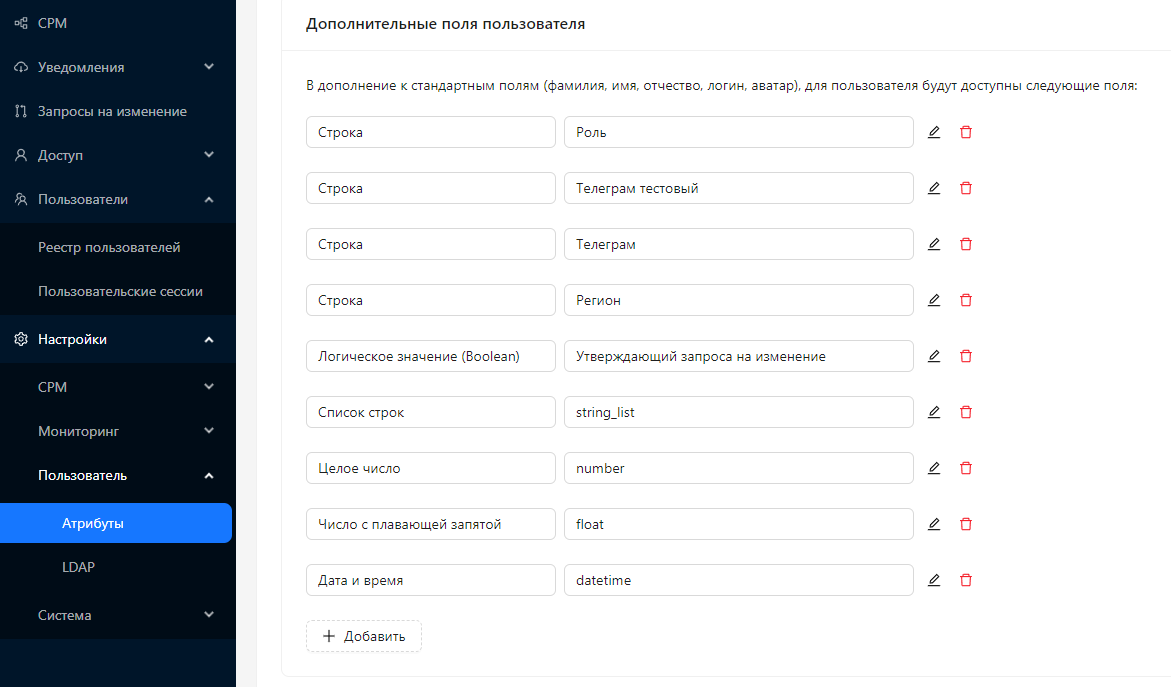


Рисунок 121. Дополнительные поля пользователя

На данной странице можно как отредактировать или удалить существующий дополнительный атрибут, так и добавить свой с указанием его типа (строка, число, дата и время и прочее). После чего данный атрибут будет доступен для указания/редактирования у локальных пользователей. Таким образом, например, пользователям были добавлены атрибуты Телеграм, Регион и прочие (см. Рисунок 119. Добавление пользователя и Рисунок 120. Изменение пользователя ).

#### Настройка LDAP

###### Основные понятия LDAP

LDAP (Lightweight Directory Access Protocol) – «легковесный протокол доступа к каталогам» — открытый протокол прикладного уровня с множеством реализаций для доступа к службе каталогов, использующий TCP/IP и позволяющий производить операции аутентификации (bind), поиска (search) и сравнения (compare), а также операции добавления, изменения или удаления записей об организации, ее активах и пользователях.

Основные понятия LDAP представлены ниже.

DN (Distinguished Name) – Уникальное имя; отличительное имя. Всякая запись в каталоге LDAP состоит из одного или нескольких атрибутов и обладает уникальным именем. Примеры уникальных имен:

* «CN=Администратор,CN=Users»;
* «CN=Иван Петров,OU=Сотрудники,DC=example,DC=com».

Уникальное имя состоит из одного или нескольких относительных уникальных имён (RDN).

RDN (Relative Distinguished Name) – Относительное уникальное имя. Уникальное имя (DN) состоит из одного или нескольких относительных уникальных имён (RDN), разделённых запятыми. Относительное уникальное имя имеет вид ИмяАтрибута=значение. На одном уровне каталога не может существовать двух записей с одинаковыми относительными уникальными именами. Примеры RDN:

* «CN=Иван Петров»;
* «OU=Сотрудники»;
* «DC=example»;
* «DC=com».

Атрибуты записи. Каждая запись в LDAP-системе обладает уникальным именем (DN) и хранится как совокупность элементов, называемых атрибутами. Атрибуты являются парами «ключ-значение», например: «mail: [admin@example.com](mailto:admin@example.com)», или «CN: Иван Петров». Каждый атрибут определён в наборе схемы данных.

DIT (Data Information Tree) – Информационное дерево Каталога (Информационное дерево справочника). Все записи в системе LDAP иерархически упорядочены в виде веток на деревьях, называемых DIT. DIT представляет собой организационную структуру (Каталог), где каждая запись (кроме корневой записи верхнего уровня) имеет ровно одну родительскую запись и под ней может находиться любое количество дочерних записей. Запись по ее DN (distinguished name) используется для идентификации в DIT в целом и функционирует как полный путь до "корня" DIT. «Корнем» является само доменное имя (например: foiv.local; citadel.com; example.com и т.д.).

Base DN – База поиска, ветка DIT, от которой начинается поиск данных для LDAP-идентификации. Например: «DC=foiv,DC=local».

Group DN – Отличительное имя для групп (групп безопасности), которое добавляется (может добавляться) к «Base DN», для уточнения (сужения) поиска для LDAP-идентификации. «Group DN» может быть эффективен для иерархических структур Каталога LDAP. Пример: «OU=Groups». При поиске будет учитываться содержимое «Base DN» и «Group DN».

User DN – Отличительное имя для пользователей в пределах групп (групп безопасности), которое добавляется (может добавляться) к «Group DN» и «Base DN» для уточнения (сужения) поиска для LDAP-идентификации. «Group DN» и «User DN» могут быть эффективны, например, если пользователи находятся в локациях, отличных от групп, членами которых они являются. Пример: «OU=Users». При поиске будет учитываться содержимое «Base DN», «Group DN» и «User DN».

Schema – схема данных (набор схемы данных): пакет определений атрибутов и объектных классов, которые используют LDAP-приложения (согласно DIT).

Также частями спецификации Каталога LDAP, которые определяют его узлы, являются: CN = Common Name (Общее Имя); OU = Organizational Unit (Подразделение). Определяет произвольное подразделение организации и может использоваться на нескольких уровнях иерархии. Его значение, как правило, соотносится с контекстом, в котором оно используется; DC = Domain Component (Компонент Домена).

###### Настройка LDAP

Для работы с настройками LDAP необходимо выбрать в основном меню: Главная > Настройки > Пользователь > LDAP (см. Рисунок 122. Меню работы с LDAP ).

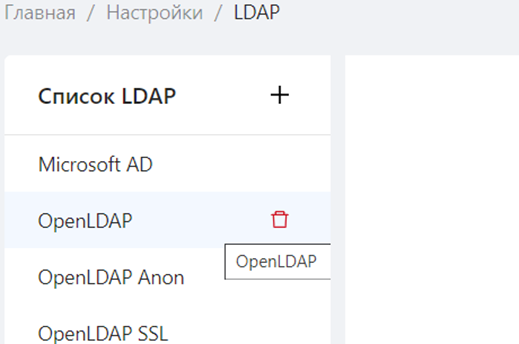


Рисунок 122. Меню работы с LDAP

При выборе конкретного LDAP при создании и/или корректировке требуется корректно заполнить его поля на вкладке «Соединение» (см. Рисунок 123. Создание или изменение конфигурации LDAP ). .

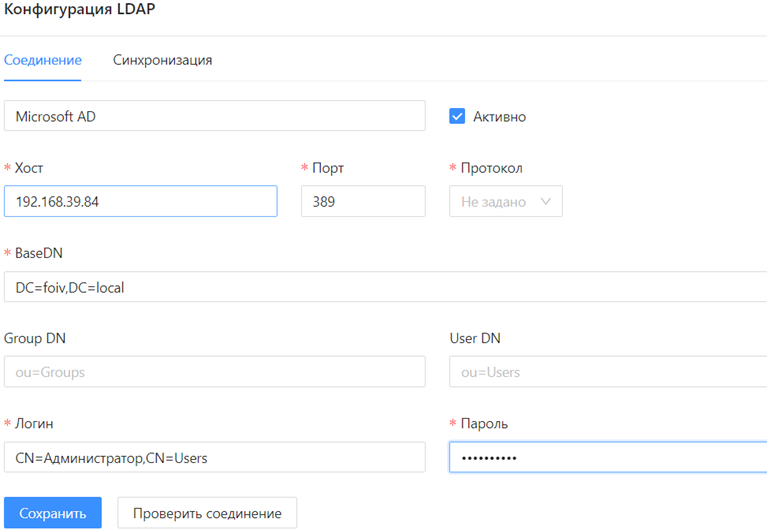


Рисунок 123. Создание или изменение конфигурации LDAP

 Разберем представленные поля и примеры их заполнения:

* название – название LDAP-соединения исходя из его  назначения;
* хост – IP-адрес или полное доменное имя соединения (FQDN);
* порт – номер используемого порта для LDAP;
* протокол – тип используемого протокола (Без шифрования, SSL, TLS);
* Base DN – База поиска.  Пример: DC=foiv,DC=local (см. также Основные понятия LDAP);
* Group DN – Отличительное имя для групп (групп безопасности). Пример: «OU=Groups» (см. также Основные понятия LDAP);
* User DN – Отличительное имя для пользователей в пределах групп (групп безопасности). Пример: «OU=Users» (см. также Основные понятия LDAP);
* логин – имя для идентификации в LDAP (например: CN=Администратор,CN=Users);
* пароль – пароль пользователя для идентификации в LDAP.

Опции «Сохранить» и «Проверить соединение» позволяют соответственно сохранить сделанные изменения либо проверить имеющееся LDAP-соединение. Пример успешности проверки LDAP-соединения представлен ниже (см. Рисунок 124. Проверка конфигурации LDAP ).

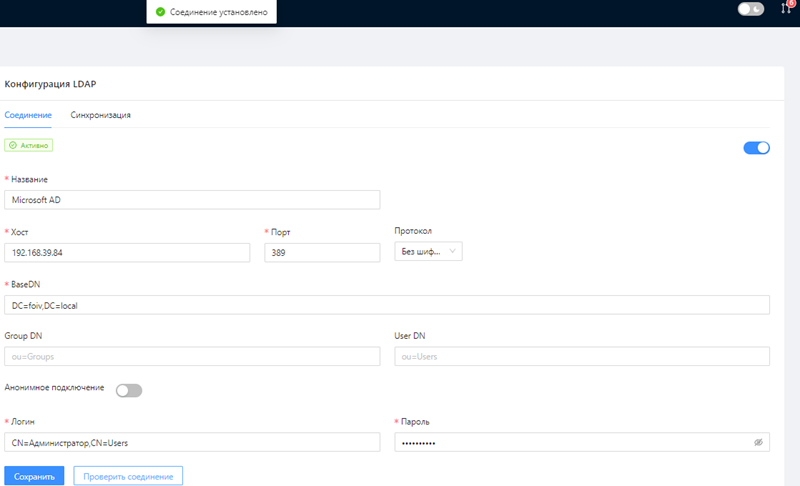


Рисунок 124. Проверка конфигурации LDAP

### Отправка уведомлений

#### Мои уведомления

Данная страница позволяет настраивать личные уведомления пользователя о событиях в работе приложения (см. Рисунок 125. Мои подписки ).

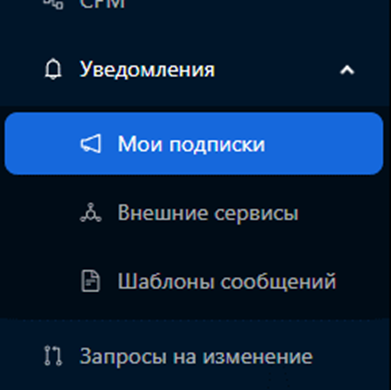


Рисунок 125. Мои подписки

На выбор доступные следующие виды событий (см. Рисунок 126. Страница редактирования уведомлений пользователя ):

* добавление/удаление группы КЕ;
* запуск приложения;
* изменение группы атрибутов;
* изменение группы индикаторов здоровья КЕ;
* включение/выключение получения метрик;
* включение/выключение индикаторов здоровья;
* добавление группы пользователей в систему;
* сообщение по умолчанию.

Предусмотрено 3 варианта отправки уведомлений:

* сообщение через Telegram;
* письмо на электронную почту;
* уведомление на странице приложения.

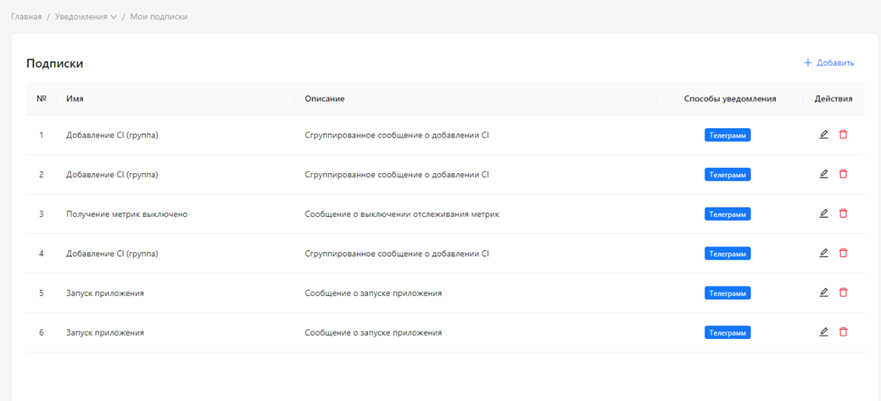


Рисунок 126. Страница редактирования уведомлений пользователя

Для добавления новой конфигурации уведомления необходимо нажать на кнопку “Добавить”, после чего откроется модальное окно для настройки новой конфигурации (см. Рисунок 127. Настройка конфигурации уведомления ).

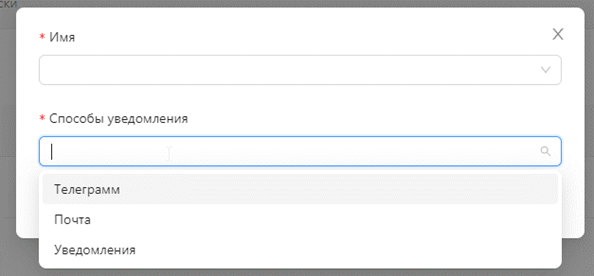


Рисунок 127. Настройка конфигурации уведомления

#### Внешние сервисы

Для настройки внешних сервисов необходимо в навигационном меню перейти в вкладку «**Уведомления**», далее «**Внешние сервисы**» ( см. Рисунок 128. Настройка внешних сервисов и далее пройти по указателю №1 на данном рисунке).

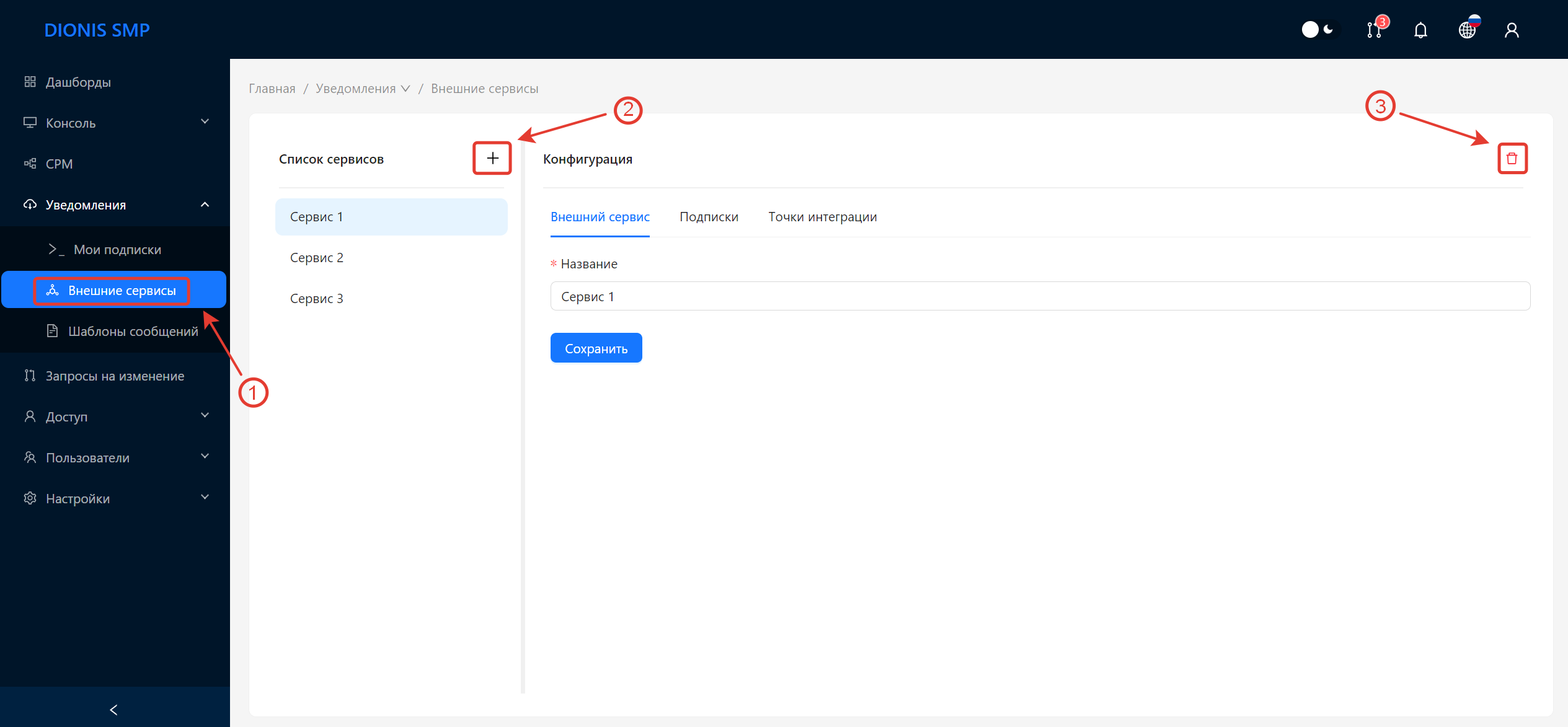


Рисунок 128. Настройка внешних сервисов

При выборе существующего внешнего сервиса становится доступным область настройки конфигурации. Она включает в себя вкладки: «Внешний сервис», «Подписки» и «Точки интеграции».

Для создании нового внешнего сервиса необходимо нажать на кнопку создания (см. Рисунок 128. Настройка внешних сервисов и указатель №2 данного рисунка ) . После нажатия открывается вкладка «Внешний сервис», в которой необходимо присвоить название. В дальнейшем она может быть использована для его редактирования.

Для удаления внешнего сервиса необходимо нажать на кнопку удаления (см. Рисунок 128. Настройка внешних сервисов и указатель №3 данного рисунка ).

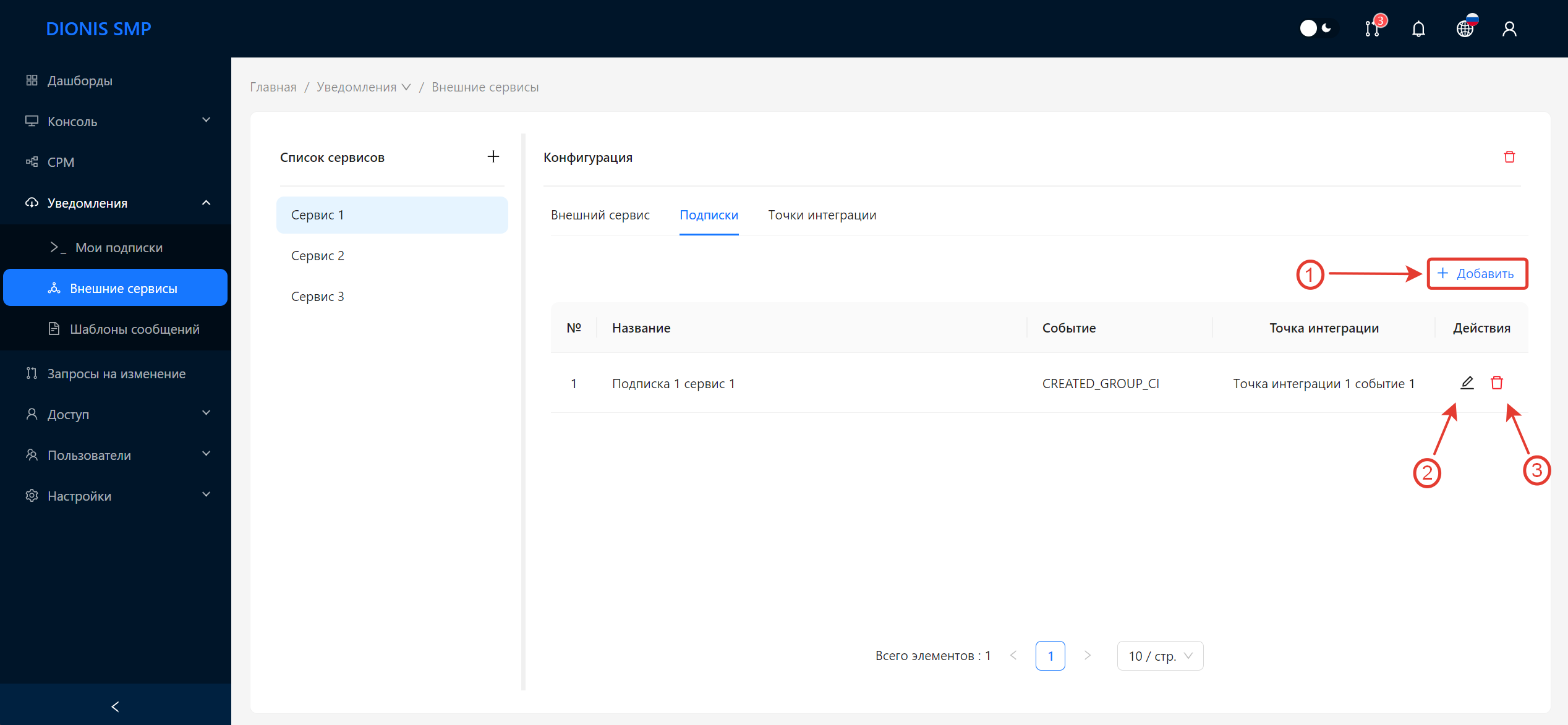


Рисунок 129. Настройка подписки внешнего сервиса

Управление подписками осуществляется с помощью вкладки «Подписки» (см. Рисунок 129. Настройка подписки внешнего сервиса ).

Для редактирования и удаления подписки используются соответствующие кнопки в колонке «**Действия**» (указатели №2 и №3 названного рисунка ) . Для добавления нового шаблона используется кнопка «**Добавить**» (указатель №1 названного рисунка).

При добавления новой подписки открывается модальное окно настройки нового шаблона (см. Рисунок 130. Окно создания новой подписки ).

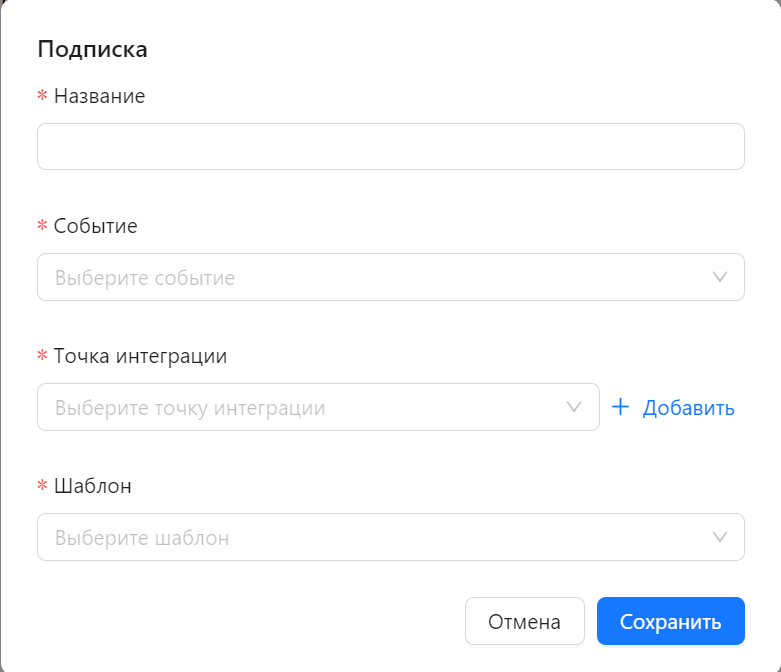


Рисунок 130. Окно создания новой подписки

Для завершения создания необходимо заполнить название, выбрать событие, на которое вы подписываетесь, точку интеграции и шаблон сообщения (см. Шаблоны сообщений ).

Все поля являются обязательными для заполнения.

Точка интеграции необходимо выбрать из уже существующих точек для выбранной подписки или же создать с помощью кнопки «**Добавить**». При добавлении открывается модальное окно создания точки интеграции.

Управление точками интеграции осуществляется с помощью вкладки «Точки интеграции» (см. Рисунок 131. Вкладка управления точками интеграции ).

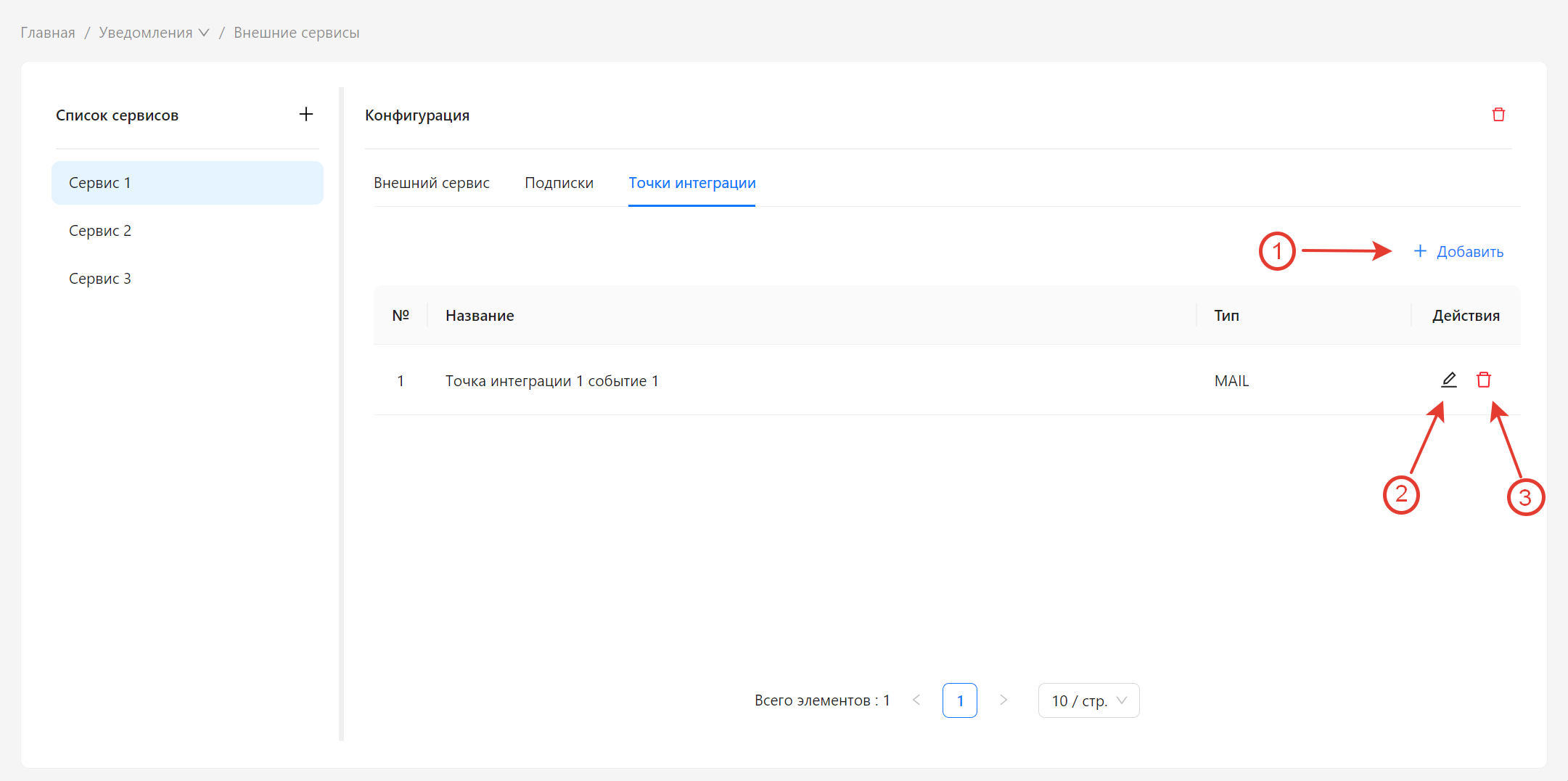


Рисунок 131. Вкладка управления точками интеграции

Для редактирования и удаления подписки используются соответствующие кнопки в колонке «**Действия**» (см. Рисунок 131. Вкладка управления точками интеграции и указатели №2 и №3 данного рисунка ). Для добавления нового шаблона используется кнопка «**Добавить**» (см. Рисунок 132. Окно создания новой точки интеграции и указатель №1 данного рисунка ).

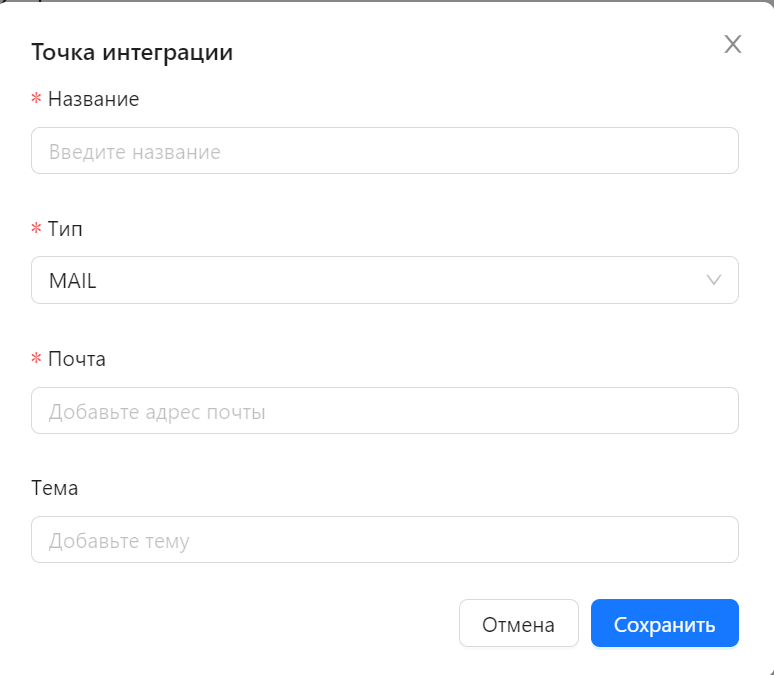
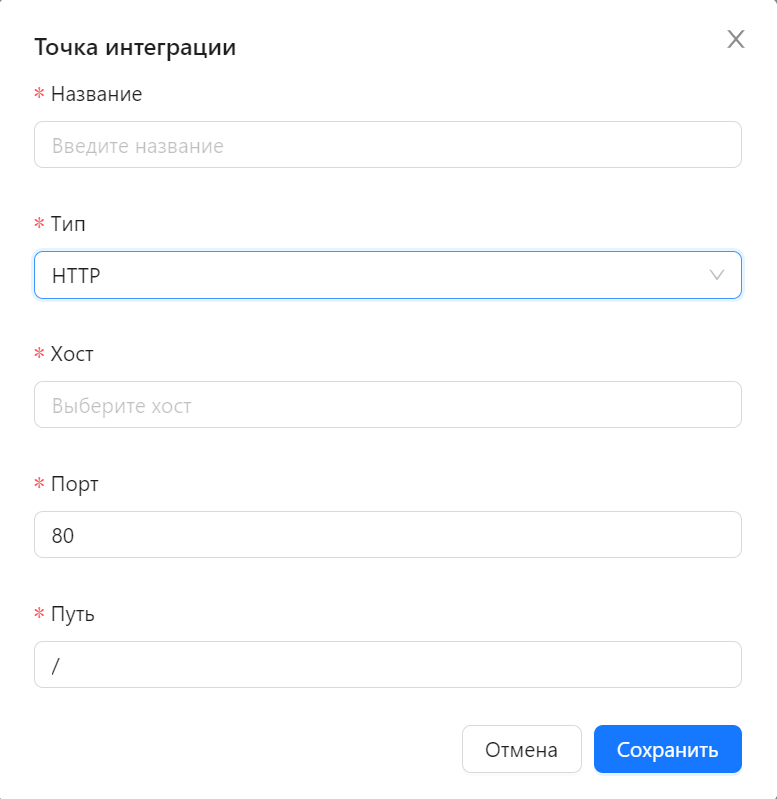


Рисунок 132. Окно создания новой точки интеграции

Для создания новой точки интеграции необходимо ввести название и выбрать тип: «HTTP» или «MAIL».

Для типа «HTTP» дополнительно указываются хост, порт и также путь. Для типа «MAIL» дополнительно указываются адрес электронной почты и тема письма. Все поля, кроме поля «Тема», являются обязательными для заполнения.

#### Шаблоны сообщений

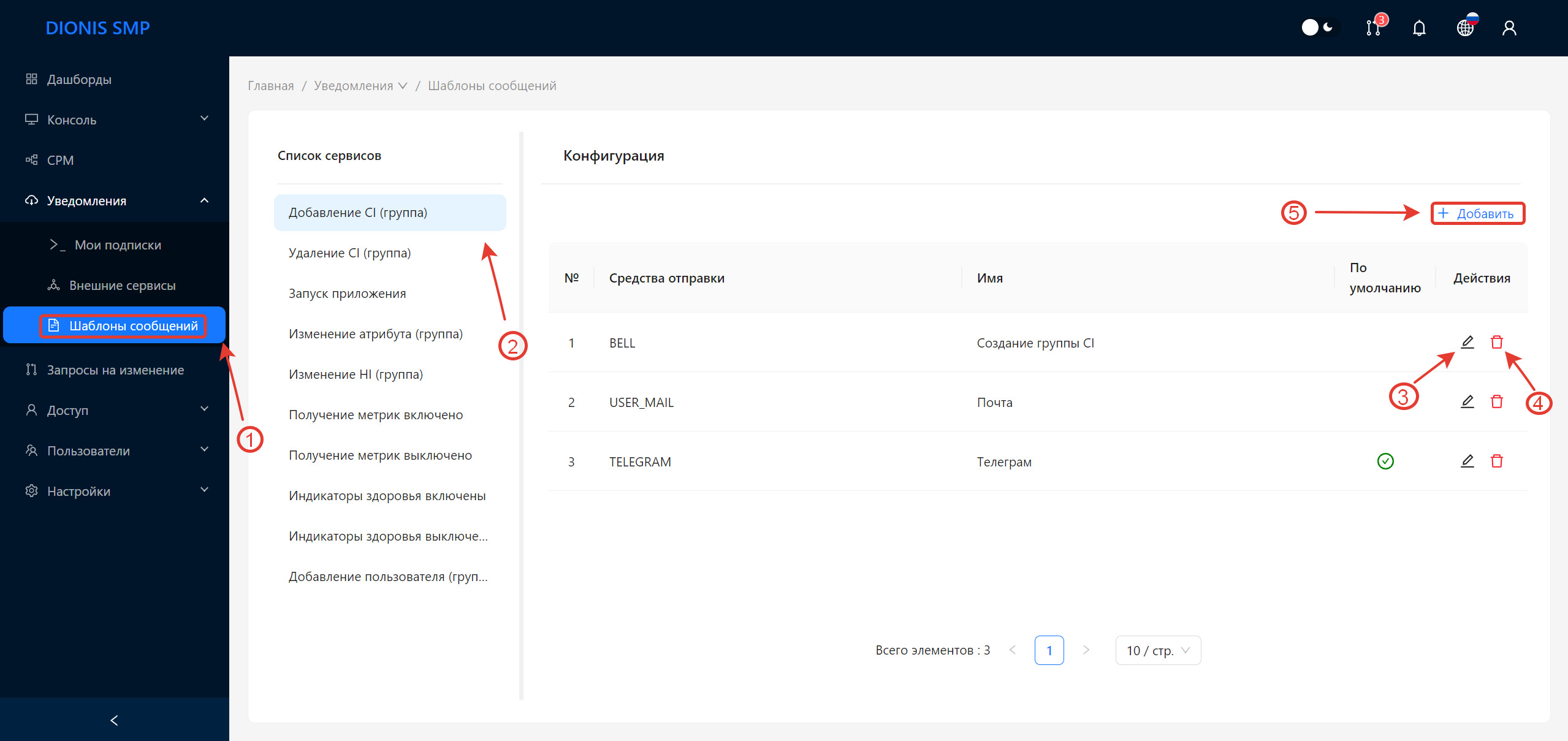


Рисунок 133. Управление шаблонами сообщений

Для настройки шаблонов сообщений уведомлений необходимо в навигационном меню перейти в вкладку «**Уведомления**», далее «**Шаблоны сообщений**» (см. Рисунок 133. Управление шаблонами сообщений и указатель №1 данного рисунка ).

В колонке «**Список сервисов**» необходимо выбрать один из следующих сервисов (см. Рисунок 133. Управление шаблонами сообщений и указатель №2 данного рисунка ):

* добавление CI (группа);
* удаление CI (группа);
* запуск приложения;
* изменение атрибута (группа);
* изменение Hi (группа);
* получение метрик включено;
* получение метрик выключено;
* индикаторы здоровья включены;
* индикаторы здоровья выключены;
* добавление пользователя (группа).

Далее, станут доступны просмотр, редактирование и удаление текущих шаблонов для выбранного сервиса, а также создание новых.

Для редактирования и удаления текущего шаблона используются соответствующие кнопки в колонке «**Действия**» (см. Рисунок 133. Управление шаблонами сообщений и указатели №3 и №4 данного рисунка). Для добавления нового шаблона используется кнопка «**Добавить**» (см. Рисунок 133. Управление шаблонами сообщений и указатель №5 данного рисунка»).

При добавления нового шаблона открывается модальное окно настройки нового шаблона (см. Рисунок 134. Окно настройки нового шаблона сообщений ).

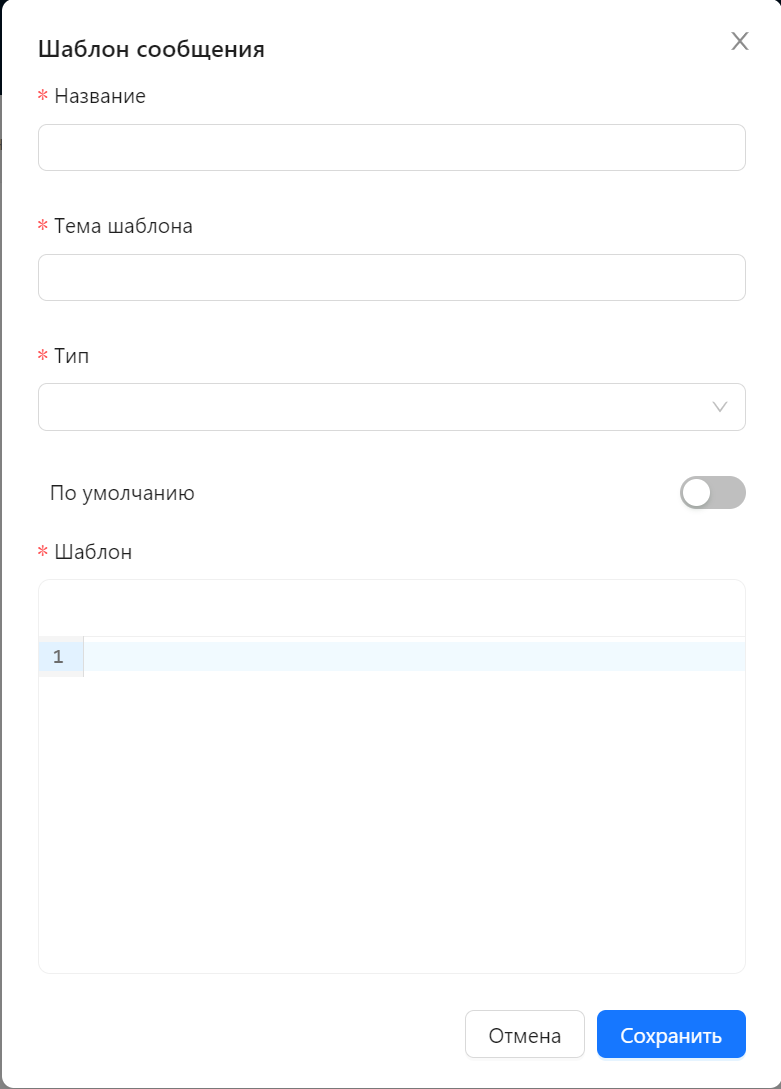


Рисунок 134. Окно настройки нового шаблона сообщений

В поле «Тип» необходимо выбрать один из типов доставки сообщения до пользователя: сообщение через Telegram, письмо на электронную почту или уведомление на странице приложения.

В поле «По умолчанию» положение тумблера определяет, будет ли шаблон являться шаблоном по умолчанию для выбранного типа доставки сообщения до пользователя.

В поле «Шаблон» необходимо ввести непосредственно сам шаблон сообщения.

Все поля окна создания нового шаблона являются обязательными для заполнения.

1. Наименование КЕ проставляется в момент генерации цели и не обновляется при изменении по правилу наименования [↑](#footnote-ref-2)
2. Наименование КЕ проставляется в момент генерации цели и не обновляется при изменении по правилу наименования [↑](#footnote-ref-3)