G3AppAdministrationGuide

Выпуск 0.0.15

"Бизнес Технологии" ООО

апр. 01, 2025

Содержание

1 Предисловие

Добро пожаловать в `Руководство администратора GlobalFramework`. Цель руководства предоставить информацию о принципах и способах администрирования решений на основе `GlobalFramework`.

1.1 На кого ориентировано руководство

Руководство ориентировано на:

- Разработчиков системы
- Аналитиков, настраивающих доступ к системе на проектах

2 Введение

Действие пользователя в системе требует наличия у него тех или иных привилегий. Администрирование доступа производится путем выдачи `пользователям` прав на `объектные привилегии` или `элементарные привилегии`.

2.1 Пользователь

Для работы в системе необходимо зайти под пользователем системы. С точки зрения системы пользователем является учетная запись, на которую выдается перечень прав, дающий возможность юридическому или физическому лицу выполнять действия в системе.

Список пользователей можно открыть из пункта меню:

- `Администратор \ Доступ > Пользователи`

Ограничение имени пользователя

Имя пользователя имеет вид <локальная_часть>@<домен>, домен не является обязательной частью имени пользователя.

Имя пользователя уникально в пределах системы.

Общая длина имени пользователя не должна превышать 1024 байт (октет), включая символ @

Ограничения локальной части (До последнего символа @):

- Прописные и строчные буквы латинского алфавита: (A-Z), (a-z)
- Цифры: (0-9)
- Специальные символы: !#$%&»*+-/=?^_`{|}~

- Символ (.), если он не является первым или последним, а так же если они не идут друг за другом.
- Следующие специальные символы разрешены, только если локальная часть выделена в кавычки: пробел и «(),:;<>[]

Ограничения домена (После последнего символа @):

- Прописные и строчные буквы латинского алфавита: (A-Z), (a-z)
- Цифры: (0-9)
- Специальный символ: -, если он не является первым или последним

Имя пользователя должно составляться по стандарту RFC 822

Длина имени используется из стандарта Active Directory

Отключение администрирования

В случае, если у пользователя стоит признак `Супер-пользователь`, настройки администрирования не применяются, под таким пользователем можно совершать в системе любые действия. Данный признак удобно использовать для разработки и тестирования, так как это позволяет отделить администрирование от разработки.

2.2 Администрируемый объект

`Администрируемый объект` - это логически связанный для массовой раздачи прав набор привилегий. Административный объект генерируется по объектам системы и позволяет единым образом выдавать права на них.

Администрируемый объект генерируется в момент установки или обновления системы по таким объектам как:

- Произвольная выборка
- Класс
  - Справочник
  - Настройка
  - Журнал
  - Документы
- Пакеты
- Приложения

Список административных объектов можно открыть из пункта меню:

- `Администратор \ Настройки > Администрируемые объекты`

Отключение администрирования

Возможно отключить администрирование в разрезе административного объекта, данным механизмом удобно пользоваться в случае поэтапной настройки прав, когда администрирование включается только для тех объектов, на которых оно уже настроено. Для отключения администрирования на административном объекте существуют следующие поля:

- `Не распространяются настройки администрирования`
  Если флаг установлен, на объекты представляющие состав административных объектов не распространяются настройки администрирования
- `Не требуется настройка прав доступа на состояния`
  Если флаг установлен, на документе данного административного объекта выключается контроль доступных переходов между состояниями.

Узел администрирования

Группирует элементы администрируемого объекта для массовой раздачи прав, по умолчанию создается одна основная группа на каждый администрируемый объект.

Узел администрирования зарезервирован на случай, если в системе потребуется настраивать права в зависимости от `типа объекта` или `состояния`

Элементы администрируемого объекта

Элемент административного объекта создается на объекты которые требуют раздачи прав на `элементарные привилегии`.

Выборка

По умолчанию выборка со всеми отображениями регистрируется в один `элемент администрируемого объекта` с именем `{SelName}#Default`. При этом на каждую операцию и атрибут в разрезе системного имени создается `элементарная привилегия`.

Для переноса отображения в отдельный `элемент администрируемого объекта` существует аннотация:

```
@AcItemRep(name = "{Name}")
```

В таком случае `элемент администрируемого объекта` получит имя `{SelName}#{Name}` Так же `элемент администрируемого объекта` можно задать в `avm.xml` через атрибут отображения `acItemRep`

Примечание: Изменения вступят в силу после обновления административного объекта (см. главу `Групповая настройка привилегий ролей`)

Для создания `элементов администрируемого объекта` по выборкам, не имеющим класса, в `avm` файле обязательно должен быть тег `<acObject/>`. Для выборок без класса формируется `администрируемый объект` и `элемент администрируемого объекта` с суффиксом `Avi` в имени.

Дискретные привилегии

Дискретные привилегии используют правила дискретного доступа при проверке прав, что позволяет раздавать доступ по объектно (в зависимости от значений атрибутов строки). В интерфейсе настройки ролей дискретные привилегии можно выдать с ограничением по `дискретном правилу`.

Элементарные привилегии

`Элементарные привилегии` используются для разграничения действий пользователя и создаются следующим образом:

- по выборке на:
  - Операции
    Создаются дискретные привилегии
  - Атрибуты
  - Сеттеры атрибута
    Создаются дискретные привилегии

Типы элементарных привилегий

- Чтение
  Содержит настройки видимости атрибутов.
- Редактирование
  Содержит сеттеры атрибутов.
- Добавление
  Содержит операции вставки и копирования.
- Удаление
  Содержит операция удаления.
- Интерактивные права
  Содержит остальные операции.

Для изменения типа привилегии, которой соответствует операция выборки, существует аннотация `@AcPrivilegeType`. Пример:

```
@AcPrivilegeType(value = PrivilegeTypes.Read)
def myOperation(): Unit = {}
```

Примечание: Изменения вступят в силу после обновления административного объекта (см. главу `Групповая настройка привилегий ролей`)

Объектные привилегии

`Объектные привилегии` позволяют раздавать права на бизнес логику.

Авто генерируемые объектные привилегии

- edit#
  Создаются дискретные привилегии
- view#
  Создаются дискретные привилегии
- viewReport#
  Создаются дискретные привилегии

Привилегии на переход состояний

Привилегии на переход состояний ограничивают доступные переходы состояния. Ограничения на переходы реализовано в стандартном выпадающем списке для состояний `ru.bitec.app.btk.Btk_ClassStateAvi.Lookup_Class`

Имя привилегии формируется по правилу:

```
s"StateChange_OT:${idObjectType}_SS:${idStartState}_FS:$idFinishState"
```

Функция для определения возможности перехода: `ru.bitec.app.btk.Btk_AdminPkg#hasStateChangePriv`.

Ручное создание объектных привилегий

Для ручного создания объектных привилегий на классе(`odm.xml`) или пакете(`pkg.xml`) используются xml разметка, пример:

```
<admin>
    <privileges>
        <privilege name="SbtManage" caption="Управление решением"/>
    </privileges>
</admin>
```

Примечание: Изменения вступят в силу после обновления административного объекта (см. главу `Групповая настройка привилегий ролей`)

Приложение

Для администрирования приложений создается адм. объект `Приложения`(`Btk_ApplicationsListPkg`). Объектные привилегии импортируются в него из xml файлов `META-INF/applications.xml` всех подключенных модулей.

Выданные права на такую привилегию обеспечивают отображение приложения в списке в меню выбора приложений.

Дискретные ограничения

Дискретные ограничения позволяют задавать выражения для создания правил выдачи дискретных привилегий на конкретной роли. Подробности смотрите в главе `Дискретный доступ` Принцип настройки дискретных ограничений:

1. Администратор определяет дискретные ограничения на административном объекте
2. Администратор определяет дискретные правила в роли
   Каждое правило содержит дискретное ограничение и набор параметров
3. Администратор выдает грант на дискретную привилегию роли
   При этом при синхронизации пользователей проходит расчет дискретных привилегий данных пользователю по алгоритму объединения грантов

При проверки привилегии в бизнес логике, в функцию проверки передается перечень строк, для которых выполняется проверка. При ограничении запроса по дискретной привилегии, каждое ограничение добавляется в where условие.

Алгоритм объединение грантов:

1. Пусть `g1` и `g2` гранты пользователю на объектную привилегию `p`
2. Если `g1` или `g2` с типом полный доступ, вернуть грант на `p` с полным доступом
3. Иначе вернуть грант на `p` объединив ограничения

Алгоритм объединения ограничений:

1. Пусть `A` массив ограничений для грантов `g1` и `g2`
2. Пусть `S` пустое множество ограничений(параметры ограничения не в ходят в ключ множества)
3. Для каждой `g` из `A`
   1. Если `g` существует в `S`, добавить новые параметры к ограничению в `S`
   2. Иначе добавить `g` к `S`
4. Вернуть `S`

Скрипт фильтрации

Используется в универсальной фильтрации для ограничения видимости по объектам.

Скрипт проверки объектного кэша

Используется в автономной бизнес логики для проверки возможности действия со строкой.

2.3 Роль

`Роль` - административная единица для выдачи прав. Для раздачи дискретных привилегий на роли могут быть настроены дискретные правила, которые определяют дискретные ограничения и передаваемые в них на гранте параметры.

В роле выдаются гранты на привилегии.

Список ролей можно открыть из пункта меню:

- `Администратор \ Доступ > Роли`

2.4 Профиль пользователя

`Профили пользователя` используется для группировки ролей для более удобной раздачи их пользователю.

Список профилей можно открыть из пункта меню:

- `Администратор \ Доступ > Профили`

2.5 Замещение прав

Позволяет указать какой пользователь будет обладать всеми правами другого и на какое время. Настройки замещения прав учитываются при выполнении индексации прав пользователя. Для отслеживания окончания периода действия замещения каждую ночь выполняется задание, которые выполняет индексацию пользователей.

Список замещений можно открыть из пункта меню:

- `Администратор \ Доступ > Замещение прав`

2.6 Индексация привилегий пользователей

Для быстрой проверки привилегий пользователя используются `индексы привилегий`. Индексы привилегий необходимо пересчитывать после завершения редактирования ролей и их выдаче пользователю.

2.7 Предустановленные профили

Профиль «Отладчик»

`Профиль "Отладчик"` предназначен для пользователей, для которых нужно предоставить доступ к инструментам отладки. Таким как:

- доступ к операциям управления решением (управление SBT)
- возможность открыть дебаггер
- выполнение jexl скриптов в дебагере
- доступ к инструментам меню Сервиса

- доступ к инструментам отладки клиента

Профиль «Разработчик»

`Профиль "Разработчик"` предназначен для пользователей, которые являются разработчиками. Включает в себя все права профиля «Отладчик». А также имеет доступ к следующим инструментам:

- доступ к редактированию справки

3 Приложение администратор

Приложение `Администратор` используется для администрирования пользователей.

3.1 Список пользователей

Список пользователей можно открыть из пункта меню:

- `Администратор \ Доступ > Пользователи`

Операции списка пользователей:

- `Пересчитать индексацию привилегий для текущего пользователя`
  Операция предназначена для обновления индексов системных привилегий для текущего пользователя. В результате операции обновятся записи в индексе(`Btk_AcUserItemPrivFlat`) по всем элементам администрируемых объектов на привилегии, которых у пользователя есть права, а также записи в индексе(`Btk_AcUserObjPrivFlat`) по всем администрируемым объектам, на объектные права которых у пользователя есть права.
- `Пересчитать индексацию привилегий для всех пользователей`
  Операция предназначена для обновления индексов системных привилегий для всех пользователей.

3.2 Карточка пользователя

Операции карточки пользователя:

- `Изменить пароль`
  Операция предназначена для установки пароля для входа в систему для пользователя.
- `Пересчитать индексацию привилегий для текущего пользователя`
  Операция предназначена для обновления индексов привилегий для текущего пользователя.

3.3 Детализация пользователя

В карточке и списке доступна детализация, содержащая закладки:

- `Профили доступа`
  Список записей в классе-развязке(`Btk_AcUserGrant`) со ссылкой на профиль. Для закладки `Профили доступа` доступна детализация с закладкой `Роли доступа профиля`, отображающая список записей в классе-развязке(`Btk_AcProfileGrant`) со ссылкой на роль
- `Роли доступа`
  Список ролей, связанных с пользователем через профиль

- `Индексация элементарных привилегий`
  Список записей в `Btk_AcUserItemPrivFlat` для текущего пользователя.
- `Индексация объектных привилегий`
  Список записей в `Btk_AcUserObjPrivFlat` для текущего пользователя.

Для закладок `Индексация элементарных привилегий` и `Индексация объектных привилегий` доступна операция

- `Посмотреть значение индекса`
  Операция открывает в модальном окне содержимое поля(`cPriv`) с перечнем доступных привилегий для элемента или объекта.

Синхронизация с Active Directory

Для синхронизации:

1. Выполните операцию `Синхронизация с Active Directory`
   Операция доступна в списке пользователей. При этом откроется интерфейс синхронизации отображающий список логов.
2. При необходимости укажите шаблон в шапке интерфейса
   Вновь созданные пользователи будут созданы по этому шаблону
3. Нажмите кнопку `Синхронизировать`
   При этом будут просканированы домены на наличие групп и пользователей. Произойдет синхронизация пользователей

Синхронизируемы поля пользователей:

- `Системное имя`
  Атрибут `sAMAccountName`, если он пустой, то берётся из `uid`
- `Пароль`
  Равен системному имени при создании пользователя
- `Группа`
  Определяется через атрибут `memberOf`
- `Подразделение Active Directory` Атрибут `department`
- `Наименование Active Directory` Атрибут `title`
- `Описание Active Directory` Атрибут `description`
- Для физ. лица:
  - `Фамилия`
    Первое слово из атрибута `displayName`, если в нём три слова. В ином случае атрибут `sn`.
  - `Имя`
    Второе слово из атрибута `displayName`, если в нём три слова. Первое слово из атрибута `givenName`, если в нём два слова. В ином случае атрибут `givenName` целиком.
  - `отчество`
    Третье слово из атрибута `displayName`, если в нём три слова. Второе слово из атрибута `givenName`, если в нём два слова.
  - `email`
    Атрибут `mail`

Синхронизируемые поля групп:

- Наименование
  Атрибут name, если он не указан, то полное имя. Например: CN=Domain users,CN=Users
- Системное имя
  Полное имя группы. Например: CN=Domain users,CN=Users

Пользователи создаются только в случае, если они не заблокированы в домене. Если пользователь заблокирован, и уже зарегистрирован в системе Global, то ему выставляется признак «Заблокирован».

Помимо непосредственной синхронизации пользователей, происходит поиск/создание физ. лиц для пользователей. Для тех пользователей, у которых нашлось/создалось физическое лицо, создаётся ещй и сотрудник в Bs_Employee. В случае если у группы пользователя настроена ещё и связь с ОФС, то для сотрудника создастся соответствующая запись в Bs_OFSEmployee. Если у связанной ОФС настроены профили доступа, то эти пользователи применятся и на пользователя.

Настройка синхронизации

Для настройки подключения используется универсальный класс настроек (Btk_Setting). Открыть настройки можно из пункта меню Настройка системы \ Настройки и сервисы > Настройки модулей системы > Общие настройки системы Далее следует открыть настройки модуля btk и выбрать настройку ldapSync Доступные параметры:

- Создавать физ. лицо
  Если параметр выключен - для пользователей, которые появились в результате синхронизации, будут искаться соответствующие физ. лица (Bs_Person). Поиск осуществляется по совпадению фамилии, имени и отчества. В физ. лице будет установлена ссылка на пользователя и почта (при наличии в AD).

Если параметр включен - для пользователей, у которых не удалось найти существующее физ. лицо, будет создано новое физ. лицо с ФИО и почтой из AD

- Тип синхронизации
  Все пользователи - Настройка фильтра по группе отключается и синхронизация возьмёт всех пользователей из выбранных доменов вместе с их группами. Группа переносится из AD только в случае, если она находится в одном из выбранных доменов. Если группа находится вне списка выбранных доменов, то она не переносится.

Пользователи группы - Настройка фильтра по группе включается и в выбранных доменов идёт поиск всех подгрупп группы указанной в фильтре. Далее ищутся пользователи, которые входят в найденные подгруппы.

- Фильтр по группе
  Механика работы описана выше. Задаётся в виде CN=...,OU=...,DC=...
- Домены
  В этом параметре можно задать список доменов, в которых будет осуществляться поиск, у каждого домена есть свои параметры:
  - Добавить домен к пользователю
    Если параметр активен, то к имени пользователя будет добавляться имя домена, в котором он был найден. В противном случае - имя пользователя будет браться из AD без изменений
  - Домен
    Пример: part.gs.local
  - Логин
  - Пароль

- URL
  Пример: ldap://192.168.2.11:389

Так же для применения профилей доступа на пользователей, можно задать связь между пользовательскими группами и ОФС в классе «Соответствие групп пользователей и офс» (Bs_OFStructureUserGroupLink). Тогда ко всем пользователям группы в будущем при синхронизации будет применяться профиль доступа настроенный на ОФС.

3.4 Список ролей

Список ролей можно открыть из пункта меню:

- Администратор \ Доступ > Роли

3.5 Карточка роли

Используется для выдачи привилегий для роли от дерева административных объектов.

Права роли

В данном интерфейсе отображается дерево административных объектов с уровнями:

- Группа администрируемых объектов(только чтение)
  - Объект администрирования
    - Узел администрирования
      - Элемент администрируемого объекта

Совет: По умолчанию отображаются только административные объекты использующиеся в роли. Для отображения всех административных объектов выключите флаг Только объекты, на которые у роли имеются права в панели фильтрации.

В данной интерфейсе каждый тип элементарной привилегии отображен отдельным столбцом, что позволяет выдать права массово по типу для всех элементарных привилегий этого типа для текущего уровня и уровней ниже.

Расшифровка прав

В случае если выбран объект администрирования отображает:

- перечень типов элементарных привилегий
- перечень объектных привилегий

В случае если выбран элемент администрируемого объекта отображает:

- перечень элементарных привилегий для данной строки

Перевод состояний

Отображается в случае если выбран объект администрирования. Позволяет настроить возможные переходы между состояниями.

Выдача прав на объектные привилегии

1. Выберите требуемый адм. объект.
   При этом в расшифровке прав отобразятся объектные привилегии
2. Раздайте права на объектные привилегии роли \
3. Обновите индексы системных привилегий

Выдача прав на тип элементарных привилегий

В случае если выдано право на тип элементарной привилегии, доступ будет дан всем элементарным привилегиям данного типа во всех элементах администрируемого объекта.

1. Выберите требуемый адм. объект
   При этом в расшифровке прав отобразятся типы элементарных привилегий.
2. Раздайте права на типы элементарных привилегий
   Права на типы можно раздавать либо в расшифровке пара либо в дереве административных объектов
3. Обновите индексы системных привилегий

Выдача прав на элементарную привилегию

1. Выберите требуемый `элемент администрируемого объекта`
   При этом в расшифровке прав отобразятся типы элементарных привилегий.
2. Раздайте права на типы элементарных привилегий
   Для этого в расшифровке прав выберите нужный элемент и предоставьте к нему доступ.
3. Обновите индексы системных привилегий

Примечание: Если установить признак «Запрещено», то пользователи, обладающие этой ролью, не будут иметь прав на запрещенную привилегию, даже если другие роли дают на нее доступ.

Выдача прав на вызов Api и Pkg методов в jexl-контексте

Примечание: Для того, чтобы при выполнении jexl-скриптов учитывались настроенные права, необходимо включить в общих настройках модулей btk - «Выполнение jexl-скриптов через безопасный диалект».

---

1. В карточке `Роли` на закладке `Права вызовов через jexl-script` слева отображен список Api и Pkg классов проекта. Справа - их методы.
2. Для выбора/снятия доступа ко всем методом класса используйте чек-бокс `Доступны все методы`, в списке методов можно настроить доступ к конкретному методу класса.

3. Обновите индексы системных привилегий

Выдача прав на пакеты и классы в jexl-контексте

Примечание: Для того, чтобы при выполнении jexl-скриптов учитывались настроенные права, необходимо включить в общих настройках модулей btk - «Выполнение jexl-скриптов через безопасный диалект».

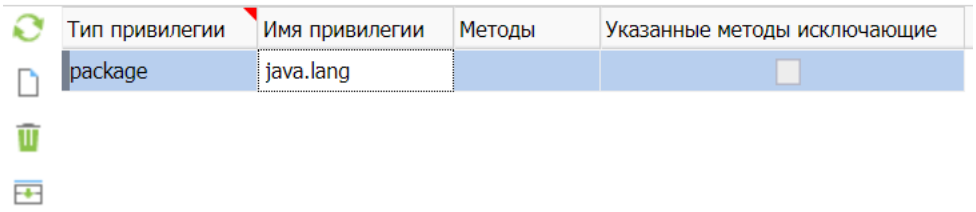
---

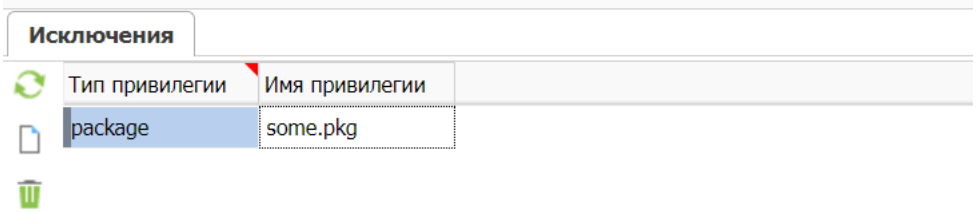
1. В карточке `Роли` на закладке `Права на пакеты/классы из jexl` можно задавать пакеты и классы, к которым у пользователя должен быть доступ:

- `Тип привилегии` - указывается `package` или `class` для пакета или класса, соответственно
- `Имя привилегии` - полное имя пакета или класса
- `Методы` - методы класса, на которые надо выдать права (если пусто, то доступны все методы). Указываются в квадратных скобках, через запятую, каждый метод обернут в кавычки - `["methodName1", "methodName2"]`
- `Указанные методы исключающие` - если включить галку, то будут доступны все методы класса, кроме указанных

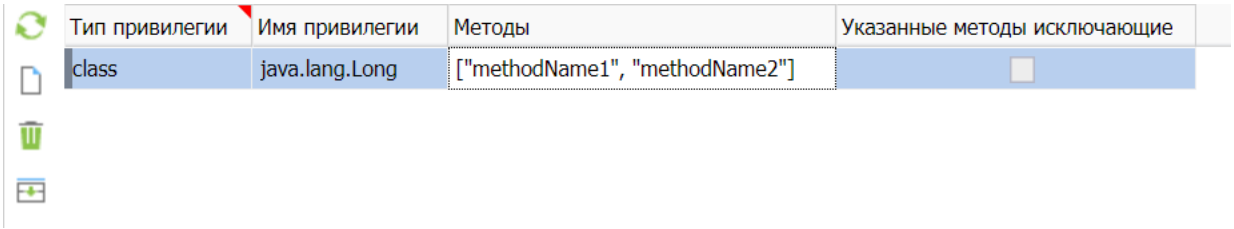
1. В детализации на закладке `Исключения` для пакетов можно указывать их внутренние пакеты или классы, на которые доступ выдаваться не должен.

Пример выдачи прав на пакет `java.lang`, исключая вложенный пакет `java.lang.some.pkg`





Пример выдачи прав на методы `methodName1` и `methodName2` класса `java.lang.Long`



Выдача прав на использование Rest-пакетов

1. В карточке `Роли` на закладке `Права на Rest-пакеты` отображен список пакетов. Чекбокс `Rest-пакет` указывает, является ли пакет Rest-пакетом.
2. Для выдачи\отзыва доступа используйте чекбокс `Имеет доступ`.
3. Обновите индексы системных привилегий по роли.

Примечание: Обновление списка Rest-пакетов происходит при обновлении адм.объектов по модулю. С пакетов, которые перестали быть Rest-пакетами, чекбокс `Rest-пакет` автоматически снимается.

3.6 Групповая настройка привилегий ролей

Выборка `Групповая настройка привилегий ролей` предназначена для массовой раздачи привилегий по ролям от административного объекта.

Выборку можно открыть из пункта меню:

- `Администратор \ Доступ > Групповая настройка привилегий ролей`

Общий принцип работы с выборкой:

1. Выберите требуемый административный объект
2. Выберите требуемые роли
   Для этого в панели фильтрации привилегий откройте список подбора для поля `Перечень ролей`
3. Проставьте гранты на требуемых привилегиях
4. Пересчитайте индексы привилегий
   Для этого выполните операцию `Пересчитать индексацию привилегий` в интерфейсе `Перечень ролей`

Дерева администрируемых объектов

Дерево администрируемых объектов состоит из следующих уровней:

- Группа администрируемых объектов(Опционально)
  - Объект администрирования
    - Узел администрирования
      - Элемент администрируемого объекта

Объекты администрирования могут не входить в группу, тогда они будут считаться корневыми записями.

Операции дерева администрируемых объектов

- `Обновить выбранный объект`
  Операция предназначена для обновления выделенного объекта или элемента адм. объекта. При обновлении административного объекта происходит сканирование исходного кода по которому добавляются недостающие привилегии и элементы.
- `Обновить адм. объект по имени`
  При этом произойдет запрос имени и произойдет обновление административного объекта по выбранному имени
- `Обновить адм. объекты модуля`
  Операция предназначена для обновление адм. объектов по всему модулю

Примечание: Классы-коллекции не имеют собственных объектов администрирования и входят в объект мастера. Поэтому для обновления настроек для коллекций необходимо обновлять адм. объект по имени класса мастера.

Настройка прав

Закладка отображает настройку прав для ролей в зависимости от выделенной записи в дереве администрируемых объектов:

1. Для административного объекта:

- объектные привилегии

1. Для элемента администрирования:

- типы элементарных привилегий
  - элементарные привилегии

Колонки ролей в настройке прав формируются динамически по ролям из фильтра.

Операции настройки прав:

- `Подобрать роли`
  Операция предназначена для подбора в фильтр ролей, для которых есть какие-либо права для текущего объекта или элемента.

Пользователи

Отображаются пользователи для роли в активной ячейки.

Профили

Отображаются профили для роли в активной ячейки.

3.7 Индексация привилегий для пользователей

Индексация пользователя

Для пересчета индекса по 1-му пользователю существует операция `Пересчитать индексацию привилегий для текущего пользователя` Операция доступна:

- в списке и карточке `Пользователей`
- на закладке `Пользователи роли` в списке и карточке Ролей.

Индексация всех пользователей

Для индексации привилегий всех пользователей предназначена операция `Пересчитать индексацию привилегий для всех пользователей`. Операция доступна:

- в списке `Пользователей`.

3.8 Аудит прав доступа

Историю изменений прав доступа можно посмотреть в `Аудит прав доступа`, находящемся в пункте меню `Аудит`.

Примечание: Для включения аудита изменений прав доступа необходимо выполнить операцию `Включить аудит для структур прав доступа` в этом меню.

В самом типе необходимо выбрать `Вид объекта` (Пользователь, Профиль или Роль), `Объект` и, если необходимо, период времени.

Пользователи

Для пользователей в аудите отображаются `Профили`, которые были выданы пользователю или отобраны у него, дата изменения и изменивший пользователь.

Профили

Для профилей в аудите отображаются `Роли`, которые включали в профиль или убирали из него, и `Пользователи`, которым выдавался профиль или его отбирали, а также дата изменения и изменивший пользователь.

Примечание: В списке изменений для `Пользователей` и `Профилей` есть операция `Изменения по объекту`, которая позволяет перейти к аудиту другого объекта, который является выделенным.

Роли

Для ролей список изменений представляет собой две выборки:

- `Дерево администрируемых объектов`
  Отображаются администрируемые объекты или элементы администрирования, по которым для данной роли были совершены изменения
- `Список привилегий конкретного администрируемого объекта`
  Список привилегий конкретного адм. объекта или элемента администрирования, которые изменялись для данной роли

Список привилегий представляет из себя:

1. Имя адм. объекта или элемента администрирования
2. Тип привилегии:
   - Изменение атрибута (например, при простановке `Не распространяются настройки администрирования` для адм. объекта),
   - Объектная привилегия,
   - Элементарная привилегия,
   - Группа элементарных привилегий (когда изменяются доступ не на конкретные элементарные привилегии, а на всю группу, например, Редактирование)
   - Перевод состояния
1. Системное имя привилегии
2. Привилегия - наименование привилегии
3. Изменивший пользователь
4. Дата изменения
5. Системное имя атрибута
6. Наименование атрибута
7. Старое значение - старое значение атрибута
8. Новое значение - новое значение атрибута

Примечание: Для списка привилегий есть фильтр `С входящими`, который в случае, когда в дереве адм. объектов выбран адм. объект, а не элемент администрирования, отображает изменения не только администрируемого объекта, но и элементов администрирования, входящих в этот объект.

3.9 Трассировка прав доступа

Запуск трассировки прав пользователя

Для запуска трассировки прав пользователя, выполните следующие шаги:

- Перейдите в раздел «Сервис» в меню системы.
- В меню «Инструменты» найдите опцию «Начать трассировку прав» и выберите её.

После начала трассировки, система будет отслеживать все ваши действия. Выполняйте необходимые действия с данными в системе, так как вам требуется.

Чтобы завершить трассировку прав пользователя, выполните следующие шаги:

- Перейдите снова в раздел «Сервис» в меню системы.
- В меню «Инструменты» выберите опцию «Закончить трассировку прав».

Отчет сессии трассировки прав

Отчет «сессии трассировки прав пользователей» выведен в приложении «Администратор», в меню «Отчеты». Отображает список сессий трассировки. При открытии карточки сессии в детализации будет отображена более подробная информация по проверкам прав, которые вызывались в ходе сессии трассировки.

Очистка записей

Сессии трассировки сохраняются в таблицу Btk_AcTraceSession, а детализация сессии в Btk_AcTraceJournal. Для очистки этих таблиц предусмотрен по умолчанию регистрируется задание очистки данных по классу «Btk_AcTraceSession», удаляющий записи, хранящиеся больше 30 дней.

Реализация трассировки

При запуске трассировки

- в параметрах рабочей сессии сохраняется параметр активности трассировки;
- создается новая запись в Btk_AcTraceSession в логирующей транзакции;
- в параметрах рабочей сессии сохраняется параметр с идентификатором запущенной сессии трассировки;
- для уже открытых выборок подключается подписка на события, так же она подключается на onLoadAdminMeta при открытии новой выборки; При действиях пользователя во время трассировки
- на методах проверки прав доступа, вызов операции/сеттера на администрируемых выборках, создается запись в логирующей транзакции в Btk_AcTraceJournal; При завершении трассировки
- вызывается коммит логирующих транзакций со всех запущенных сессий выборок;
- в сессию трассировки Btk_AcTraceSession сохраняется дата завершения;
- отключается подписка на события, подключенная для логирования;
- из параметров рабочей сессии удаляются параметр активности трассировки и идентификатор сессии трассировки.

4 Дискретный доступ

Дискретный доступ позволяет выдавать привилегии в рамках объектов классов или записей в администрируемой выборке.

4.1 Настройка привилегии

Административный объект

1. Открыть карточку `Адм. объекта`
2. Установить признак `Дискретный доступ`
3. Перейти на закладку `Дискретные ограничения доступа`
4. Создать `Дискретное ограничение`.

Дискретное ограничение доступа

1. Создать новую запись или встать на запись для редактирования
2. Указать уникальное в рамка адм. объекта имя и наименование
3. Настроить тип правила
4. Настроить параметры правила.
5. На закладке `Скрипт для фильтрации объектных привилегий` написать sql-запрос, который будет фильтровать списки объектов
6. На закладке `Скрипт проверки строк по объектному кешу` написать jexl-скрипт, который будет по rop-у объекта или строке выборки возвращать признак, что условие выполнилось для этого объекта.

Тип правила

Доступны 3 варианта типа правила:

- Примитивное
- Составное
- Без параметров

Примитивное правило

Состоит из одного параметра.

При анализе прав пользователя со всех его ролей будут собраны значения параметров и собраны в один массив примитивных значений. Например для числового правила будет собран массив вида: `[10, 20, 30]`

Составное правило

Состоит из нескольких параметров.

При анализе прав пользователя используется следующий алгоритм:

1. Для каждого набора значений, указанных на роли, формируется json-объект вида:

```
{
  <ИД параметра 1>: [<Массив указанных значений>],
  <ИД параметра 2>: [<Массив указанных значений>],
  ...
  <ИД параметра n>: [<Массив указанных значений>]
}
```

2. Каждая роль может обладать несколькими наборами значений на одно правило. По этому результатом агрегации настроенных значений на роли будет массив, содержащий объекты, указанные в п.1:

```
[
  {json-объект для набора 1},
  {json-объект для набора 2},
  ...
  {json-объект для набора n},
]
```

3. При агрегации прав пользователя:
   - со всех ролей анализируются массивы, описанные в п.2
   - в json-е заменяются ИД параметров на их имена
   - каждый элемент массива приводится к строке
   - собирается результирующий массив, содержащий уникальные строки.

     Таким образом исключаются дублирующие настройки на ролях.

Пример агрегации прав пользователя:

```
[
  {
    "paramName1": ["a", "b", "c"],
    "paramName2": [1, 2, 3]
  },
  {
    "paramName1": ["c", "d", "e"],
    "paramName2": [3, 4, 5]
  }
]
```

Правило без параметров

Не содержит параметров.

При анализе прав пользователя, если хотя на одной роли будет правило без параметров, оно будет проверяться.

Параметры правила

Позволяют настроить тип данных и ссылочность для значений, которые будут указаны на ролях или определяться динамически, если на роли для параметра проставить флаг «Динамическое определение».

Если правило примитивное, то у него доступен для настройки только один параметр с системным именем «SimpleAcObjectRuleParam». Для составного правила доступны несколько параметров.

Динамическое определение значения параметров

Для некоторых случаев бывает удобнее не задавать на каждую роль набор параметров, а определять их динамически, например от текущего пользователя или даты. Для этого на роли в настройке правила дискретного доступа нужно:

- В карточке администрируемого объекта, на закладке с параметрами под динамическими параметрами написать jexl-скрипт, который возвращает массив строк Пример:

```
var idvUser = Btk_UserApi.getCurrentUserID(); //получаем id текущего пользователя
var res =[...];  //объявляем массив для сбора значений параметра
if (idvUser != null){
  var rvUser = Btk_UserApi.load(idvUser);
  var idvObjAttr = Btk_UserApi.getAttrValue(rvUser, "idObjAttr"); // получаем значение␣
↪атрибута с именем "idObjAttr", хранящийся для текущего пользователя
  if (!empty idvBisObj){
    res.add(idvBisObj.toString()); // добавляем полученное значение в массив
  }
}
res; // возвращаем массив со значениями
```

- На роли, на закладке с настройками дискретного доступа для нашего правила включаем флаг «Динамическое определение» у параметра.

Скрипт для фильтрации объектных привилегий

Используется для фильтрации списков объектов, через наложение макроса универсального фильтра `&DefUniFltMacros`.

Доступные макросы внутри текста скрипта:

- `(&<Имя атрибута>)`
  Будет заменено на указанное имя атрибута. Например, `t.id`
  Для фильтрации доступны все атрибуты дата-сета выборки шапки БО.
- `(&params)`
  Массив строк, в котором будут параметры, собранные со всех ролей пользователя.

- `(&CurrentUserID)`
  Будет заменено на ИД текущего пользователя.
- `(&CurrentUserName)`
  Будет заменено на имя текущего пользователя.

Запрос чаще всего представляет из себя `exist`, в котором джойнится таблица класса, и через массив значений накладываются условия фильтрации.

Скрипт проверки строк по объектному кешу

Используется для проверки ограничения по объектам, используя объектный кеш.

Представляет из себя `jexl-скрипт`, который возвращает `true`, если объект проходит условие, и `false` - если не проходит.

Доступные переменные:

- `row` \
  – Если адм. объект создан по классу, то в этой переменной будет `rop` проверяемого объекта. Это не исходный `rop`, а его обертка `JexlRop`, которая позволяет обращаться к полям объекта.

  Совет: Для получения исходного `rop` воспользуйтесь следующим примером:

  ```
  //в переменной jexlRop находится объект класса JexlRop
  var rop = jexlRop.data()
  ```

  – Если адм. объект создан не по классу, то здесь будет json-представление строки дата-сета выборки.
- `params`
  массив значений параметров, собранных со всех ролей пользователя.
  – если правило примитивное - то будет содержать примитивы (строка или число)
  – если правило составное - то будет содержать строки, каждая из которых - json

Роль

1. Открыть карточку роли или выбрать ее в списке;
2. перейти на закладку `Дискретный доступ`;
3. создать новую запись, выбрав адм. объект;
4. указать значение параметров для ограничений;
5. сопоставить объектные и элементарные привилегии с нужным значением параметров ограничений, заполнив поле `Ограничение дискретного доступа`

Ограничение доступа к объектам класса.

Выдается через объектные привилегии с системными именами `edit#` и `view#`

Совет: Если таких привилегий нет у адм. объекта, то выполните его синхронизацию.

Дискретные ограничения применяются только, если:

- это главная выборка адм. объекта (шапка БО);
- это не супер-пользователь;
- к адм. объекту приминаются настройки администрирования;
- адм. объект имеет признак `Дискретный доступ`.

Определение объектов для проверки дискретных прав

При проверке дискретных привилегий строки, в контексте которых требуется проверить наличие прав, определяются методом `acRows`

Алгоритм работы метода:

1. Если выборка принадлежит классу (`thisApi() != null`), то для всех выделенных записей через поле первичного ключа загружаются `провайдеры строк`, из них через поле `gidRoot_dz` определяются шапки БО
2. Если выборка не принадлежит классу (`thisApi() == null`), то по всем выделенным строкам собирается json-массив, содержащий на каждую строку json-объект, в котором ключ - имя атрибута, значение - значение атрибута.

Ограничение списков

Используя значения параметров всех ролей и настроенные «Скрипты для фильтрации объектных привилегий» формируется условие ограничения и накладывается фильтрация через макрос `&DefUniFltMacros`.

Ограничение открытия карточек

На открытие карточки используя значения параметров всех ролей выполняется `jexl-скрипт`, настроенный в `Скрипт проверки строк по объектному кешу`. Если у пользователя нет прав на объект, то будет выдана ошибка.

Ограничение на редактирование объектов

На `beforeEdit` или `delete` проверяется наличие привилегии `edit#` (выполняется `jexl-скрипт`, настроенный в `Скрипт проверки строк по объектному кешу`). Если у пользователя нет прав на объект, то будет выдана ошибка.

Ограничение на объектные привилегии

Если для объектной привилегии указано `Ограничение дискретного доступа`, то такая привилегия может быть проверена только в контексте какого-либо объекта. Для проверки объектных привилегий используется метод `Btk_AdminPkg().hasObjPriv`.

Пример:

```
if (Btk_AdminPkg().hasObjPriv("SomeObjectName", "SomePrivName", Seq(rop))) {
   //код, выполняемый при наличии привилегии
}
```

Если для привилегии, имеющей ограничение, выполнить проверку без передачи объектов, то будет выдана ошибка.

Ограничение на элементарные привилегии

Для элементарной привилегии может быть указано ограничение дискретного доступа. Если ограничение указано, то при выполнении операции будут проверены права в контексте записей в выборке.

На загрузку выборки или перезагрузку `ClassLoader`-а происходит подписка на событие выполнения операции.

При выполнении операции проверяются дискретные права на эту элементарную привилегию:

1. Получение `acRows`
2. Для полученных строк выполняется `jexl-скрипт`, настроенный в `Скрипт проверки строк по объектному кешу`
3. Если прав нет, то выдается ошибка.

Принудительное включение и отключение проверки дискретного доступа

Принудительное включение или отключение проверки дискретного доступа позволяет игнорировать настройки дискретного доступа на административном объекте. Признак устанавливается через параметр `UseDiscreteAccess` при открытии `lookUp-формы` через метод-обработчик.

Таким образом, установка параметра `UseDiscreteAccess` в значение равное 1 позволит принудительно применить настройки дискретного доступа, независимо от настройки на административном объекте. Пример принудительного включения проверки дискретного доступа:

```
Bs_BisObjAvi.processIdMCEvent(
      event,
      { id => setClientidBisObjFrom(editRef(), id) },
      params = Map(RefEventParam.UseDiscreteAccess -> 1)
)
```

При установка значения параметра `UseDiscreteAccess` в значение равное `0` настройки дискретного доступа будут игнорироваться, вне зависимости от настройки дискретного доступа на административном объекте. Пример принудительного выключения проверки дискретного доступа:

```
Bs_BisObjAvi.processIdMCEvent(
      event,
      { id => setClientidBisObjFrom(editRef(), id) },
      params = Map(RefEventParam.UseDiscreteAccess -> 0)
)
```

Если параметр не был передан при вызове метода-обработчика то проверка дискретного доступа осуществляется по настройке на административном объекте.

Параметр учитывается на всех методах-обработчиках (`processRefEvent`, `processHLEvent`, `processIdMCEvent` и т.д.).

Проверка прав доступа на объект после его редактирования

1. Открыть карточку `Адм. объекта`
2. Установить признак `Дискретный доступ`
3. Установить признак `Проверять изменение прав объектов на сохранении`

При включенном дискретном доступе и проверке изменения прав на сохранении, если при редактировании пользователем объекта, объект становится недоступным даже для чтения, система выдаст ошибку с откатом изменений.

4.2 Подправила

Особенности подправил

Для каждого правила можно создать подправило. Оно будет использовать те же параметры, что и основное правило, но при этом автоматически нигде не проверяется. Программист должен самостоятельно внедрить соответствующую проверку в коде. После этого администратор, зная имя подправила, сможет задать для него нужные ограничения и проверки.

Для администрируемого объекта можно создать подправило, выполнив следующие шаги:

1. Открыть карточку администрируемого объекта.
2. Перейти в закладку Дискретные ограничения доступа.
3. Выбрать существующие правило или создать его.
4. Под выбранным или созданным правилом добавить подправило.
5. Назвать подправило именем, которое используется в коде для проверки доступа.
6. Задать необходимые скрипты для подправила.

После выполнения этих шагов подправило будет создано и применено в соответствии с заданными параметрами.

Пример проверки подправила в выборке:

```
override protected def onRefresh: Recs = {
   lazy val sSubRule = "SomeSubRuleName"

   val ropav = SomeApi().refreshByParent(getIdMaster)

   if (needApplyDiscreteAccess) {
      lazy val bHasGroup = Btk_AcLib().hasBtkGroup(acObject, acObject)
      ropav.filter { rv =>
         val rows = AcRows(Seq(rv))

         val viewPriv = if (bHasGroup) Btk_AdminPkg.viewGroupPriv.get else Btk_AdminPkg.
↪viewObjPriv.get
         val editPriv = if (bHasGroup) Btk_AdminPkg.editGroupPriv.get else Btk_AdminPkg.
↪editObjPriv.get

         Btk_AdminPkg().hasObjPrivByRows(acObject.get, viewPriv, rows, spSubRule =␣
↪sSubRule) ||
                Btk_AdminPkg().hasObjPrivByRows(acObject.get, editPriv, rows, spSubRule␣
↪= sSubRule)
      }.map { rop =>
         Row(rop.idJ, SomeApi().getHeadLine(rop.idJ))
      }
   } else {
      ropav.map { rop =>
         Row(rop.idJ, SomeApi().getHeadLine(rop.idJ))
      }
   }
}
```

4.3 Примеры скриптов

Примитивное правило по атрибуту класса

Правило значимое, тип данных строка. Фильтрация по like атрибута sCode

Скрипт для фильтрации объектных привилегий:

```
select 1
  from RplTst_ClassTree tt
  join (
     select cast(json_array_elements_text(cast((&params) as json)) as varchar) as sCode
      )  as codes
   on tt.sCode like codes.sCode
where tt.id = (&id)
```

Скрипт проверки строк по объектному кешу:

```
for (p: params) {
  if (row.sCode != null && p != null) {
    if (like(row.sCode, p)) {
     return true;
    }
```

(continues on next page)

(продолжение с предыдущей страницы)

```
  }
}
```

Примитивное правило по атрибуту коллекции

Правило ссылочное на `Btk_Group`. Фильтрация по прямому вхождению объекта в группу (фильтрация коллекции `Btk_ObjectGroup`).

Скрипт для фильтрации объектных привилегий:

```
select 1
  from Rpltst_TestGroup tt
  join Btk_ObjectGroup og
    on tt.gid = og.gidSrcObject
  join (
     select cast(json_array_elements_text(cast((&params) as json)) as int8) as id
      )  as params
   on og.idGroup = params.id
where tt.id = (&id)
```

Скрипт проверки строк по объектному кешу:

```
for (p: params) {
  var rops = toJRops(Btk_ObjectGroupApi.byParent(row.data())).asList();
  if (p != null) {
    for (r: rops) {
      if (r.idGroup == p) {
        return true;
      }
    }
  }
}
```

Примитивное правило по адм. объекту, созданному не по классу (произвольная выборка)

Правило значимое, тип данных строка. Фильтрация регистронезависимая по like поля `sClass`

Скрипт для фильтрации объектных привилегий:

```
select 1
   from json_array_elements_text(cast((&params) as json)) as p
  where upper((&sClass)) like upper(p)
```

Скрипт проверки строк по объектному кешу:

```
for (p: params) {
  if (row.sClass != null && p != null) {
    if (like(row.sClass.toUpperCase(), p.toUpperCase())) {
     return true;
    }
  }
}
```

Составное правило по 3 атрибутам класса

Параметры правила:

1. `dDate` - Значимый, тип данных дата.
2. `nNumber` - Значимый, тип данных число
3. `sCaption` - Значимый, тип данных строка

Фильтрация отбирает записи у которых дата меньше указанной, число равно указанному, и Наименование по like регистронезависимо совпадает с указанной

Скрипт для фильтрации объектных привилегий:

```
select 1
  from RplTst_AllDbTypes tt
 where tt.id = (&id)
   and exists (
    select 1
      from (
        select v -> 'dDate' as dDate
              ,v -> 'nNumber' as nNumber
              ,v -> 'sCaption' as sCaption
          from json_array_elements(cast((&params) as json)
            ) as v
        ) p
      where exists (
        select 1
          from json_array_elements_text(p.dDate) pp
          where tt.dDate <= to_timestamp(pp, 'DD.MM.YYYY HH24:MI:SS')
      )
        and exists (
        select 1
          from json_array_elements_text(p.nNumber) pp
          where tt.nNumber = cast(pp as numeric)
      )
        and exists (
        select 1
          from json_array_elements_text(p.sCaption) pp
          where upper(tt.sCaption) like upper(pp)
      )
   )
```

Скрипт проверки строк по объектному кешу:

```
for (p: params) {
  var jObj = toJObject(p);
  var res = true

  //проверка даты
  if (res) {
    res = false
    var aValues = jObj.childJArray("dDate");
    var i = 0
    while(i < aValues.size() && !res) {
```

(continues on next page)

(продолжение с предыдущей страницы)

```
      var value = aValues.getDate(i);
      if (row.dDate <= value) {
        res = true;
      }
      i = i + 1;
    }
  }

  //проверка числа
  if (res) {
    res = false
    var aValues = jObj.childJArray("nNumber");
    var i = 0
    while(i < aValues.size() && !res) {
      var value = aValues.getNumber(i);
      if (row.nNumber == value) {
        res = true;
      }
      i = i + 1;
    }
  }

  //проверка наименования
  if (res) {
    res = false
    var aValues = jObj.childJArray("sCaption");
    var i = 0
    while(i < aValues.size() && !res) {
      var value = aValues.getString(i);
      if (row.sCaption != null && value != null) {
        if (like(row.sCaption.toUpperCase(), value.toUpperCase())) {
          res = true;
        }
      }
      i = i + 1;
    }
  }

  if (res) {
    return true;
  }

}
```

Составное правило по атрибуту класса и коллекции

Параметры правила:

1. `sCaption` - Значимый, тип данных строка
2. `idGroup` - ссылочный на `Btk_Group`

Фильтрация отбирает записи, у которых Наименование по like регистронезависимо совпадает с указанным, и есть прямое вхождение в группу (фильтрация коллекции `Btk_ObjectGroup`).

Скрипт для фильтрации объектных привилегий:

```
select 1
  from RplTst_TestGroup tt
 where tt.id = (&id)
   and exists (
     select 1
     from (
       select v -> 'sCaption' as sCaption
            ,v -> 'idGroup' as idGroup
         from json_array_elements(cast((&params) as json)) as v
       ) p
     where exists (
       select 1
         from json_array_elements_text(p.sCaption) pp
         where upper(tt.sCaption) like upper(pp)
     )
       and exists (
         select 1
           from Btk_ObjectGroup og
           join json_array_elements_text(p.idGroup) pp
             on og.idGroup = cast(pp as int8)
           where og.gidSrcObject = tt.gid
       )
   )
```

Скрипт проверки строк по объектному кешу:

```
for (p: params) {
  var jObj = toJObject(p);
  var res = true

  //проверка наименования
  if (res) {
    res = false
    var aValues = jObj.childJArray("sCaption");
    var i = 0
    while(i < aValues.size() && !res) {
      var value = aValues.getString(i);
      if (row.sCaption != null && value != null) {
        if (like(row.sCaption.toUpperCase(), value.toUpperCase())) {
          res = true;
        }
      }
```

(continues on next page)

(продолжение с предыдущей страницы)

```
      i = i + 1;
    }
  }

  //проверка группы
  if (res) {
    res = false
    var rops = toJRops(Btk_ObjectGroupApi.byParent(row.data())).asList();
    var aValues = jObj.childJArray("idGroup");
    var i = 0
    while(i < aValues.size() && !res) {
      var value = aValues.getLong(i);
      for (r: rops) {
        if (r.idGroup == value) {
          res = true;
        }
      }
      i = i + 1;
    }
  }

  if (res) {
    return true;
  }

}
```

Составное правило по 2 атрибутам адм. объекта, созданного не по классу (произвольная выборка)

Параметры правила:

1. `sClass` - Значимый, тип данных строка
2. `idRefClass` - ссылочный на `Btk_Class`

Фильтрация отбирает записи, у которых поле `sClass` по like регистронезависимо совпадает с указанным, и поле `idRefClass` совпадает с указанным

Скрипт для фильтрации объектных привилегий:

```
select 1
   from (
     select v -> 'sClass' as sClass
          ,v -> 'idRefClass' as idRefClass
       from json_array_elements(cast((&params) as json)) as v
    ) p
  where exists (
      select 1
        from json_array_elements_text(p.sClass) pp
       where upper((&sClass)) like upper(pp)
    )
    and exists (
      select 1
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        from json_array_elements_text(p.idRefClass) pp
       where (&idRefClass) = cast(pp as int8)
    )
```

Скрипт проверки строк по объектному кешу:

```
for (p: params) {
  var jObj = toJObject(p);
  var res = true

  //проверка sClass
  if (res) {
    res = false
    var aValues = jObj.childJArray("sClass");
    var i = 0
    while(i < aValues.size() && !res) {
      var value = aValues.getString(i);
      if (row.sClass != null && value != null) {
        if (like(row.sClass.toUpperCase(), value.toUpperCase())) {
          res = true;
        }
      }
      i = i + 1;
    }
  }

  //проверка idRefClass
  if (res) {
    res = false
    var aValues = jObj.childJArray("idRefClass");
    var i = 0
    while(i < aValues.size() && !res) {
      var value = aValues.getLong(i);
      if (row.idRefClass == value) {
        res = true;
      }
      i = i + 1;
    }
  }

  if (res) {
    return true;
  }

}
```

5 Приложение 1

5.1 Функции отображения

Для работы системы администрирования в отображении объявлены следующие функции:

- `onLoadAdminMeta`
  Применяет настройки администрирования;
- `acObject`
  Возвращает имя администрируемого объекта;
- `acObjectBundle`
  Имя узла администрирования;
- `acObjectItem`
  Имя элемента администрирования;
- `isAdministraded`
  Признак, что выборка администрируется.

5.2 Функции проверки привилегий

- `ru.bitec.app.btk.Btk_AdminPkg#hasObjPriv`
  Проверяет наличие объектной привилегии у пользователя
- `ru.bitec.app.btk.Btk_AdminPkg#hasStateChangePriv`
  Проверяет наличе привилегии на перевод состояния
- `ru.bitec.app.btk.Btk_AdminPkg#hasRole`
  Проверяет наличие у пользователя роли

6 Связь выборок с администрируемыми объектами

6.1 основные понятия

Элемент администрирования

Элемент администрирования (класс `Btk_AcItem`) - логический объект, который соответствует одной выборке.

На каждую выборку, обладающую своей `avm.xml`, создается отдельный `элемент администрирования`, обладает перечнем `элементарных привилегий`

Элементарная привилегия

Элементарная привилегия (класс Btk_AcItemPrivilege, коллекция к Btk_AcItem) - логический объект, который соответствует одному из элементов выборки: атрибуту или операции.

Элементарные привилегии обладают типом привилегии. Например, атрибуты относятся к типу Чтение.

Тип элементарной привилегии

Определяет группу Элементарной привилегии. Позволяет управлять доступом к группам привилегий, а не по отдельности к каждой элементарной привилегии

Объект администрирования

Объект администрирования (класс Btk_AcObject) - логический объект, который соответствует Бизнес объекту, и обладает перечнем элементов администрирования, входящих в этот объект.

Бизнес объект

Бизнес объект (класс Btk_BoEntity) - логическое объединение нескольких классов в единую сущность.

Бизнес объект создается на корневые классы, и включает все его коллекции.

Базовая выборка

Выборка, которая соответствует классу. Например, выборка Btk_ClassAvi является базовой выборкой класса Btk_Class

Произвольная выборка

Выборка, которая не принадлежит какому-либо классу. В том числе и выборки-наследники от базовой выборки

Корневой класс

Класс, на который формируется отдельный бизнес объект. Является логическим отдельным объектом, который обладает своими коллекциями.

Это классы с типом Справочник, Документ, Настройка, Журнал.

6.2 Принцип построения адм. объектов

Формирование на основе структуры odm.xml

1. На каждый корневой класс создается свой администрируемый объект, имя которого равно имени класса.
2. В администрируемый объект включаются элементы администрирования, которые соответствуют выборкам всех классов входящий в бизнес объект (сам класс и все его коллекции).

Пример структуры бизнес объекта:

```
Документ Some_Document
  + Коллекция Some_DocumentPosition
    + Коллекция Some_DocumentPositionDet
  + V-коллекция Коллекция Some_DocumentLink
```

Пример сформированного администрируемого объекта:

```
Адм. объект Some_Document
  + Элемент Some_DocumentAvi
  + Элемент Some_Document\Some_DocumentPositionAvi
  + Элемент Some_Document\Some_DocumentPosition\Some_DocumentPositionDetAvi
  + Элемент Some_Document\Some_DocumentLinkAvi
```

Элементы администрирования в составе объекта администрирования имеют имя, которое соответствует их пути внутри бизнес объекта. Зачем используются такие пути будет описано в главе Определение связи выборки и адм. объекта

Подключение произвольной выборки в структуру адм. объекта

По мимо элементов администрирования, соответствующих базовым выборкам классов бизнес объектов, есть возможность добавить произвольную выборку в структуру администрируемого объекта. Для этого существует 2 способа:

1. Подключение в исходном коде
   В avm.xml базовой выборки корневого класса добавить тег acObject, в котором указать дополнительные элементы администрирования.

   Пример разметки:

   ```
   <view xmlns="http://www.global-system.ru/xsd/global3-view-1.0"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.global-system.ru/xsd/global3-view-1.0"
      name="Some_Document">

    <acObject>
        <acItems>
            <acItem name="Some_Document\Some_CustomAvi" caption="Произвольная выборка"/
   ↪>
        </acItems>
    </acObject>
   </view>
   ```

   В этом примере объявлен один дополнительный элемент администрирования в составе объекта «Some_Document», с именем «Some_Document\Some_CustomAvi» и наименованием «Произвольная выборка». Наименование используется для визуального отображения этого элемента в приложении администратор.

2. Подключение через приложение администратор
   - Открыть карточку нужного объекта администрирования
   - добавить новый элемент на закладке Элементы администрируемого объекта

6.3 Определение связи выборки и адм. объекта

Каждая выборка на открытии в событии `onLoadAdminMeta` через методы выборки `acObject` и `acObjectItem` определяет свою принадлежность к `элементу администрирования` и `объекту администрирования`. Все настройки прав доступа к выборке будут искаться по паре значений, которые вернули эти два метода.

Значение, которые вернули эти два метода для выборки, можно увидеть через `окно отладки`.

Т.е. чтобы к выборке корректно применялись настройки доступа, в приложении `администратор` должен быть `объект администрирования`, имя которого равно результату метода `acObject`, и в составе этого объекта должен быть `элемент администрирования`, имя которого равно результату метода `acObjectItem`.

Метод acObject

Возвращает системное имя `объекта администрирования` к которому относится выборка. Например, «Some_Document»

Основной принцип:

1. Если выборка принадлежит классу, и класс `корневой`, то имя этого класса = имя адм. объекта
2. Иначе вызывается метод `acObject` мастер-выборки.

Метод acObjectItem

Возвращает системное имя `элемента администрирования` в составе объекта. например, «Some_Document\Some_DocumentPositionAvi»

Основной принцип:

1. Вызывается метод `acObjectItem` мастер-выборки.
2. К полученному значению добавляется текущее имя выборки.

Базовые выборки классов

Для `базовых выборок классов` в большинстве случаев корректно отрабатывает стандартная логика определения `элемента администрирования` и `объекта администрирования`.

Исключением может служить `базовая выборка v-коллекций`, если она открывается вне формы, на которой расположена `базовая выборка мастер-объекта`. Например, если открыть список объектов `v-коллекции`, как главную выборку формы.

Произвольные выборки

Для `произвольных выборок` определить принадлежность к `объекту администрирования` автоматически не так просто, как для `базовых выборок`, которые соответствуют структуре `бизнес объектов`.

В большинстве случаев разработчику требуется самостоятельно переопределить работу методов `acObject` и `acObjectItem`.

Выборка как самостоятельный адм. объект

Если произвольную выборку предполагается использовать как самостоятельный объект администрирования, то требуется:

1. в avm.xml объявить тег acObject
2. в Avi переопределить методы acObject и acObjectItem следующим образом:

```
override def acObject: NString = {
  baseAvi.simpleName
}

override def acObjectItem: NString = {
  baseAvi.simpleName
}
```

Выборка как часть другого адм. объекта

Если произвольную выборку предполагается использовать как часть существующего объекта администрирования, то требуется:

1. В avm.xml базовой выборки корневого класса этого бизнес объекта объявить тег acObject и внутри него добавить новый элемент администрирования

```
<view xmlns="http://www.global-system.ru/xsd/global3-view-1.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.global-system.ru/xsd/global3-view-1.0"
   name="Some_Document">

 <acObject>
     <acItems>
         <acItem name="Some_Document\Some_CustomAvi" caption="Произвольная выборка"/
↪>
     </acItems>
 </acObject>
</view>
```

2. в Avi произвольной выборки переопределить методы acObject и acObjectItem следующим образом:

```
override def acObject: NString = {
  //имя объекта
  "Some_Document"
}

override def acObjectItem: NString = {
  //путь, который мы указали в теге acItem
  "Some_Document\\Some_CustomAvi"
}
```

3. в Api произвольной выборки зарегистрировать с помощью метода Btk_AcObjectItemApi().registerByAcObjectItemName

```
def regAcObjectItem(): Unit = {
val bvDefinedAcObjectItem = TxIndex(Btk_AcObjectItemAta.Type)(_.sCode).byKey("Some_
↪Document\\Some_Document\\Some_CustomAvi#Default").isEmpty
if (bvDefinedAcObjectItem) Btk_AcObjectItemApi().registerByAcObjectItemName("Some_
↪Document\\Some_Document\\Some_CustomAvi#Default")
}
```

7 Телеметрия

7.1 Основные понятия

Отправка телеметрической информации во внешние аналитические системы по стандарту OpenTelemetry.

Телеметрия - удобный процесс сбора информации о состоянии приложения.

Смотрите также:

- Документация OpenTelemetry
- Сигналы

Сигналы

Трассировка

Детальный отчёт о действиях происходящих на сервере при обработке входящих запросов.

Метрики

Периодически измеряемый количественный показатель.

Пример:

- количество http запросов к серверу по нарастанию
- процент использование процессора в минуту

Счетчик

Позволяет получать информацию о количестве случившихся событий.

Спидометр

Позволяет получать информацию о текущей загрузке ресурса.

Административный объект

Сигналы собирающиеся в разрезе административного объекта могут быть включены или выключены для конкретного административного объекта

Объектная метрика

Используется в случаи когда необходимо измерить специфичный показатель через сигнал метрики. В таком случаи прикладной программист регистрирует объектную метрику в метаданных модуля.

Объектная трассировка

Используется в случаи когда необходимо измерить специфичный показатель через сигнал трассировки. В таком случаи прикладной программист регистрирует объектную трассировку в метаданных модуля.

Серверная телеметрия

Отслеживает общее время взаимодействие между тонким клиентом и сервером приложения.

7.2 Настройка телеметрии

Для настройки телеметрии откройте форму `Администратор: Настройки->Телеметрия`

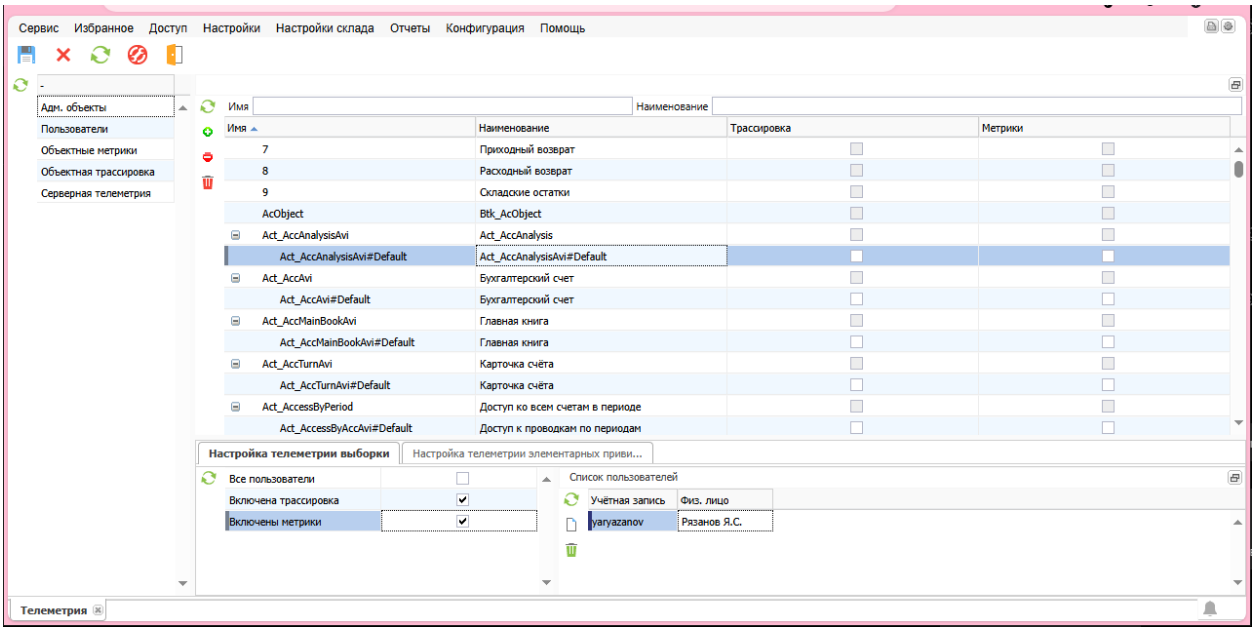
Разделы

Адм. объекты

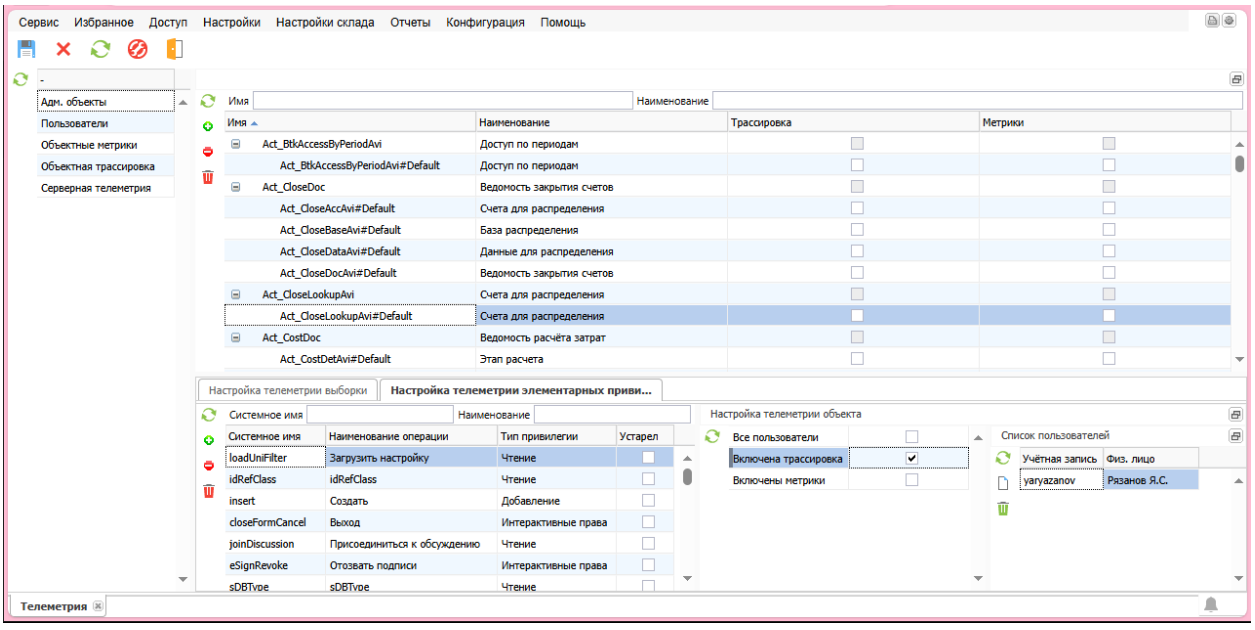
Позволяет включать сбор *сигналов* по административным
объектам в разрезе всех или конкретных пользователей.

Примеры

Включение метрик и трассировки на административном объекте `Act_AccAnalysisAvi` для пользователя `yaryazanov`.



Включение трассировки элементарной привилегии `loadUniFilter` на административном объекте `Act_CloseLookupAvi` для пользователя `yaryazanov`.

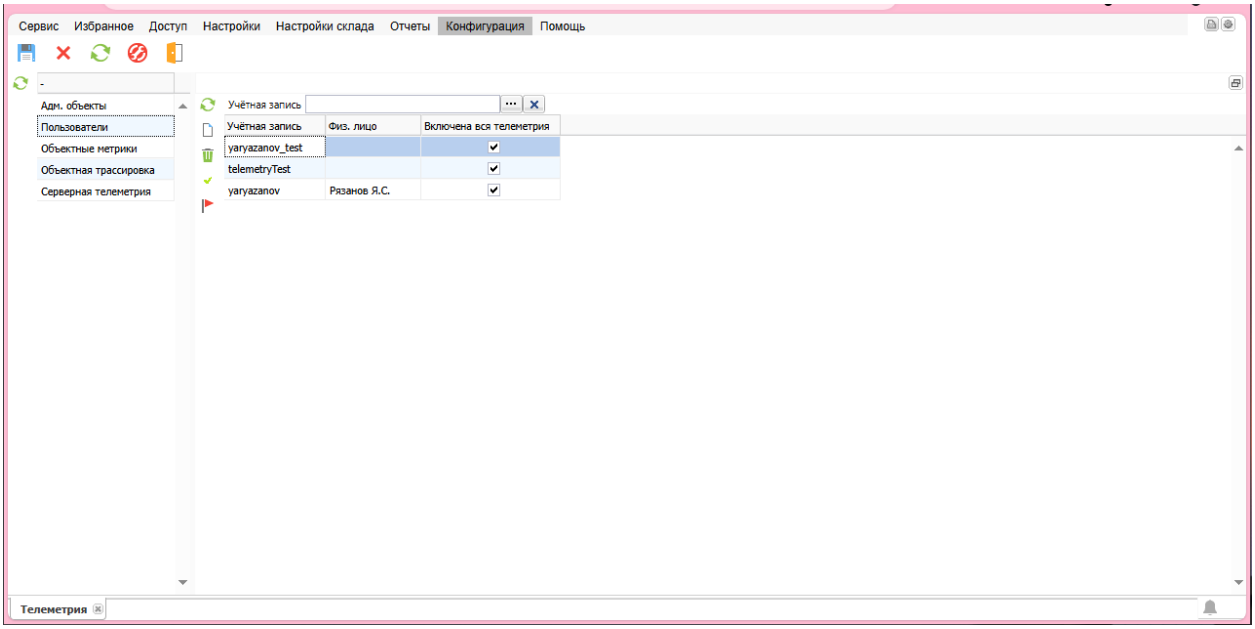


Пользователи

Позволяет включать сбор *сигналов* по всем адм. объектам на конкретном пользователе

Примеры

Включение сбора *сигналов* по всем адм. объектам для пользователей `yaryazanov`, `telemetryTest` и `yaryazanov_test`.



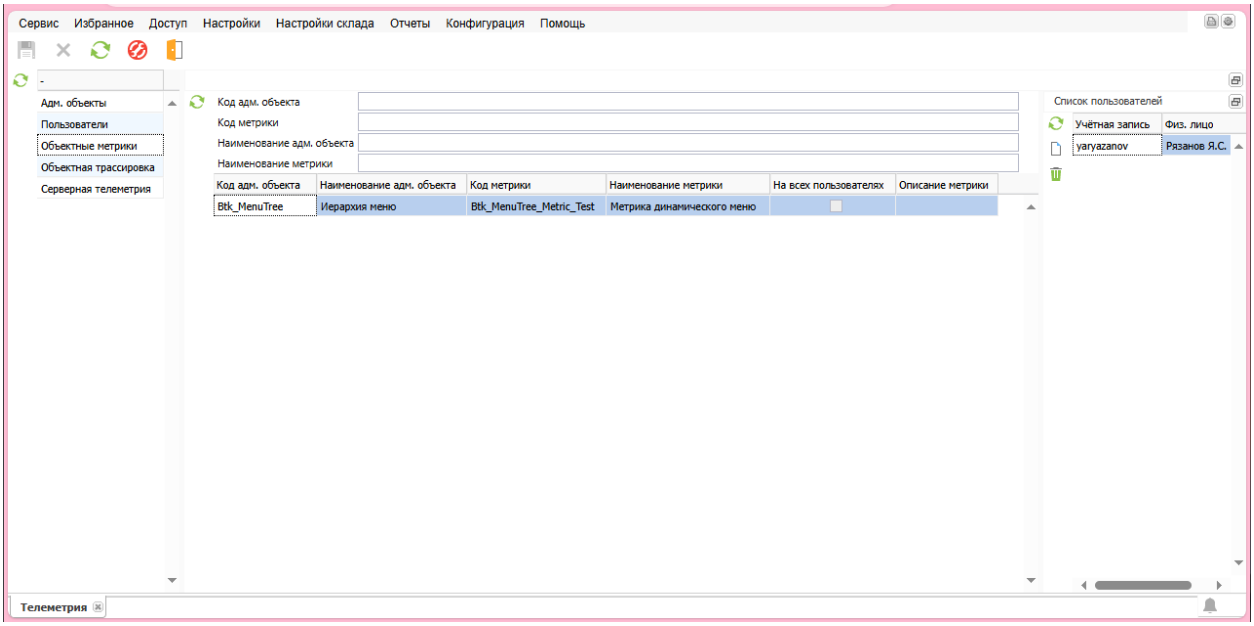
Объектные метрики

Данный раздел позволяет увидеть общий список объектных метрик, а так же настроить их сбор для всех или конкретных пользователей.

> **Внимание:** Регистрацией объектных метрик занимается разработчик в приклодном коде.

Примеры

Включение сбора объектной метрики `Btk_MenuTree` для пользователя `yaryazanov`.



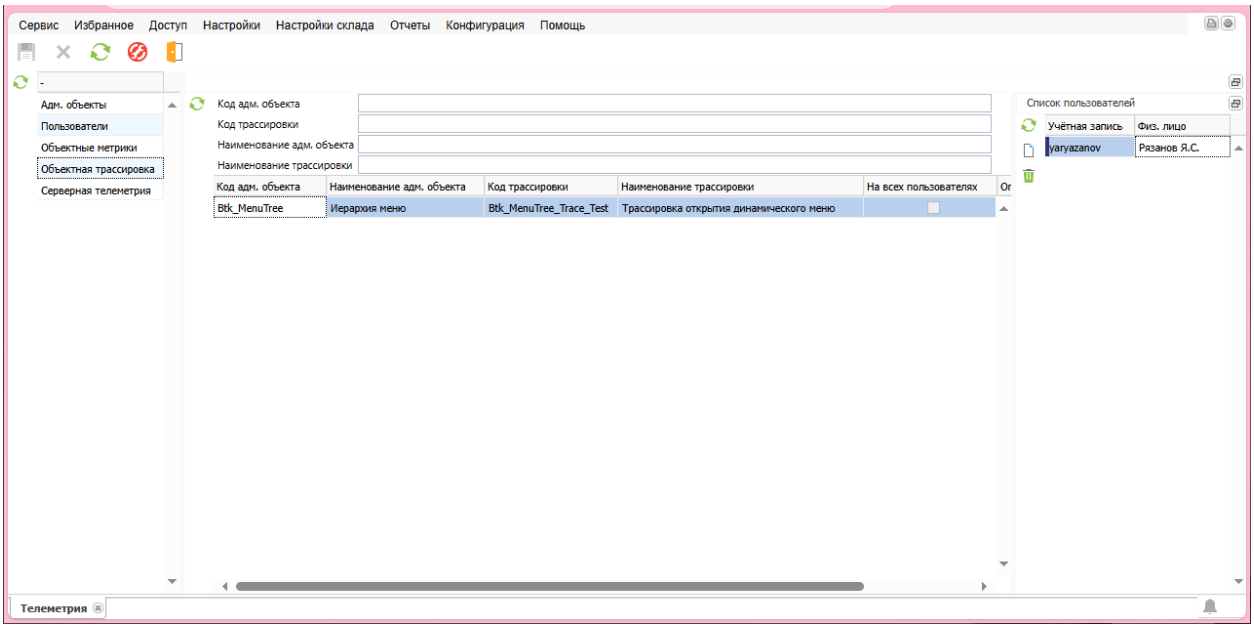
Объектная трассировка

Данный раздел позволяет увидеть общий список объектных трассировок, а так же настроить их сбор для всех или конкретных пользователей.

> **Внимание:** Регистрацией объектных трассировок занимается разработчик в приклодном коде.

Примеры

Включение сбора объектной трассировки `Btk_MenuTree` для пользователя `yaryazanov`.



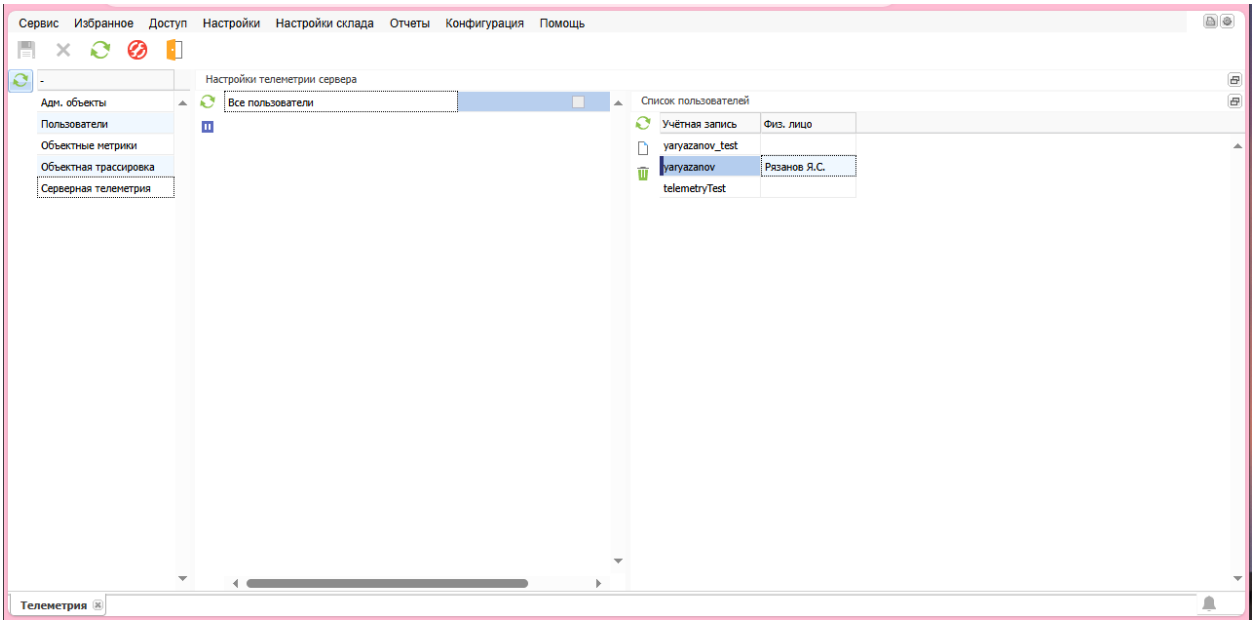
Серверная телеметрия

Данный раздел позволяет настроить серверную телеметрию всех или конкретных пользователей.

> **Внимание:** В данном разделе можно отключить всю телеметрию полностью.

Примеры

Включение сбора *сигналов* серверной телеметрии для пользователей `yaryazanov`, `telemetryTest` и `yaryazanov_test`.



7.3 Объектные метрики и трассировка

У программистов есть возможность создавать свои метрики и трассировку по администрируемым объектам.

Что бы создать метрику, необходимо в классе «Метрики объектов» зарегистрировать запись:

```
val ropAcMetric = Btk_AcObjectMetricApi().register(
  idParent = Btk_AcObjectApi().findByMnemoCode("Btk_MenuTree"),
  sCode = "Btk_MenuTree_Metric_Test",
  sCaption = "Метрика динамического меню"
)
```

> **Внимание:** Код метрического прибора или трассировки должен быть не более 255 символов, должен начинаться с буквы и может содержать только следующие специальные символы: „_“, „.“, „-“.

После регистрации администратор в настройках телеметрии во вкладке «Объектные метрики» может включить её для всех пользователей или определённого набора.

Для использования метрики в прикладном коде, необходимо получить метрику и увеличить счётчик.

```
val longCounterOpt = Btk_TelemetryPkg().getAcObjectLongCounter(
  idUser = Btk_UserApi().getCurrentUserID(),
  idAcObjectMetric = ropAcMetric.idj
)

longCounterOpt.foreach(_.add(1))
```

Внимание: Методы получения метрических приборов или трассировки возвращают монаду Option. В контейнере значение присутствует, только если на текущем пользователи включены настройки.

Для использования трассировки аналогичные методы, только регистрация происходит в классе Btk_AcObjectTrace.

7.4 Справка

Метрика

Имена

- «btk_item_telemetry_user_operation_exec_time» - Гистограмма времени выполнения пользовательской операции.
- «btk_item_telemetry_user_operation_calls_successful_total» - Счётчик количества успешных вызовов пользовательской операции.
- «btk_item_telemetry_user_operation_calls_failed» - Счётчик количества вызовов пользовательской операции, которые завершились ошибкой.
- «btk_item_telemetry_user_operation_calls_total» - Счётчик количества вызовов пользовательской операции.

Трассировка

Имена

- «btk_item_telemetry_user_operation_trace» - Трассировка пользовательских операций.

Атрибуты

Каждая метрика содержит в себе перечень атрибутов по которым можно фильтровать значения в мониторе.

- «solution» - Алиас БД
- «solution.image» - Алиас SBT
- «userName» - Логин пользователя
- «session.secret» - «Секретный» ключ рабочего сеанса
- «session.sid» - Идентификатор сессии
- «work.session.sid» - Идентификатор рабочей сессии
- «cluster.node» - Наименование кластерной ноды
- «selection.name» - Системное наименование выборки
- «selection.caption» - Наименование выборки
- «oper.name» - Системное наименование операции
- «oper.caption» - Наименование операции
- «representation.name» - Системное наименование отображения
- «form.name» - Системное наименование формы
- «form.caption» - Наименование формы
- «form.representation.name» - Системное наименование отображения формы

Серверные метрики

Имена

- «rpc_client_duration» - Длительность выполнения RPC с точки зрения клиента.
- «rpc_server_duration» - Длительность обработки RPC сервером.
- «rpc_transport_duration» - Длительность передачи по сети (Разница между «rpc_duration_client_ms» и «rpc_duration_server_ms»).