ООО «Газинформсервис»

УТВЕРЖДАЮ

_____

«____» _____ 2022 г.

ПРОГРАММНЫЙ КОМПЛЕКС ANKEY IDENTITY MANAGER

Руководство администратора

ЛИСТ УТВЕРЖДЕНИЯ

72410666.00054-03 96 01-ЛУ

Представители предприятия-разработчика:

_____

Исполнитель

_____

Нормоконтролер

_____

2022

Изм.	Подп.	Дата

Литера

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл.	

72410666.00054-03 96 01

ООО «Газинформсервис»

УТВЕРЖДЕН
72410666.00054-03 96 01-ЛУ

ПРОГРАММНЫЙ КОМПЛЕКС ANKEY IDENTITY MANAGER

Руководство администратора

72410666.00054-03 96 01

Листов 254

2022

Изм.	Подп.	Дата

Литера

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл.	

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Область применения

Наименование программы: программный комплекс Ankey Identity Manager (IDM) (далее – Комплекс).

Обозначение программы: 72410666.00080-01.

Комплекс предназначен для автоматизации процессов централизованного управления учетными записями (УЗ) и привилегиями пользователей в целевых системах (ЦС).

Комплекс функционирует с использованием следующего программного обеспечения:

- операционные системы (ОС) семейства Linux;
- система управления базами данных (СУБД) PostgreSQL;
- комплекты разработчика приложений на языке Java (Java Development Kit (JDK)) Oracle JDK или OpenJDK, которые включают виртуальную машину Java JDK 8 или Java JDK 11;
- служба поиска OpenSearch.

Комплекс состоит из следующих компонентов:

- сервер приложений Ankey IDM – может устанавливаться вместе или раздельно от СУБД;
- сервер СУБД – может устанавливаться вместе или раздельно от сервера приложений;
- сервер коннекторов – обеспечивает выполнение функций по интеграции с ЦС средствами коннекторов. Может устанавливаться совместно с сервером приложений или на выделенном сервере.

1.2. Основные функции администратора Комплекса

Администратор Комплекса выполняет следующие основные функции:

- установка программных компонент;
- установка коннекторов;

- управление учетными записями ресурсов (УЗР);
- управление периодическими заданиями;
- настройка справочников;
- настройка рабочих потоков;
- настройка дополнительных атрибутов;
- настройка безопасности;
- настройка журналов работы;
- настройка аудита событий.

Действия администратора Комплекса по установке программных компонент описаны в документе «Руководство по инсталляции» (72410666.00080-01 94 01).

Порядок выполнения остальных функций администратора Комплекса приведен в данном документе.

1.3. Общий порядок администрирования

Комплекс позволяет администратору выполнять настройки с помощью:

1) Rest-запросов для управления объектами Комплекса. Описание поддерживаемых Rest-запросов приведено в Приложении А.
2) Консоли администрирования.

Для работы в консоли администрирования требуется web-браузер Internet Explorer версии 11 и выше.

Язык интерфейса консоли администрирования зависит от установленного языка web-браузера. Консоль администрирования поддерживает следующие языки:

- русский;
- английский.

Чтобы войти в консоль администрирования, необходимо выполнить следующие действия:

1) Перейти по следующей ссылке в web-браузере: http://<адрес сервера>:<порт>. Пример: http://localhost :8080/.
2) Пройти аутентификацию, указав идентификатор (логин) и пароль пользователя с правами администратора. При первоначальной

установке по умолчанию создается служебный пользователь с идентификатором – ankey, паролем – ankey.

2. НАСТРОЙКА ПРОГРАММНОГО КОМПЛЕКСА

2.1. Настройки запуска приложения

При загрузке Комплекс использует три следующих файла, которые загружаются в указанном порядке, а значения устанавливаются в обратном порядке:

- system.properties;
- config.properties;
- boot.properties.

Если значение для одного и того же параметра установлено в system.properties и boot.properties, то будет использовано значение из boot.properties.

В файле boot.properties, расположенном в директории ankey/conf/boot, указываются порты для соединения по протоколам http/https и порт для взаимной проверки подлинности с помощью сертификата.

2.2. Установка коннекторов

Коннекторы к ЦС установливаются одним из двух способов:

1) Локально в каталог сервера приложений Ankey IDM в папку «connectors».
2) Удаленно на сервер коннекторов.

Служба сервера коннекторов необходима для запуска исполняемого кода коннекторов к ЦС в следующих случаях:

1) Коннектор должен запускаться в окружении, не поддерживаемом сервером приложений Ankey IDM, например, в ОС Windows.
2) Программные библиотеки, используемые коннектором, могут вызывать конфликт с библиотеками, используемыми сервером приложений Ankey IDM.

Установку коннекторов к ЦС администратору необходимо производить, руководствуясь документацией к коннектору, в следующей последовательности:

1) Размещение библиотек коннектора в папке «connectors» сервера приложений Ankey IDM или удаленно на сервере коннекторов.

2) Размещение внешних библиотек, используемых коннектором, в папке «lib» сервера приложений Ankey IDM или удаленно на сервере коннекторов.
3) Настройка типовых объектов коннектора в Ankey IDM:
   – объект «Коннектор»;
   – объект «Тип ресурса»;
   – объект «Ресурс»;
   – объект «Форма ресурса»;
   – объект «Маппинг»;
   – объект «Учетная запись ресурса» согласно подразделу 2.3;
   – периодические задания согласно подразделу 2.4.

Список установленных коннекторов доступен в меню «Админ. системы»/«Коннекторы». Поиск в списке коннекторов осуществляется по атрибутам: «Название», «Описание». В случае, если Комплекс не отображает результат поиска, следует запустить периодическое задание «reindextask».

2.2.1. Настройка объекта «Коннектор»

Объект «Коннектор» («connector») содержит настройки схемы объектов ЦС, а также другие параметры для интеграции с ней. Параметры объекта «Коннектор»:

1) «connectorName» – наименование коннектора (например, ADConnector).
2) «connectorDesc» – описание коннектора (например, ADConnector).
3) «statusAttr» – настройка атрибута коннектора, значение которого изменяет статус объекта в ЦС (блокирует, разблокирует). Настройка включает указание следующих параметров:
   – «account» – объект коннектора. В качестве объекта коннектора поддерживается только значение «account»;
   – «name» – имя атрибута;
   – «enableValue» – значение для операции разблокировать;
   – «disableValue» – значение для операции заблокировать.

Пример настройки «statusAttr»:

```
{ "account": {
 "name": "enable",
 "enableValue": true,
 "disableValue": false } }
```

4) «connectorBody» – настройки объектов коннектора.
5) «connectorBody/name» – внутреннее наименование коннектора (например, ADConnector).
6) «connectorBody/connectorRef» – сведения о версии и библиотеке коннектора соответствуют содержимому файла MANIFEST.MF внутри архива «jar» библиотеки коннектора. Включает следующие обязательные параметры:
   – «bundleName» – соответствует параметру «ConnectorBundle-Name» файла MANIFEST.MF;
   – «productName» – название продукта, используется для активации лицензии на коннектор. Должен совпадать с названием продукта в системе лицензирования;
   – «bundleVersion» – соответствует параметру «ConnectorBundle-Version» файла MANIFEST.MF;
   – «connectorName» – класс Java, в котором реализован коннектор.

   Пример настройки «connectorBody/connectorRef»:

```
"connectorRef": {
"bundleName": "com.gis.openicf.sapbo",
"productName": "Ankey IDM connector SAP BO",
"bundleVersion": "[0.0,2.0)",
"connectorName": "com.gis.openicf.sapbo.SAPBOConnector"
  }
```

7) «connectorBody/objectTypes» – настройки типов объектов поддерживаемых коннектором (например, account, group, role и т.д.). Наименования могут задаваться любые. Пример настройки «connectorBody/objectTypes»:

```
"objectTypes": {
"account": {...},
```

```
"group": {...}
}
```

8) «connectorBody/objectTypes/». Настройки типа объекта включают следующие параметры:
   - id – идентификатор объекта OpenICF. По умолчанию коннектор использует идентификатор «**ACCOUNT**». Можно вводить свои идентификаторы, при наличии их поддержки в коде коннектора;
   - type – тип объекта, по умолчанию «object»;
   - $schema – схема по которой проводится валидация JSON. По умолчанию «http://json-schema.org/draft-03/schema»;
   - nativeType – тип объекта внутри коннектора. По умолчанию коннектор использует тип «**ACCOUNT**». Можно вводить свои типы, при наличии их поддержки в коде коннектора. При использовании нескольких объектов коннектора типы объектов должны быть уникальны;
   - properties – перечень (атрибутов) параметров объекта.

   Пример настройки «connectorBody/objectTypes/»:

```
"objectTypes": {
"account": {
"id": "__ACCOUNT__",
"type": "object",
"$schema": "http://json-schema.org/draft-03/schema",
"nativeType": "__ACCOUNT__",
"properties": { ....}
}.....
}
```

9) «connectorBody/objectTypes//properties» – содержит перечисление настроек поддерживаемых коннектором атрибутов в следующем формате:

```
{
"<Имя атрибута 1>": {
"type": "<Тип данных>",
"required": <Флаг обязательности (true/false)>,
"nativeName": "<Имя атрибута в целевой системе>",
"nativeType": "<Тип данных в целевой системе>",
"flags": [ "<Дополнительные флаги>"]
},
```

"<Имя атрибута 2>": {..... },
.....}

Атрибут «type» может принимать следующие значения:

– string;
– object;
– array;
– integer;
– boolean.

Атрибут «nativeType» может принимать следующие значения:

– string;
– object;
– array;
– integer;
– boolean;
– JAVA_TYPE_BIGDECIMAL;
– JAVA_TYPE_BIGINTEGER;
– JAVA_TYPE_BYTE;
– JAVA_TYPE_BYTE_ARRAY;
– JAVA_TYPE_CHAR;
– JAVA_TYPE_CHARACTER;
– JAVA_TYPE_DATE;
– JAVA_TYPE_DOUBLE;
– JAVA_TYPE_FILE;
– JAVA_TYPE_FLOAT;
– JAVA_TYPE_GUARDEDBYTEARRAY;
– JAVA_TYPE_GUARDEDSTRING;
– JAVA_TYPE_INT;
– JAVA_TYPE_INTEGER;
– JAVA_TYPE_LONG;
– JAVA_TYPE_OBJECT;
– JAVA_TYPE_PRIMITIVE_BOOLEAN;

– JAVA_TYPE_PRIMITIVE_BYTE;
– JAVA_TYPE_PRIMITIVE_DOUBLE;
– JAVA_TYPE_PRIMITIVE_FLOAT;
– JAVA_TYPE_PRIMITIVE_LONG;
– JAVA_TYPE_STRING.

Атрибут «flags» – дополнительные флаги предоставляют возможность задавать особый порядок обработки атрибута. Может принимать значения:

– NOT_CREATABLE – атрибут не может задаваться Ankey IDM при создании объекта в ЦС (операция «CREATE»);
– NOT_UPDATEABLE – атрибут не может меняться Ankey IDM при обновлении объекта в ЦС (операция «UPDATE»);
– NOT_READABLE – атрибут не может быть прочитан Ankey IDM при синхронизации из ЦС, например поле с паролем;
– NOT_RETURNED_BY_DEFAULT – атрибут ЦС не будет прочтен по умолчанию, а только по требованию (отдельным вызовом, если реализовано в коннекторе), например если атрибут содержит большое значение.

Пример настроек поддерживаемых коннектором атрибутов:

```
{
"userPrincipalName": {
"type": "string",
"required": false,
"nativeName": "userPrincipalName",
"nativeType": "string"
},
"Password": {
"type": "string",
"required": false,
"nativeName": "__Password__",
"nativeType": "JAVA_TYPE_GUARDEDSTRING",
"flags": [
"NOT_READABLE",
"NOT_RETURNED_BY_DEFAULT"
]
},
"cn": {
"type": "string",
"required": true,
```

"nativeName": "cn",
"nativeType": "string",
"flags": [
"NOT_UPDATEABLE"
]
},
"__NAME__": {
"type": "string",
"required": true,
"nativeName": "__NAME__",
"nativeType": "string"
},
"objectGUID": {
"type": "string",
"required": false,
"nativeName": "objectGUID",
"nativeType": "JAVA_TYPE_BYTE_ARRAY",
"flags": [
"NOT_CREATABLE",
"NOT_UPDATEABLE"
]
}}

10) «connectorBody/operationOptions» – настройка специальной обработки для каждой операции коннектора. Например, можно настроить запрет изменений типа объекта. Включает параметры:
   - denied – включение запрета на выполнение операции (пример, true);
   - onDeny – действие, выполняемое в случае, когда параметр denied включен. По умолчанию принимает значение «DO_NOTHING», при котором ничего к запрету действия дополнительно не выполняется. Также может принимать значение THROW_EXCEPTION, при котором в журнал работы будет выведена ошибка.

   Пример настройки «connectorBody/operationOptions»:

"operationOptions" : {
{
"SYNC" :
{
"denied" : true,
"onDeny" : "DO_NOTHING"
}
  }
}

11) «connectorBody/operationTimeout» – настройка времени ожидания завершения для отдельных операций коннектора в миллисекундах. В

случае значения «-1» ожидание не ограниченно. Пример настройки «operationTimeout»:

```
"operationTimeout": {
"GET": -1,
"SYNC": -1,
"TEST": -1,
"CREATE": -1,
"DELETE": -1,
"SCHEMA": -1,
"SEARCH": -1,
"UPDATE": -1,
"VALIDATE": -1,
"AUTHENTICATE": -1,
"SCRIPT_ON_RESOURCE": -1,
"SCRIPT_ON_CONNECTOR": -1 }
```

12) «connectorBody/poolConfigOption» – настройка пула соединений включает следующие параметры:
   - maxIdle – максимальное количество простаивающих экземпляров коннектора (по умолчанию 1);
   - maxWait – максимальное время ожидания получения данных объекта коннектора из ЦС в миллисекундах (по умолчанию 150000);
   - minIdle – минимальное количество простаивающих экземпляров коннектора (по умолчанию 1);
   - maxObjects – максимальное количество простаивающих и активных экземпляров коннектора (по умолчанию 1);
   - minEvictableIdleTimeMillis – минимальное время, которое объект может быть простаивающим в пуле до его удаления в миллисекундах (по умолчанию 120000).

   Пример настройки «connectorBody/poolConfigOption»:

```
"poolConfigOption": {
"maxIdle": 1,
"maxWait": 150000,
"minIdle": 1,
"maxObjects": 1,
"minEvictableIdleTimeMillis": 120000
}
```

13) «connectorBody/producerBufferSize» – размер буфера, по умолчанию принимает значение 100 (пример, «producerBufferSize»: 100).

14) «connectorBody/syncFailureHandler» – параметры повтора неверно завершившихся операций коннектора. Включает следующие параметры:
   – maxRetries – максимальное число повторов;
   – postRetryAction – действие в случае завершения повторов операций (например, запись в лог).

   Пример настройки «connectorBody/syncFailureHandler»:

```
"syncFailureHandler": {
"maxRetries": 5,
"postRetryAction": "logged-ignore"
}
```

15) «connectorBody/connectorPoolingSupported» – включение поддержки пула соединений коннектором. (Например, «connectorPoolingSupported»: true).
16) «connectorBody/enableCaseInsensitiveFilter» – включение поиска объектов в коннекторе без учета регистра. Если параметр выключен, то сравнение идентификаторов на стороне Комплекса и в ЦС ведется с учетом регистра. (Пример «enableCaseInsensitiveFilter»: true).

Пример использования Rest–запроса для создания коннектора:

```
 {
"connectorName": "ADConnector",
"statusAttr": {
   "account": {
      "name": "enable",
      "enableValue": true,
      "disableValue": false
   }
},
"connectorDesc": "ADConnector",
"connectorBody": {
   "name": "ADConnector",
   "objectTypes": {
      "account": {
         "id": "__ACCOUNT__",
         "type": "object",
         "$schema": "http://json-schema.org/draft-03/schema",
         "nativeType": "__ACCOUNT__",
         "properties": {
            "userPrincipalName": {
               "type": "string",
               "required": false,
```

```
            "nativeName": "userPrincipalName",
            "nativeType": "string"
         },
         "Password": {
            "type": "string",
            "required": false,
            "nativeName": "__Password__",
            "nativeType": "JAVA_TYPE_GUARDEDSTRING",
            "flags": [
               "NOT_READABLE",
               "NOT_RETURNED_BY_DEFAULT"
            ]
         },
         "cn": {
            "type": "string",
            "required": true,
            "nativeName": "cn",
            "nativeType": "string",
            "flags": [
               "NOT_UPDATEABLE"
            ]
         },
         "__NAME__": {
            "type": "string",
            "required": true,
            "nativeName": "__NAME__",
            "nativeType": "string"
         },
         "memberOf": {
            "type": "array",
            "items": {
               "type": "string",
               "nativeType": "string"
            },
            "nativeName": "memberOf",
            "nativeType": "string"
         },
         "whenChanged": {
            "type": "string",
            "required": false,
            "nativeName": "whenChanged",
            "nativeType": "JAVA_TYPE_DATE",
            "flags": [
               "NOT_CREATABLE",
               "NOT_UPDATEABLE"
            ]
         }
      }
   },
   "group": {
      "$schema": "http://json-schema.org/draft-03/schema",
      "id": "__GROUP__",
      "type": "object",
      "nativeType": "__GROUP__",
      "properties": {
         "dn": {
            "type": "string",
```

```
            "required": false,
            "nativeName": "distinguishedName",
            "nativeType": "string"
          },
          "objectGUID": {
            "type": "string",
            "required": false,
            "nativeName": "objectGUID",
            "nativeType": "JAVA_TYPE_BYTE_ARRAY"
          }
        }
      }
    },
    "connectorRef": {
      "bundleName": "com.gis.openicf.ad",
      "bundleVersion": "1.1-SNAPSHOT",
      "connectorName": "com.gis.openicf.ad.ADConnector"
    },
    "operationOptions": {},
    "operationTimeout": {
      "GET": -1,
      "SYNC": -1,
      "TEST": -1,
      "CREATE": -1,
      "DELETE": -1,
      "SCHEMA": -1,
      "SEARCH": -1,
      "UPDATE": -1,
      "VALIDATE": -1,
      "AUTHENTICATE": -1,
      "SCRIPT_ON_RESOURCE": -1,
      "SCRIPT_ON_CONNECTOR": -1
    },
    "poolConfigOption": {
      "maxIdle": 1,
      "maxWait": 150000,
      "minIdle": 1,
      "maxObjects": 1,
      "minEvictableIdleTimeMillis": 120000
    },
    "producerBufferSize": 100,
    "syncFailureHandler": {
      "maxRetries": 5,
      "postRetryAction": "logged-ignore"
    },
    "connectorPoolingSupported": true
  }
}
```


При создании коннектора осуществляются следующие проверки:

1) Наличие обязательных параметров в «connectorBody»/«connectorRef». Если отсутствует обязательный параметр, выводится ошибка, в деталях которой указано «failedPolicy»: «isValidConnectorRef».

2) Уникальность «nativeType» среди всех объектов коннектора. Если типы объектов не уникальны, выводится ошибка, в деталях которой указано «failedPolicy»: «isValidConnectorObjectNativeType».

Пример ошибки, когда не указаны обязательные параметры:


```
{
  "code": 403,
  "reason": "Forbidden",
  "message": "Policy validation failed: resource=managed/connector/*, property=connectorBody,
policyId=isValidConnectorBody, value=<..>",
  "detail": {
    "result": false,
    "localizedMessage": "policy.validation.failed",
    "policyId": "isValidConnectorBody",
    "resource": "managed/connector/*",
    "property": "connectorBody",
    "appErrorCode": 5000,
    "detail": {
      "failedPolicy": "isValidConnectorRef"
    },
    "message": null
  }
}
```


2.2.2. Настройка объекта «Тип ресурса»

Объект «Тип ресурса» («restype») содержит в себе перечень атрибутов подключения к ЦС, является определением для создаваемого в дальнейшем ресурса.

Для управления типами ресурсов администратор выполняет следующие действия:

- создание типа ресурса;
- редактирование типа ресурса;
- удаление типа ресурса.

Управление типами ресурсов выполняется администратором с помощью Rest-запроса.

Объекты «Тип ресурса» приведены в таблице 2.1.

Таблица 2.1 – Объекты «Тип ресурса»

Параметр	Описание	Обязательность
resTypeName	Имя типа ресурса	да
resTypeBody	Основные настройки, включает в себя подразделы	да
resTypeBody/properties	Перечень атрибутов ресурса, их типов («type») и	да

Параметр	Описание	Обязательность
	значений по умолчанию («default»). Поддерживаемые типы: – string; – integer; – boolean; – "$ref": "#/definitions/stringArray" (специальный тип для указания массивов значений string)	
resTypeBody/order	Порядок, в котором будут отображаться атрибуты в ресурсе. Должен содержать все атрибуты, указанные в resTypeBody/properties	да
resTypeBody/private	Массив атрибуты, для которых значения в ресурсе будут маскироваться знаком "*"	да

Пример настройки «resTypeBody/properties»:

```
"language": {
"type": "string",
"default": "EN"
},
"reconcileDeletedOrganizations": {
"type": "boolean",
"default": false
},
"topOrganizations": {
"$ref": "#/definitions/stringArray",
"default":[]
}
```

Пример настройки «resTypeBody/order»:

```
"order": ["configurationFilePath",
"appServerHost" ]
```

2.2.3. Настройка объекта «Ресурс»

Объект «Ресурс» в Комплексе содержит параметры подключения к ЦС.

Для управления ресурсами администратор выполняет следующие действия:

- создание ресурса;
- редактирование ресурса;
- удаление ресурса.

Управление ресурсами выполняется администратором двумя способами:

- с помощью Rest-запроса;
- в меню «Админ. системы».

Параметры объекта «Ресурс» приведены в таблице 2.2.

Таблица 2.2 – Параметры объекта «Ресурс»

Параметр	Описание	Обязательность	Пример
resName	Имя ресурса	да	1C Database
resDesc	Описание ресурса	нет	Параметры подключения к кадровой системе
restype_id	Ссылка на тип ресурса	да	1
connector_id	Ссылка на используемый коннектор	нет в случае, когда выбран тип ресурса «ICF Connector Server»	2
connectorServer_id	Ссылка на используемый сервер коннекторов	нет	3
resBody	Значения параметров подключения	нет, в случае автономного типа ресурса «Offline resource», т.к. параметров соединения с системой может не быть	-

Пример настройки для параметра «resBody»:

```
    "host": {
       "type": "string",
       "default": "localhost"
    },
    "port": {
       "type": "integer",
       "default": 389
    },
    "useSsl": {
       "type": "boolean",
       "default": false
    },
    "baseContexts": {
       "$ref": "#/definitions/stringArray",
       "default": ["ou=Users,dc=gis,dc=lan", "ou=ServiceAccounts,dc=gis,dc=lan"]
    }
}
```

Кроме типов ресурсов для подключения к ЦС Комплекс поддерживает следующие специальные типы ресурсов:

1) Коннектор сервер «ICF Connector Server» – ресурсы данного типа содержат параметры подключения к серверу коннекторов ICF, через который выполняется подключение к ЦС.
2) Автономный ресурс «Offline resource» – ресурсы данного типа не содержат параметров подключения. Они могут использоваться, когда

удаленное управление УЗ в ЦС невозможно (например, отсутствует API), но при этом необходимо вести учет выделенных УЗ пользователям. Offline-ресурс выделяется пользователям аналогичным образом, как для обычных ресурсов ЦС. Операции синхронизации для таких ресурсов не предусмотрены, т.к. фактического подключения к ЦС нет.

2.2.3.1. Создание ресурса

При создании ресурса выполняется следующее условие: для типа ресурса «ICF Connector Server» игнорируется значение указанное в параметре «Коннектор».

Для создания ресурса в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Ресурсы», затем нажать кнопку «Создать». Появится страница создания ресурса.
2) Заполнить обязательные поля «Имя» и «Тип». При выборе типа ресурса отображаются параметры конфигурации коннектора, которые можно отредактировать в соответствии с параметрами конкретной системы. Выбрать необходимый коннектор в поле «Коннектор». При необходимости заполнить описание ресурса.
3) Если для подключения к системе используется коннектор сервер, то его необходимо выбрать в поле «Сервер коннекторов».
4) Нажать кнопку «Создать». Чтобы увидеть созданный ресурс, в «Списке ресурсов» следует нажать кнопку «Обновить».

В случае подключения к ЦС через сервер коннекторов администратору необходимо сначала создать ресурс для сервера коннекторов, а затем создать ресурс для ЦС.

При создании ресурса для ЦС необходимо выбрать в поле «Сервер коннекторов» один из ресурсов с типом «ICF Connector Server».

2.2.3.2. Редактирование ресурса

При редактировании ресурса выполняются следующие условия:

1) Параметры «Тип ресурса» и «Коннектор» недоступны для редактирования. При необходимости их изменения должен создаваться новый ресурс.
2) Параметр «Коннектор» при редактировании не отображается.

Для редактирования ресурса в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Ресурсы» и выбрать из списка ресурсов необходимый ресурс. Откроется карточка ресурса с параметрами.
2) Изменить параметры ресурса и нажать кнопку «Сохранить».

2.2.3.3. Удаление ресурса

При удалении ресурса выполняются следующие условия:

1) Нельзя удалить ресурс, связанный с объектом «Учетная запись ресурса». Для этого сначала должен быть удален сам объект «Учетная запись ресурса».
2) Нельзя удалить ресурс сервера коннекторов (тип ресурса «ICF Connector Server»), если он связан с другим ресурсом. Для этого сначала должна быть удалена связь в ресурсе с коннектор сервером.
3) При удалении ресурса автоматически удаляются все связанные с ним записи из справочников.

Для удаления ресурса в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Ресурсы» и выбрать из списка один или несколько ресурсов. Откроется окно подтверждения с перечнем удаляемых ресурсов.
2) Нажать кнопку «Удалить».

2.2.3.4. Настройка ресурса с типом « ICF Connector Server»

Комплекс поддерживает встроенный не редактируемый служебный тип ресурса «ICF Connector Server», используемый для подключения Ankey IDM к серверу коннекторов. Параметры заполняемых параметров для ресурсов с типом «ICF Connector Server» приведены в таблице 2.3.

Таблица 2.3 – Параметры объекта «Ресурс»

Параметр	Описание	Пример
host	Имя хоста или IP-адрес компьютера, на котором установлен сервер коннекторов	localhost
port	Номер порта, на котором прослушивается сервер коннекторов	8759
useSSL	Параметр, отвечающий за безопасную передачу данных по протоколу Secure Sockets Layer (SSL). Параметр имеет значение «true», если настроен SSL между Ankey IDM и сервером коннекторов, в противном случае «false»	false
timeout	Целое значение, указывающее количество миллисекунд, по истечении которых соединение между сервером коннекторов и Ankey IDM превысит лимит времени. Если параметр установлен в значение «0», то лимит времени отсутствует	0
protocol	Протокол взаимодействия с сервером коннекторов. По умолчанию необходимо задавать значение «websocket»	websocket
key	Ключ для аутентификации на сервере коннекторов	*******

2.2.4. Настройка объекта «Форма ресурса»

Объект «Форма ресурса» («resform») представляет собой шаблон УЗ в ЦС с перечнем атрибутов.

Для управления формами ресурсов администратор выполняет следующие действия:

- создание формы ресурса;
- редактирование формы ресурса;
- удаление формы ресурса.

Управление формой ресурса выполняется администратором с помощью Rest-запроса.

Параметры объекта «Форма ресурса» приведены в таблице 2.4.

Таблица 2.4 – Параметры объекта «Форма ресурса»

Атрибут	Описание	Пример
resFormName	Имя формы ресурс	ADForm
resFormDesc	Описание формы ресурса	AD Form
resFormTable	Имя таблицы, в которой будут храниться учётные записи	rf_ad
displayField	Атрибут УЗ, значение которого будет её отображаемым именем в системе. Например, при отображении списка учётных записей будет отображаться значение этого атрибута. С целью	userprincipalname

Атрибут	Описание	Пример
	идентификации разных учётных записей в списках, рекомендуется использовать уникальный атрибут. В случае, если настройка не задана, будет использоваться идентификатор УЗ в ЦС («__UID__»)	
resFormBody	Настройка полей формы ресурса	–
resFormBody/properties	Настройка полей формы ресурса. Поддерживаемые типы полей «fieldType»: – textField – однострочное текстовое поле; – textArea – многострочное текстовое поле; – number – однострочное поле для ввода цифр; – passwordField – однострочное поле, скрывающее введённый текст; – flag – флаг с отображением в виде чекбокса; – lookup – текстовое поле с возможностью поиска и выбора из предопределённых значений; – date – дата с отображением в виде календаря; – timestamp – дата со временем, с отображением в виде календаря и возможностью указывать время; – multivaluedField – многозначное поле с отображением в виде таблицы	–
resFormBody/order	Порядок отображения атрибутов формы ресурсов в интерфейсе Комплекса	–
resFormBody/required	Список обязательных атрибутов формы ресурсов	–

Пример настройки полей формы ресурса:

```
{
  "resFormName": "ADForm",
  "resFormDesc": "AD Form",
  "resFormTable": "rf_ad",
  "displayField": "userprincipalname",
  "resFormBody": {
    "properties": {
      "userprincipalname": {
        "fieldType": "textField",
        "displayName": "UserPrincipaName"
      },
      "password": {
        "fieldType": "passwordField",
        "displayName": "Password"
      },
      "samaccountname": {
        "fieldType": "textField",
        "displayName": "User logon name",
        "maxLength": 20
      },
      "cn": {
        "fieldType": "textField",
        "displayName": "Common name (CN)",
```

```
      "maxLength": 64
     },
     "sn": {
      "fieldType": "textField",
      "displayName": "Last name",
      "maxLength": 64
     },
     "memberofs": {
      "fieldType": "multivaluedField",
      "displayName": "Member of",
      "properties": {
       "memberof": {
        "fieldType": "lookup",
        "displayName": "AD Groups",
        "keyField": true,
        "object": "managed/referencebook",
        "filter": "/refBookType eq \"AD Groups\"",
        "searchStartLength": 3,
        "resultsDisplayAmount": 5,
        "lookupField": "refBookCode",
        "placeholder": "templates.managed.form.approleresaccount.placeholder",
        "lookupDisplayFields": "refBookName",
        "lookupDisplayFields2": "refBookCode"
       }
      },
      "order": [
       "memberof"
      ],
      "required": [
       "memberof"
      ]
     },
     "orgunit": {
      "filter": "/refBookType eq \"AD Organizations\"",
      "object": "managed/referencebook",
      "fieldType": "lookup",
      "displayName": "Org Unit",
      "lookupDisplayFields": "refBookName",
      "lookupField": "refBookCode"
     }
    },
    "order": [
     "userprincipalname",
     "samaccountname",
     "cn",
     "password",
     "orgunit",
     "sn",
     "memberofs"
    ],
    "required": [
     "cn",
     "samaccountname"
    ]
   }
}
```

2.2.4.1. Настройка поискового поля с типом «lookup»

Поисковые поля с типом «lookup» позволяют для выбора значения выполнять поиск по определенному настройками справочнику. Поиск при этом выполняется как по коду, так и наименованию значения из справочника. Поисковые поля могут настраиваться как отдельно в форме ресурса, так и в составе многозначного атрибута формы ресурса.

Ограничение при использовании поисковых полей: при добавлении нового объекта, в поле типа lookup-поле значение должно быть обязательно типом string (не number, boolean, и т.д.).

Пример настройки отдельного поискового поля в форме ресурса:

```
"usertype": {
   "fieldType": "lookup",
   "displayName": "Тип пользователя",
   "maxLength": 100,
   "object": "managed/referencebook",
   "filter": "/refBookType eq \"UserType\"",
   "lookupField": "refBookCode",
   "lookupDisplayFields": "refBookName"
}
```

Параметры настройки отдельного поискового поля приведены в таблице 2.5.

Таблица 2.5 – Параметры настройки отдельного поискового поля

Параметр	Описание	Пример
fieldType	Тип атрибута формы ресурса	lookup
displayName	Отображаемое наименование атрибута	Тип пользователя
maxLength	Максимальная длина хранимого значения атрибута	100
object	Имя ссылочного объекта системы, из которого будут выбираться значения	managed/referencebook
filter	Фильтр «queryFilter», накладываемый на значения объекта. В случае, если в объекте присутствует поле идентификатор ресурса «resource_id», к фильтру автоматически добавится условие «and resource_id eq *значение»*	/refBookType eq \"UserType\"
lookupField	Атрибут ссылочного объекта, значение которого будет записано в данное поле	refBookCode
lookupDisplayFields	Атрибут ссылочного объекта, являющийся отображаемым значением поля, так же показывается при выборе объекта из списка, как основной	refBookName

Список поддерживаемых операторов при поиске в фильтре «filter» приведен в таблице 2.6.

Таблица 2.6 – Список поддерживаемых операторов при поиске в фильтре «filter»

Операция	Описание	Примеры
eq	Поиск объектов с указанным значением атрибута без учёта регистра	lastName eq 'Глазырин'
in	Содержит значение из списка	lastName in 'Иванов,Адуев'
and	Логическая операция объединения двух условий	lastName eq 'Макаров' and firstName eq 'Анатолий'
pr	У атрибута задано значение	description pr
!	Отрицание	!(description pr)
boolean-фильтр	Фильтр для выборки либо всех объектов, либо ни одного	true/false

2.2.4.2. Настройка многозначных полей с типом «multivaluedField»

Многозначные поля позволяют задавать множество уникальных значений, связанных с УЗ, например, ее полномочия (роли, группы и т.п.). Многозначные поля могут включать в свой состав поисковые поля.

Ограничение при использовании поисковых полей: при добавлении нового объекта, в поле типа lookup-поле значение должно быть обязательно типом string (не number, boolean, и т.д.).

Пример настройки многозначного поля:

```
"groups": {
   "fieldType": "multivaluedField",
   "displayName": "Группы",
   "properties": {
      "groupname": {
         "fieldType": "lookup",
         "displayName": "Имя группы",
         "keyField": true,
         "object": "managed/referencebook",
         "filter": "/refBookType eq \"Groups\"",
         "lookupField": "refBookCode",
         "placeholder": "templates.managed.form.approleresaccount.placeholder",
         "searchStartLength": 3,
         "resultsDisplayAmount": 5,
         "lookupDisplayFields": "refBookName",
         "lookupDisplayFields2": "refBookCode"
      },
      "read": {
         "fieldType": "flag",
         "displayName": "Чтение"
```

```
      },
      "write": {
         "fieldType": "flag",
         "displayName": "Запись"
      },
      "startDate": {
         "fieldType": "date",
         "displayName": "Дата начала"
      },
      "endDate": {
         "fieldType": "date",
         "displayName": "Дата окончания"
      }
   },
   "order": [
      "groupname",
      "read",
      "write",
      "startDate",
      "endDate"
   ],
   "required": [
      "groupname"
   ]
}
```


Параметры настройки многозначного поля приведены в таблице 2.7.

Таблица 2.7 – Параметры настройки многозначного поля

Параметр	Описание	Пример
fieldType	Тип атрибута формы ресурса	multivaluedField
displayName	Отображаемое наименование атрибута	Группы
properties	Содержит список атрибутов, в многозначное поле, в формате properties/<Имя атрибута>/	"properties": {"groupname":{...}, "read"{...}, "write"{...}, "startDate"{...}, "endDate"]
order	Порядок отображения атрибутов, включенных в многозначное поле	["groupname", "read", "write", "startDate", "endDate"]
required	Список атрибутов, включенных в многозначное поле, для которых обязательно должно быть задано значение	["groupname"]

Параметры атрибутов, включенных в многозначное поле (properties/Имя атрибута/Параметры) приведены в таблице 2.8.

Таблица 2.8 – Параметры атрибутов, включенных в многозначное поле (properties/Имя атрибута/Параметры)

Параметр	Описание	Пример
fieldType	Тип атрибута формы ресурса	lookup
displayName	Отображаемое наименование атрибута	Имя группы
keyField	Параметр keyField у атрибутов многозначных полей используется для проверки существования значения многозначного поля. По совпадению атрибутов с выставленным свойством keyField определяется существует значение или нет. Значение считается существующим, если одновременно совпадают значения всех ключевых полей. Если хотя бы значение одного из ключевых атрибутов отличается, значение считается новым	true
object	Имя ссылочного объекта системы, из которого будут выбираться значения	managed/referencebook
filter	Фильтр «queryFilter», накладываемый на значения объекта. В случае, если в объекте присутствует поле идентификатор ресурса «resource_id», к фильтру автоматически добавится условие «and resource_id eq значение»	/refBookType eq \"Groups\"
searchStartLength	Количество вводимых символов, после которых начинает отправляться поисковый запрос. Сообщение берётся по указанному ключу для данной локали. Если ключ локализации не найден, он и будет отображаться в форме	3

Параметр	Описание	Пример
resultsDisplayAmount	Количество отображаемых результатов поиска (вариантов) в поисковом поле	5
lookupField	Атрибут ссылочного объекта, значение которого будет записано в данное поле	refBookCode
placeholder	Ключ локализации для подсказки, отображаемой при незаполненном поле при создании/редактировании объекта	templates.managed.form.approleresaccount.placeholder
lookupDisplayFields	Основной атрибут ссылочного объекта, отображаемый в интерфейсе при выборе значения в поле	refBookName
lookupDisplayFields2	Дополнительный атрибут ссылочного объекта, отображаемый в интерфейсе при выборе значения в поле	refBookCode

2.2.4.3. Настройка параметра шифрования значения атрибута формы ресурса «encryption»

Для всех типов полей может задаваться параметр шифрования значения «encryption». Пример настройки параметра «encryption» для текстового поля:

```
"properties":
{
    "userprincipalname":
       {
        "fieldType": "textField",
        "displayName": "UserPrincipaName",
        "encryption":
            {
             "cipher": "AES/CBC/PKCS5Padding",
             "key" : "custom-key"
            }
       },
....
}
```

Параметр «encryption» содержит следующие настройки:

– key – имя ключа шифрования в хранилище ключей Комплекса;

– cipher – алгоритм шифрования, поддерживаемый виртуальной машиной Java. По умолчанию используется алгоритм «AES /CBC/PKCS5Padding».

2.2.4.4. Настройка объекта «Форма ресурса» для создания полномочий в AD через создание роли

Для создания полномочий в AD необходимо, чтобы было включено создание роли по заявке («Руководство Администратора», п. 2.6.5.3 «Включение создания роли по заявке»). Также необходимо создать для управления группами AD форму ресурса и УЗР.

Созданные для управления полномочиями форму ресурса и УЗР следует, с помощью REST-запроса (см. «Руководство Администратора», п.п. 2.2.3 и 2.2.4), добавить в качестве дополнительных параметров multivaluedField в форме ресурсов для которой создаются полномочия.

Список дополнительных параметров multivaluedField для создания полномочий в AD:

– entitlementsResAccId - числовой параметр; идентификатор УЗР: указывает на УЗР, в котором хранятся УЗ создаваемых полномочий;
– entitlementsField - текстовый параметр; название поля в форме ресурса, в которое создаваемые полномочия записываются для передачи в ЦС;
– entitlementsLookupField - необязательный параметр; поле для поиска полномочий в справочнике, по умолчанию поиск работает по полю “refBookName”.

```
"memberofs": {
  "fieldType": "multivaluedField",
  "displayName": "Member of",

  "properties": {
    "memberof": {
      "fieldType": "lookup",
      "displayName": "AD Groups",
      "keyField": true,
      "object": "managed/referencebook",
      "filter": "/refBookType eq \"AD Groups\"",
      "searchStartLength": 3,
      "resultsDisplayAmount": 5,
```

```
      "lookupField": "refBookCode",
      "placeholder": "templates.managed.form.approleresaccount.placeholder",
      "lookupDisplayFields": "refBookName",
      "lookupDisplayFields2": "refBookCode",
      "entitlementsResAccId": 2,
      "entitlementsField": "dn"
    }
  },
  "order": [
    "memberof"
  ],
  "required": [
    "memberof"
  ]
}
```


Для создания полномочий в AD через заявку на создание роли необходимо выполнить следующие действия:

1) В разделе «Роли» нажать кнопку «Создать» и убедится, что включена возможность создания роли по заявке: открывшееся окно называется «Заявка на создание роли». Затем в поле «Название» ввести название роли.
2) На вкладке «УЗР» выбрать УЗР для которой была проведена настройка возможности создания полномочий и нажать кнопку «Добавить права».
3) В открывшемся после нажатия кнопки «Добавить права» модальном окне «Добавление прав» нажать кнопку «Добавить полномочия».
4) Ввести в строке название создаваемого нового полномочия. Если в строке нет никаких найденных значений, то появится кнопка «Добавить новое полномочие». Если кнопка не появляется, значит настройки возможности создания полномочий были произведены неправильно.
5) Сохранить добавленные полномочия: после нажатия кнопки «Добавить новые полномочия» добавленное полномочие сохраняется в списке полномочий. Для создаваемого полномочия, не найденного в справочнике, появится информационное сообщение «Новое полномочие будет создано после согласования роли».
6) Нажать кнопку «Сохранить» после завершения ввода полномочий.
7) Нажать кнопку «Отправить» после чего появится всплывающая надпись «Заявка на создание роли успешно создана».

Дальнейшее согласование заявки производится в соответствии с процедурами, изложенными в «Руководство пользователя», п. 2.1.1.

В деталях заявки («Руководство пользователя», п. 2.1.5. «Просмотр деталей заявки»), в сведениях об активируемых УЗ пользователя, новые полномочия помечены всплывающим информационным сообщением «Новые полномочия».

Новые полномочия вносятся в УЗР AD после первого запуска периодического задания типа workflowrequesttask (периодическое задание, отвечающее за работу заявочной системы - «Руководство Администратора», п. 2.4.5.). При следующих запусках периодического задания типа workflowrequesttask производится проверка наличия созданных полномочий в справочнике. Когда все создаваемые полномочия там окажутся, то заявка исполнится и будет создана роль с полномочиями для AD.

2.2.4.5. Настройка объекта «Форма ресурса» для создания полномочий в AD через редактирование роли

Для создания полномочий в AD через заявку на редактирование роли необходимо начать создавать полномочия при редактировании роли во вкладке УЗР.

Редактирование производится аналогично описанной выше процедуре для заявки на создание роли (требуется настройка возможности создания новых полномочий: создание формы ресурса и УЗР и т. д.), в окне «Заявка на редактирование роли» (открывается в разделе «Роли» кнопкой «Редактировать»).

В окне «Заявка на редактирование роли» для редактируемой УЗР необходимо выбрать кнопку «Изменить» и в модальном окне «Добавление прав» провести изменения аналогично процедуре создания полномочий в AD, описанной выше (п.п. 2 - 4). Кнопка «Добавить новые полномочия» расположена справа от записи (при наведении появляется соответствующее всплывающее информационное сообщение). Также как в п. 5, для создаваемого полномочия, не найденного в справочнике, появится информационное сообщение «Новое полномочие будет создано после согласования роли».

После сохранения изменений (кнопка «Сохранить»), сделанных в модальном окне «Добавление прав», необходимо в окне «Заявка на редактирование роли» заполнить поля для оформления заявки.

После оформления заявки на редактирование роли сделанные изменения отображаются в подразделе «Входящие заявки» (детали заявки, на вкладке «Информация») в списке «Запрашиваемые изменения» как для уже существующей, так и для вновь созданной записи. В деталях заявки вновь созданные полномочия (так же как при создания полномочий в AD через заявку на создание роли) помечены информационным сообщением «Новые полномочия».

Согласованная заявка на редактирование роли находится в режиме «Ожидает исполнения» до тех пор, пока создаваемые полномочия не будут внесены в справочник по описанной выше процедуре с периодическим заданием типа workflowrequesttask.

Если заявка исполняется, то новое полномочие отображается в деталях измененной роли (вкладка «Информация») в подразделе «Роли».

2.2.5. Настройка объекта «Маппинг»

Объект «Маппинг» (далее – маппинг) в Комплексе содержит параметры преобразования атрибутов объектов (пользователей, ролей, организаций и т.д.) при синхронизации из ЦС или при распространении в ЦС.

Для управления маппингом администратор выполняет следующие действия:

- просмотр маппинга.
- создание маппинга. Приведено в подпункте 2.2.5.1;
- редактирование маппинга. Приведено в подпункте 2.2.5.2;
- удаление маппинга. Приведено в подпункте 2.2.5.3.

Для просмотра настроенных маппингов в Комплексе администратору необходимо перейти в меню «Админ.системы»/«Маппинги».

Список отображает следующие данные:

- название;
- тип;
- источник;
- получатель.

Поиск в списке маппингов осуществляется по атрибутам: «Название», «Источник», «Получатель». Фильтрация выполняется по атрибуту «Тип».

Для фильтрации списка маппингов по типу администратору необходимо выполнить следующие действия:

1) Нажать «Фильтр» в крайнем правом углу строки поиска.
2) В выпадающем списке выбрать один или несколько статусов (recon, prepopulate, provision).
3) Нажать кнопку «Применить». Для отмены действия следует нажать кнопку «Отменить». Для сброса всех фильтров одновременно следует нажать «Очистить все».

В случае, если Комплекс не отображает результат поиска, следует запустить периодическое задание «reindextask».

Для просмотра настроек маппинга администратору необходимо выбрать маппинг в списке, после чего в правой части экрана отобразятся вкладки «Карточка маппинга», «Параметры», «Политики», «Скрипты». Для маппинга типа «prepopulate» вкладки «Политики», «Скрипты» недоступны.

Вкладка «Карточка маппинга» отображает следующие настройки:

- название;
- описание;
- тип;
- источник;
- получатель;
- правило корреляции. В правом верхнем углу поля приводится тип скрипта правила корреляции.

Вкладка «Параметры» отображает правила заполнения атрибутов получателя. Справа от правила приводится действие, для которого применяется данное правило (create, update, delete). Для просмотра настроек правила следует нажать по его наименованию.

Настройки правила содержат следующие параметры:

- источник. В случае отсутствия источника отображается значение «Отсутствует»;
- получатель;

- код преобразования. В правом верхнем углу поля приводится тип скрипта преобразования;
- код условия. В правом верхнем углу поля приводится тип скрипта условия выполнения преобразования.

Вкладка «Политики» отображает следующие настройки политик:

- ситуация;
- действие. Значение действия прописано в виде текста, кода скрипта, либо может быть прикреплен файл. Для просмотра кода скрипта следует нажать на кнопку «Код».

Для маппинга с типом «prepopulate» настройка политик не поддерживается.

Вкладка «Скрипты» отображает настройки скриптов для следующих действий:

- «onCreate», «onUpdate», «onDelete», «result» для маппинга типа «recon»;
- «onCreate», «onUpdate», «onDelete» для маппинга типа «provision».

Настройки скриптов отображаются в виде кода, либо файла скрипта. Для просмотра файла скрипта следует перейти в директорию проекта в папке «script».

Для маппинга с типом «prepopulate» настройка скриптов не поддерживается.

Настройки маппинга включают следующие параметры:

- mapName – имя маппинга;
- mapType – тип маппинга;
- mapDesc – описание маппинга;
- mapSource – ресурс-источник;
- mapTarget – ресурс-получатель;
- correlationQuery – правило связывания объектов ресурса-источника и ресурса-получателя;
- mapBody – описание правил маппинга;
- scripts – скрипты, вызываемые при различных событиях.

Типы маппингов mapType включают следующие типы:

- prepopulate – обеспечивает предзаполнение формы ресурса перед распространением данных в ЦС;

– provision – обеспечивает преобразование данных при распространении данных в ЦС;
– recon – обеспечивает преобразование данных при синхронизации из ЦС.

В качестве ресурсов источников и ресурсов получателей могут использоваться следующие объекты:

– любые управляемые managed объекты, например, пользователь (managed/user);
– формы ресурсов, например, форма ресурса службы каталога Active Directory (resform/aduser);
– специальный объект коннектора «connector/auto», он используется для передачи данных непосредственно в коннектор при распространении или синхронизации данных в/из ЦС;
– конкретный тип объекта коннектора в формате записи «connector/<Ресурс>/<Тип объекта коннектора>» (пример, connector/xml/account), такая запись используется при доверенной синхронизации сведений из внешних источников (например, из кадрового модуля).

Возможные сценарии использования маппингов приведены в таблице 2.9.

Таблица 2.9 – Возможные сценарии использования маппингов

Сценарий	**Тип маппинга**	**Ресурс-источник**	**Ресурс-получатель**
Создание новой УЗ в ЦС	– prepopulate; – provision	– managed/user; – resform/aduser	– resform/aduser; – connector/auto
Обновление атрибутов УЗ ЦС при смене атрибутов управляемого (managed) объекта	provision	managed/user	resform/aduser
Обновление атрибутов управляемого (managed) объекта данными из ЦС	provision	resform/aduser	managed/user
Синхронизация УЗ из ЦС	recon	connector/auto	resform/aduser
Синхронизация сведений об управляемом (managed) объекте из доверенного источника	recon	connector/xml/account	managed/user

Описание правил маппинга mapBody включает следующие параметры:

- policies – необязательные политики обработки различных ситуаций;
- properties – правила маппинга.

Политики (policies) могут присутствовать для типов provision и recon. Политики включает следующие параметры:

- situation – ситуация (событие), которая возникает в процессе сверки данных из ресурса-источника с ресурсом-получателем;
- action – действие, которое необходимо выполнить Комплексу в определенной ситуации.

Ситуации (situation) могут быть следующих типов:

- CONFIRMED – записи в ресурсе-источнике и ресурсе получателе найдены и имеют связь;
- FOUND – найдены записи в ресурсе-источнике и есть соответствующий объект в ресурсе получателе, при этом связи между ними не было;
- FOUND_ALREADY_LINKED – найдены записи в ресурсе-источнике и есть соответствующий объект в ресурсе-получателе, при этом связи между ними не было, и ресурс-получатель уже связан с другим объектом ресурса-источника;
- ABSENT – связь объекта из ресурса-источника не может быть построена с объектом ресурса-получателя, например, в случае, когда у объекта ресурса-получателя уже есть связь с другим объектом ресурса-источника;
- AMBIGUOUS – объект ресурса-источника соответствует нескольким объектам ресурсов-получателей;
- MISSING – у ресурса-источника есть связь, но отсутствует запись в ресурсе-получателе (например, УЗ была удалена в ЦС в обход Комплекса);
- UNQUALIFIED – объект ресурса-получателя не прошел валидацию (в случае если настроен скрипт «validTarget» в маппинге), но у него есть связь с объектом ресурса-источника;

- SOURCE_IGNORED – объект ресурса-источника не прошел валидацию (в случае если настроен скрипт «validSource» в маппинге), связь с объектами получателями не найдена;
- TARGET_IGNORED – объект ресурса-получателя не прошел валидацию (в случае если настроен скрипт «validTarget» в маппинге);
- UNASSIGNED – найден валидный объект ресурса-получателя, но у него нет связи с другими объектами;
- SOURCE_MISSING – найден объект ресурса-получателя, и у него есть связь с объектом ресурса-источника, но самого объекта ресурса-источника нет.

Действия (action) могут быть следующих типов:

- CREATE – создает объект ресурса-получателя и связь с объектом ресурса-источника;
- UPDATE – обновляет атрибуты объекта ресурса-получателя;
- EXCEPTION – помечает событие, как исключение;
- DELETE – удаляет объект ресурса-получателя;
- IGNORE – игнорирует изменения;
- REPORT – не выполняется никаких действий кроме вывода в журнал работы Комплекса тех действий, которые могли бы быть при выполнении действия по умолчанию;
- NOREPORT – ничего не выполнять и не формировать отчет;
- ASYNC – запущен асинхронный процесс, поэтому не выполнять никакие действия с событием и не формировать отчет;
- UNLINK – разорвать связь между ресурсом-источником и ресурсом-получателем;
- LINK – создает связь между ресурсом-источником и ресурсом-получателем.

Возможные действия в зависимости от ситуации для типа маппинга «recon» приведены в таблице 2.10.

Таблица 2.10 – Возможные действия в зависимости от ситуации для типа маппинга «recon»

Ситуация	Возможные действия
SOURCE_IGNORED	REPORT, IGNORE, EXCEPTION, NOREPORT, ASYNC
UNASSIGNED	EXCEPTION, DELETE, IGNORE
AMBIGUOUS	REPORT, IGNORE, EXCEPTION, NOREPORT, ASYNC
FOUND_ALREADY_LINKED	REPORT, IGNORE, EXCEPTION, NOREPORT, ASYNC
CONFIRMED	UPDATE, IGNORE, REPORT, NOREPORT, ASYNC, RESTORE
UNQUALIFIED	DELETE, EXCEPTION, IGNORE, REPORT, NOREPORT, ASYNC
LINK_ONLY	EXCEPTION, UNLINK, IGNORE, REPORT, NOREPORT, ASYNC
SOURCE_MISSING	REPORT, DELETE, CREATE_SOURCE, EXCEPTION
TARGET_IGNORED	REPORT, IGNORE, DELETE, UNLINK, EXCEPTION, NOREPORT, ASYNC
ABSENT	CREATE, EXCEPTION, IGNORE, REPORT, NOREPORT, ASYNC, DELETE
MISSING	EXCEPTION, CREATE, UNLINK, IGNORE, REPORT, NOREPORT, ASYNC
FOUND	UPDATE, EXCEPTION, IGNORE, REPORT, NOREPORT, ASYNC
ALL_GONE	NOREPORT, IGNORE, EXCEPTION, REPORT, ASYNC

Возможные действия в зависимости от ситуации для типа маппинга «provision» приведены в таблице 2.11.

Таблица 2.11 – Возможные действия для различных ситуации для типа маппинга «provision»

Ситуация	Возможные действия
SOURCE_IGNORED	REPORT
UNASSIGNED	EXCEPTION, IGNORE, REPORT, NOREPORT, ASYNC
AMBIGUOUS	EXCEPTION
FOUND_ALREADY_LINKED	EXCEPTION
CONFIRMED	UPDATE, IGNORE, REPORT, NOREPORT
UNQUALIFIED	DELETE, UNLINK, EXCEPTION, IGNORE, REPORT, NOREPORT, ASYNC
LINK_ONLY	NOREPORT
SOURCE_MISSING	DELETE, EXCEPTION, UNLINK, IGNORE, REPORT, NOREPORT, ASYNC
TARGET_IGNORED	REPORT, IGNORE, DELETE, UNLINK, NOREPORT, ASYNC
ABSENT	CREATE, EXCEPTION, IGNORE

Ситуация	Возможные действия
MISSING	CREATE, EXCEPTION
FOUND	UPDATE
ALL_GONE	NOREPORT

В случае, если политика не указана, используются правила обработки различных ситуаций по умолчанию, которые приведены в таблице 2.12.

Таблица 2.12 – Правила обработки различных ситуаций по умолчанию

Ситуация	Тип маппинга	Действие по умолчанию
CONFIRMED	provision/recon	UPDATE
FOUND_ALREADY_LINKED	provision/recon	EXCEPTION
ABSENT	provision/recon	CREATE
AMBIGUOUS	provision/recon	EXCEPTION
MISSING	recon	EXCEPTION
MISSING	provision	CREATE
UNQUALIFIED	provision/recon	DELETE
SOURCE_IGNORED	provision/recon	REPORT
TARGET_IGNORED	provision/recon	REPORT
UNASSIGNED	provision/recon	EXCEPTION
SOURCE_MISSING	provision/recon	REPORT

Правила маппинга (properties) включают следующие параметры:

- action – действие, для которого применяется данное правило;
- source – имя поля объекта источника (может быть пустым);
- target – имя поля объекта получателя (обязательный параметр);
- transform – скрипт преобразования поля данных из источника в получатель;
- condition – скрипт условия выполнения преобразования для конкретного атрибута.

Если условие для преобразования атрибута, заданное в параметре «condition», не выполняется, то значение для целевого атрибута (target) не задается.

Если действие (action) указано, то это преобразование выполняется только для этого действия (action). Если действие (action) не указано, то это преобразование

выполняется для любых действий. Для маппингов с типом «prepopulate» запрещено указывать действие (action).

Скрипты (scripts) маппинга, вызываются при наступлении ситуации (события) перед их фактическим выполнением (action). Скрипты могут использоваться для дополнительного формирования, преобразования атрибутов после применения маппинга, в том числе уведомлений.

Скрипты могут настраиваться на следующие действия:

- onCreate – скрипт вызывается перед созданием объекта;
- onUpdate – скрипт вызывается перед обновлением объекта;
- onDelete – скрипт вызывается перед удалением объекта;
- result – скрипт вызывается по результату выполнения всех преобразований маппинга.

Поддерживаются следующие типы скриптов («type»):

- «text/javascript» – скрипты написанные на языке JavaScript;
- «groovy» – скрипты написанные на языке Groovy.

Для скриптов, выполняемых на действия onCreate, onUpdate, onDelete, доступны следующие данные:

- mappingConfig – сведения о настройки маппинга;
- sourceId – сведения об идентификаторе ресурса-источника;
- source – сведения о ресурсе источнике (например, значения атрибутов);
- targetId – сведения об идентификаторе ресурса-получателя;
- target – сведения о ресурсе-получателе (например, значения атрибутов);
- oldTarget – предыдущее состояние ресурса-получателя (например, предыдущее значение атрибута);
- situation – сведения о типе ситуации.

Правило связывания ресурса-источника и ресурса-получателя «correlationQuery» может быть задано в следующих вариантах:

- в виде строки фильтра – параметр correlationQuery;

– в виде скрипта, возвращающего фильтр значений – параметр correlationQuery;
– в виде скрипта, возвращающего объект ресурса-получателя непосредственно – параметр correlationScript.

Если правило не задано явно, то для синхронизации используется связывание по совокупности полей служебного идентификатора объекта (systemObjectId) и идентификатора УЗР пользователя (resAccountId). Атрибуты correlationQuery и correlationScript не могут быть заданы в маппинге одновременно.

Пример настройки атрибута correlationQuery в виде строки фильтра:

```
\"correlationQuery\" : \"userName eq \'${userName}\'\"
```

Настройка атрибута correlationQuery задается в виде строки в формате «queryFilter». Для подстановки значений из объекта ресурса-источника используются разделители вида: ${имя-свойства}. Строки необходимо помещать в кавычки.

Пример настройки атрибута correlationQuery в виде скрипта, возвращающего фильтр значений:

```
{
...
        "correlationQuery" : {
           "type" : "text/javascript",
              "source" : "var qry = { '_queryFilter': '_id eq \'" + source.userName + "\'" }; qry;"
        },
...
}
```

Скрипт, возвращающий фильтр значений в качестве аргументов может использовать поля «_queryFilter» или «_queryId» для запроса.

Пример настройки атрибута correlationScript, возвращающего объект ресурса-получателя непосредственно:

```
{
...
      "correlationScript" : {
         "type" : "text/javascript",
         "file" : "script/correlateScript.js"
      },
...
}
```

2.2.5.1. Создание объекта «Маппинг»

Создание маппинга выполняется администратором двумя способами:

1) Через консоль администрирования.
2) С помощью Rest-запроса. Примеры приведены в приложении А.

Для создания маппинга в подразделе «Маппинги» администратору необходимо выполнить следующие действия:

1) Нажать на кнопку «Создать». Откроется окно «Создание маппинга». В левой части окна расположены вкладки «Карточка маппинга», «Параметры», «Политики», «Правило корреляции», «Скрипты». Некоторые вкладки могут быть недоступными в зависимости от заданного значения параметра «Тип».
2) На вкладке «Карточка маппинга» задать значения обязательных полей: «Название», «Тип», «Тип источника», «Источник», «Тип получателя», «Получатель». При указании значений следует ориентироваться на возможные сценарии использования маппингов, которые приведены в таблице 2.9.
3) Перейти на вкладку «Параметры». Нажать на кнопку «Добавить параметр» и задать значения следующих параметров:
   - источник;
   - получатель;
   - действие. По умолчанию выбрано значение «Отсутствует». Поле не отображается для маппингов с типом «prepopulate».
   - код преобразования. Нажать «Добавить преобразование». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже.
   - код условия. Нажать «Добавить условие». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже.
4) Перейти на вкладку «Политики» и добавить политику маппинга:
   - ситуация. Выбрать ситуацию из выпадающего списка;

– простое действие. Нажать на кнопку «Простое действие» и заполнить поле «Действие». Кнопка «Простое действие» неактивна пока не выбрана ситуация;
– действие по скрипту. Нажать на кнопку «Действие по скрипту». Общие действия при выборе типа скрипта описаны ниже.

5) Нажать на кнопку «Сохранить».
6) Перейти на вкладку «Правило корреляции». Добавить правило корреляции, указав один из типов скрипта «Java Script», «Groovy», «Query». Общие действия при выборе типа скрипта «Java Script» или «Groovy» описаны ниже.
7) Нажать на кнопку «Применить».
8) Перейти на вкладку «Скрипты». Нажать на кнопку «Добавить скрипт» и задать значения следующих параметров:
   – действие. Выбрать действие из выпадающего списка. Возможные действия приведены в пункте 2.2.5;
   – скрипт. Нажать на кнопку «Назначить скрипт». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже.
9) Нажать на кнопку «Сохранить». Для отмены действий следует нажать на кнопку «Отменить». В случае ошибки отображается сообщение «Ошибка при создании маппинга».

Общие действия при выборе типа скрипта:

1) Нажать на кнопку «Java Script» или «Groovy» для выбора типа скрипта.
2) Выбрать одно из двух действий:
   – нажать на кнопку «Текст», чтобы внести значение в виде кода в текстовое поле.
   – нажать на кнопку «Файл», чтобы внести значение в формате файла скрипта. Следует указать название файла, который находится в директории проекта в папке «script». Если файл отсутствует, то при сохранении изменений отобразится ошибка.

При ручном вводе кода скрипта выполняется подстветка синтаксиса, валидация и автодополнение. Для типов скрипта «Java Script», «Groovy» доступны кнопки «Копировать» и «Форматировать». Кнопка «Копировать» позволяет скопировать код в буфер обмена. Кнопка «Форматировать» позволяет автоматически форматировать код.

При создании маппинга следует обратить внимание на следующие моменты:

1) При смене типа маппинга будут сброшены все настройки, кроме имени и описания.
2) При смене значения «Тип» отображается предупреждение «При изменении основных настроек маппинга, все остальные данные будут сброшены! Сбросить выбранное?»
3) При смене источника/получателя будут сброшены все параметры, если они были заполнены. Все остальные настройки сохраняются.
4) Нельзя создать маппинг, если уже существует маппинг с такими же настройками параметров «mapType», «mapSource» и «mapTarget» одновременно. Описание параметров приведено в пункте 2.2.5.
5) Для значения connector/auto в источнике/получателе необходимо указывать форму ресурса (resform). Если у указанной формы ресурса отсутствует УЗР, то на вкладке «Параметры» отобразится предупреждение «Не создана УЗР для выбранной формы ресурса».

2.2.5.2. Редактирование объекта «Маппинг»

Для редактирования маппинга необходимо в подразделе «Маппинги» выполнить следующие действия:

1) Выбрать маппинг в списке.
2) Нажать на кнопку «Редактировать». Откроется окно «Редактирование маппинга». В левой части окна расположены вкладки «Карточка маппинга», «Параметры», «Политики», «Правило корреляции», «Скрипты». Некоторые вкладки могут быть недоступными в зависимости от заданного значения параметра «Тип».

3) Перейти на вкладку «Карточка маппинга» для изменения поля «Описание».
4) Перейти на вкладку «Параметры» для изменения параметров маппинга. Нажать на кнопку «Добавить параметр», в случае если параметр отсутствует или кнопку «Редактировать» напротив параметра. Откроется окно «Редактирование параметра». Для всех параметров, где в «Карточке маппинга» явно указан тип источника/получателя, будет выпадающий список с доступными значениями. Для значения «Форма ресурса», кроме пользовательских параметров, доступно также системное поле «Status». Доступные поля для изменения:
   – источник;
   – получатель;
   – действие. По умолчанию выбрано значение «Отсутствует». Поле не отображается для маппингов с типом «prepopulate»;
   – код преобразования. Нажать «Добавить преобразование». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже;
   – код условия. Нажать «Добавить условие». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже.
5) Нажать на кнопку «Сохранить».
6) Перейти на вкладку «Политики». Вкладка «Политики» недоступна для маппингов с типом «prepopulate». Нажать на кнопку «Добавить политику маппинга», в случае если политика отсутствует или кнопку «Редактировать» напротив политики. Откроется окно «Редактирование политики». Доступные поля для изменения:
   – ситуация. Для изменения выбрать ситуацию из выпадающего списка;
   – простое действие. Для изменения нажать на кнопку «Простое действие» и заполнить поле «Действие». Кнопка «Простое действие» неактивна, пока не выбрана ситуация;

– действие по скрипту. Для изменения нажать на кнопку «Действие по скрипту». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже.

7) Нажать на кнопку «Сохранить».
8) Перейти на вкладку «Правило корреляции». Вкладка «Правило корреляции» доступна только для маппингов с типом «recon». Нажать на кнопку «Добавить правило корреляции» в случае, если правило отсутствует или «Изменить правило корреляции» в случае, если правило добавлено ранее. Внести изменение, выбрав один из типов скрипта «Java Script», «Groovy», «Query». Общие действия при выборе типа скрипта «Java Script» или «Groovy» описаны ниже.
9) Нажать на кнопку «Применить».
10) Перейти на вкладку «Скрипты». Нажать на кнопку «Редактировать» и задать значения следующих параметров:
    – действие. Для изменения выбрать действие из выпадающего списка. Возможные действия приведены в пункте 2.2.5;
    – скрипт. Внести изменения в существующие значения или нажать на кнопку «Удалить скрипт», затем нажать на кнопку «Назначить скрипт». Выбрать тип скрипта, а затем выбрать текст или файл. Общие действия при выборе типа скрипта описаны ниже.
11) Нажать на кнопку «Сохранить». Для отмены действий следует нажать на кнопку «Отменить». В случае ошибки отображается сообщение «Ошибка при обновлении маппинга».

Общие действия при выборе типа скрипта:

1) Нажать на кнопку «Java Script» или «Groovy» для выбора типа скрипта.
2) Выбрать одно из двух действий:
    – нажать на кнопку «Текст», чтобы внести значение в виде кода в текстовое поле.
    – нажать на кнопку «Файл», чтобы внести значение в формате файла скрипта. Следует указать название файла, который находится в

директории проекта в папке «script». Если файл отсутствует, то при сохранении изменений отобразится ошибка.

При ручном вводе кода скрипта выполняется подстветка синтаксиса, валидация и автодополнение. Для типов скрипта «Java Script», «Groovy» доступны кнопки «Копировать» и «Форматировать». Кнопка «Копировать» позволяет скопировать код в буфер обмена. Кнопка «Форматировать» позволяет автоматически форматировать код.

2.2.5.3. Удаление объекта «Маппинг»

Для удаления маппинга необходимо в подразделе «Маппинги» выполнить следующие действия:

1) Выбрать маппинг из списка.
2) Нажать кнопку «Удалить». Откроется окно «Удаление маппингов».
3) В окне «Удаление маппингов» нажать кнопку «Удалить». После успешного удаления отобразится сообщение «Маппинги успешно удалены». В случае ошибки удаления отобразится сообщение «Ошибка при удалении маппингов». Для отмены действия следует нажать кнопку «Отменить».

2.3. Управление учетной записи ресурса

УЗР в Комплексе – это объект, который обеспечивает связку УЗ ЦС с их владельцами. После установки связи владелец может выполнять действия со своей УЗ ЦС из интерфейса Комплекса. Связь с владельцем (бенефициаром) может строиться в следующих режимах:

1) Автоматически, при совпадении ключевых атрибутов УЗ ЦС и владельца.
2) Вручную администратором для УЗ ЦС, которые нельзя однозначно связать с владельцем по ключевым атрибутам.

Управление УЗР включает в себя следующие действия администратора:

- создание УЗР;
- редактирование УЗР;
- связывание УЗ с ее владельцем;

– удаление УЗ пользователя;
– изменение типа УЗ;
– смена парольной политики УЗР.

2.3.1. Создание учетной записи ресурса

Создание УЗР выполняется администратором двумя способами:

1) С помощью Rest-запроса.
2) Через консоль администрирования.

Для создания УЗР в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Выбрать пункт «Учетные записи ресурсов», затем нажать кнопку «Добавить учетную запись ресурса». Появится страница создания УЗР.
2) Заполнить обязательные поля: «Имя», «Бенефициар», «Тип учетной записи» и выбрать нужный ресурс.
3) При необходимости заполнить необязательные поля: «Описание», «Форма ресурса», «Тип правила связывания», «Правило связывания».
4) Нажать кнопку «Сохранить».

Для просмотра УЗ администратору необходимо зайти в «Учетные записи ресурсов» и нажать кнопку «Обновить».

Пример заполнения атрибутов «Учетной записи ресурса» приведен в таблице 2.13.

Таблица 2.13 – Пример заполнения атрибутов «Учетной записи ресурса»

Атрибут	Описание	Пример
Имя	Имя УЗР	CPM
Описание	Описание УЗР	Сетевое рабочее место
Бенефициар	Владелец УЗ в случае синхронизации с ЦС или синхронизируемый объект в случае доверенной синхронизации/По умолчанию можно выбрать одного из владельцев: 1) Пользователь. 2) Организация. 3) Роль. 4) Заместитель. 5) Информационная система При необходимости расширить список поддерживаемых объектов, нужно указать	Пользователь

Атрибут	Описание	Пример
	дополнительные значения через расширение enum-поля beneficiarObject у объекта managed/resaccount	
Ресурс Форма ресурса	Параметры соединения с системой (имя ресурса). Поиск осуществляется по имени и описанию ресурса. Для создания УЗР для автономного ресурса, необходимо выбрать ресурс «Offline resource» или другой ресурс созданный по типу ресурса «Offline resource» Форма ресурса, заполняется в случае целевой синхронизации. Поиск осуществляется по имени и описанию формы ресурса	AD gis.lan adform
Тип объекта коннектора	Объект системы, с которым будет работать коннектор	account
Тип правила связывания	Тип правила связывания. По умолчанию можно выбрать один из типов: 1) CORRELATION_QUERY_STRING. 2) CORRELATION_QUERY. 3) CORRELATION_QUERY_SCRIPT	CORRELATION_QUERY_STRING
Правило связывания	Правила связывания УЗ с владельцем в случае синхронизации с ЦС. Описание настройки правила связывания приведено в подразделе 2.16.2	userName eq '${username}'

2.3.2. Редактирование учетной записи ресурса

Для редактирования доступны следующие атрибуты:

- Имя;
- Описание;
- Тип правила связывания;
- Правило связывания.

Для редактирования УЗР через консоль администрирования администратору необходимо выполнить следующие действия:

1) В разделе «Учетные записи ресурсов» выбрать УЗР. Откроется вкладка «Карточка учетной записи ресурса» с параметрами.
2) Выполнить редактирование необходимых параметров и нажать кнопку «Сохранить».

2.3.3. Связывание УЗ пользователя с ее владельцем

Связывание УЗ с владельцем доступно как для УЗ без владельца, так и для УЗ, у которых владелец уже определен – замена владельца.

Связывание УЗ с владельцами выполняется в меню «Админ.системы»/«Учетные записи ресурсов» в подразделе «Учетные записи».

Список УЗ доступен для тех УЗР, для которых настроена форма ресурса. Информация, которую содержит список УЗ в подразделе «Учетные записи» приведена в таблице 2.14.

Таблица 2.14 – Информация, которую содержит список УЗ в подразделе «Учетные записи»

Параметр	**Описание**	**Примечание**
Учётная запись	Отображаемое имя УЗ	Настройка выполняется для формы ресурса
Идентификатор	Идентификатор УЗ в ЦС	Определяется атрибутом «**UID**» коннектора
Владелец	Идентификатор владельца УЗ	
Тип УЗ	Тип УЗ	Возможные значения: 1) Основная (primary). Для данного типа применяются операции назначения, отзыва полномочий ролью. 2) Дополнительная (other). Данный тип у пользователя может быть в случае наличия в ЦС двух и более УЗ для одного сотрудника. Операции назначения, отзыва полномочий ролью для данного типа не поддерживаются. 3) Служебная (service). УЗ данного типа не удаляются в случае удаления владельца, они отвязываются с возможностью связи с другим владельцем. Также для служебных УЗ не применяются изменения из маппингов с типом «provision» (смена логина, пароля). Пример УЗ ЦС данного типа это системные УЗ (пример, «root» или «Администратор» в ОС). 4) Не определён (undefined). Данный тип отображается для УЗ без владельцев
Статус корреляции	Статус связи УЗ с владельцем	Возможные значения: – связана (в случае успешного связывания); – соответствий не найдено (владелец не найден); – множественное соответствие (несколько соответствий найдено)
Статус	Статус УЗ	Возможные значения: – активна (active); – заблокирована (disabled); – создаётся (provisioning)

Поиск в списке УЗ ведётся по колонкам «Учетная запись» и «Идентификатор».

Для поиска по УЗ необходимо запустить периодическое задание «reindex» после создания формы ресурсов.

Для связывания УЗ с ее владельцем администратору необходимо выполнить следующие действия:

1) Выбрать необходимую УЗР в разделе «Учетные записи ресурсов». Откроется карточка УЗР с параметрами.
2) Перейти в подраздел «Учетные записи». В списке УЗ выбрать одну запись и нажать кнопку «Выбор владельца».
3) Во всплывающем диалоговом окне выполнить поиск пользователя. Поиск выполняется по атрибутам «Имя для входа» и «Полное имя». Для просмотра дополнительных деталей пользователя следует раскрыть структуру. Отобразится карточка деталей пользователя. Для просмотра структуры подразделений необходимо в карточке деталей пользователя нажать на кнопку  .
4) В списке пользователей выбрать владельца УЗ и нажать кнопку «Выбрать».

Для отвязывания владельца от УЗ администратору необходимо перейти в раздел «Учетные записи» и там выбрать запись в списке УЗ, а потом нажать кнопку «Отвязать владельца». Кнопка активна только для УЗ со статусом «Активна» или «Заблокирована». Если владелец отсутствует или статус УЗ «Создается», то кнопка будет заблокирована.

Если отвязать или сменить владельца основной УЗ таким образом, что у него остаются УЗ с типом «дополнительная», то у одной из таких УЗ автоматически сменится тип на «основная».

2.3.4. Удаление УЗ пользователя

Для удаление УЗ пользователя администратору необходимо выполнить следующие действия:

1) Выбрать необходимую УЗР в разделе «Учетные записи ресурсов». Откроется карточка УЗР с параметрами.

2) Перейти в подраздел «Учетные записи». В списке УЗ выбрать одну или несколько записей и нажать кнопку «Удалить».
3) В окне подтверждения действия по удалению УЗ нажать кнопку «Удалить».

Администратор может удалить УЗ пользователей в любых статусах.

2.3.5. Изменение типа УЗ пользователя

При изменении типа УЗ пользователя выполняются следующие условия:

1) Изменение типа УЗ возможно осуществлять для УЗ типов «Основная», «Дополнительная», «Служебная». Для УЗ типа «Основная» возможна смена типа только на тип «Служебная».
2) При смене типа УЗ с «Дополнительная» на «Основная» будет автоматически изменен тип у текущей основной УЗ пользователя на «Дополнительная». При этом автоматически отзываются все привилегии, назначенные через роли. После запуска периодической задачи для пересчета привилегий (evaluatetask) на новую основную УЗ добавляются привилегии, выделяемые ролями пользователя. Описание поддерживаемых типов УЗ пользователя приведено в пункте 2.3.3.
3) Смена типа УЗ поддерживается только для УЗ, связанных с владельцем. Если УЗ назначается исполнением роли, то смена типа «Основная»/«Служебная» недоступна. Появится сообщение об ошибке «В статус»Служебная" можно переводить учетные записи без назначенных ролей«. При успешной смене типа УЗ с»Основная" на «Служебная» тип одной случайной дополнительной УЗ также сменится на «Основная».

Для изменения типа УЗ пользователя администратору необходимо выполнить следующие действия:

1) Выбрать необходимую УЗР в разделе «Учетные записи ресурсов». Откроется карточка УЗР с параметрами.
2) Перейти в подраздел «Учетные записи». В списке УЗ выбрать одну запись с типом «Дополнительная» и нажать кнопку «Смена типа».

3) В окне смены типа УЗ выбрать один из доступных типов: «Основная», «Дополнительная», «Служебная». При выборе типа «Основная» появится предупреждение о смене текущей основной УЗ на дополнительную.
4) Нажать кнопку «Сохранить».

2.3.6. Смена парольной политики для УЗ в УЗР

УЗ пользователя в УЗР может быть связана с парольной политикой по умолчанию. В этом случае эта связь отображается в столбце «Название парольной политики». Связать УЗ пользователя с политикой можно только сменив политику по умолчанию на ту, которая была создана для этой УЗР. Если в Комплексе отсутствуют политики по умолчанию для УЗ, то столбец «Название парольной политики» содержит значение «Отсутсвует». Подробнее о парольной политике приведено в подразделе 2.11.

Для смены парольной политики УЗ администратору необходимо в меню «Админ.системы»/«Учетные записи ресурсов» выполнить следующие действия:

1) Выбрать УЗР из списка и перейти на вкладку «Учетные записи».
2) Выбрать УЗ, для которой нужно сменить парольную политику.
3) Нажать кнопку «Сменить парольную политику».
4) Выбрать из списка парольную политику. Список отображает парольные политики созданные для данной УЗР со следующими ограничениями:
   – нельзя выбрать парольную политику по умолчанию, так как она назначается автоматически;
   – нельзя выбрать уже назначенную парольную политику.
5) Нажать кнопку «Сохранить». Отобразится сообщение «Парольная политика УЗ успешно изменена».

Для возврата к парольной политике по умолчанию администратору необходимо выполнить следующие действия:

1) Выбрать УЗР из списка и перейти на вкладку «Учетные записи».
2) Выбрать УЗ.

3) Нажать кнопку «Отвязать парольную политику», чтобы удалить связку между УЗ и парольной политикой.
4) Нажать кнопку «Отвязать». Отобразится сообщение «Парольная политика успешно отвязана». После чего для УЗ сменится парольная политика назначенная вручную на парольную политику по умолчанию.

2.4. Управление периодическими заданиями

Периодические задания Комплекса предназначены для выполнения действий с УЗ пользователей и другими объектами Комплекса по расписанию.

Управление периодическими заданиями включает в себя следующие действия администратора:

- создание периодического задания;
- редактирование периодического задания;
- удаление периодического задания;
- запуск и остановка периодического задания.

Управление периодическими заданиями выполняется администратором двумя способами:

- с помощью Rest-запроса;
- в меню «Админ. системы»/«Периодические задания».

Экземпляры периодического задания создаются на основании шаблона (тип задания), содержащего специфичные для задания атрибуты. Любое поле периодического задания можно настроить, как поле типа «Поиск объекта» («lookup»), которое позволяет выполнять поиск атрибута. Для этого следует изменить шаблон и включить в него параметры согласно пункту 2.7.1.2.3.

Параметры периодического задания включают следующее:

- общие сведения о периодическом задании;
- атрибуты задания, определяемые его шаблоном.

Общие сведения о периодическом задании приведены в таблице 2.15.

Таблица 2.15 – Общие сведения о периодическом задании

Атрибут	Описание	Пример
Название (_id)	Идентификатор (имя) запланированной задачи. Обязательный к заполнению атрибут	AD User Recon
Описание (description)	Описание запланированной задачи	Синхронизация пользователей из AD
Включено (enabled)	Значение «true» активирует расписание задания. Когда это свойство имеет значение «false», Комплекс не запускается задание	true
Тип расписания (type)	В настоящее время Комплекс поддерживает только cron-формат. По умолчанию свойство имеет значение «cron»	cron
Дата начала (startDate)	Используется для планирования запуска запланированной задачи. Если этот параметр не задан, запланированная задача запускается по значению «cron» без ограничения по дате. Для указания даты и времени используется следующий формат: YYYY-MM-DD hh:mm:ss | –	
Дата окончания (endDate)	Используется для планирования окончания запланированной задачи. Задание выполняется по указанную дату включительно. Если этот параметр не задан, запланированная задача запускается по значению «cron» без ограничения по дате.	| –
Расписание задания (schedule)	Принимает синтаксис выражений cron. Задаваться может в двух режимах в меню «Админ. системы»/«Периодические задания»: – обычный режим, в котором указывается период повторений в единицах времени секунды, минуты, часы, дни, месяцы; – в формате cron, в котором можно задавать расширенные сценарии запуска заданий, которые также учитывают день недели, год. Расписание не отображается в обычном режиме, если ранее было задано в формате cron с расширенным сценарием запуска. Обязательный к заполнению атрибут	0/1 * * * * ?
Режим восстановления (misfirePolicy)	Для постоянных запланированных задач этот параметр задает их поведение, если запланированная задача была пропущена.	fireAndProceed

Атрибут	Описание	Пример
	Возможные значения: – «Продолжение выполнения» («fireAndProceed») – запуск упавшего задания сразу же после запуска Комплекса; – «Запуск нового задания» («doNothing») – запуск нового задания по расписанию. По умолчанию принимает значение «doNothing»	
Конкурентное выполнение (concurrentExecution)	Указывает, могут ли несколько экземпляров одной и той же запланированной задачи выполняться одновременно. По умолчанию принимает значение «false». Несколько экземпляров одной и той же запланированной задачи не могут выполняться одновременно по умолчанию. При этом не допускается, чтобы новая запланированная задача была запущена прежде, чем такая же запущенная задача была завершена.Чтобы запустить несколько запланированных задач одновременно, следует установить параметр в значение «true»	false
Тип задания(invokeService)	Определяет используемый шаблон запланированного задания. Описание типовых заданий приведено в пункте 2.4.5. Обязательный к заполнению атрибут	recontask
invokeContext	Параметры, определяемые типом задания (значение параметра «invokeService»)	–
Уровень логгирования(invokeLogLevel)	Определяет уровень детализации сведений о работе задания в системных журналах Комплекса. По умолчанию уровень журнала имеет значение «info». Параметр может быть установлен в следующие значения: – trace; – debug; – info; – warn; – error; – fatal	info
timeZone	Если не установлен, Комплекс использует часовой пояс системы	–
persisted	Значение «true» является признаком сохранения задания в базе данных (БД). По умолчанию свойство имеет значение «true»	true

2.4.1. Создание периодического задания

Для периодического задания в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Периодические задания», затем нажать кнопку «Создать». Появится страница создания периодического задания.
2) Заполнить обязательные поля «Название» и «Расписание задания».
3) Расписание задания может задаваться в двух режимах: с указанием частоты повторов и единиц измерения времени, или в расширенном формате cron. Для выбора режима необходимо нажать кнопку «cron».
4) При выборе «Тип задания» отображаются дополнительные параметры задания, определяемые выбранным типом.
5) Включить задание можно отметив параметр «Активность задания».
6) Нажать кнопку «Сохранить». Чтобы увидеть созданное задание, в списке следует нажать кнопку «Обновить».

2.4.2. Редактирование периодического задания

При редактировании периодического задания выполняются следующие условия:

1) Недоступно для редактирования названия периодического задания.
2) Недоступно для редактирования значение атрибута «Тип задания». Для использования другого типа задания необходимо создавать новое периодическое задание.
3) Выполненные изменения применяться при очередном запуске задания.

Для редактирования ресурса в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Периодические задания» и выбрать из списка заданий необходимое. Откроется карточка периодического задания с параметрами.
2) Изменить параметры периодического задания и нажать кнопку «Сохранить».

2.4.3. Удаление периодического задания

При удалении периодического задания выполняются следующие условия:

1) Удаление задания не прерывает его работу, если он выполняется, а только удаляет последующие запуски этого задания.

Для удаления периодического задания в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Периодические задания» и выбрать из списка одно или несколько периодических заданий. Откроется окно подтверждения с перечнем удаляемых заданий.
2) Нажать кнопку «Удалить».

2.4.4. Запуск и остановка периодического задания

Запуск периодического задания выполняется только для остановленных/отключенных заданий. Для задач в статусе «выполняется» функция недоступна.

Комплекс поддерживает пакетный запуск заданий.

Для запуска периодического задания в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Периодические задания» и выбрать из списка одно или несколько остановленных/отключенных заданий.
2) Нажать кнопку . Задание будет запущено в течении некоторого времени.

При остановке периодического задания выполняются следующие условия:

1) Остановка периодического задания доступна только для заданий, которые в настоящий момент имеют статус «Выполняется».
2) Выполнение действия остановки должно обеспечиваться периодическим заданием, в противном случае задание будет продолжать выполнение. Все задания, включенные в состав Комплекса, поддерживают остановку.
3) Для периодических заданий, имеющих поддержку функции остановки. Сама остановка может происходить не мгновенно, а по прошествии

какого-то времени, необходимого для завершения минимальных изменений уже начатых на момент остановки.

Для остановки периодического задания в меню «Админ. системы» администратору необходимо выполнить следующие действия:

1) Перейти в раздел «Периодические задания» и выбрать из списка одно выполняемое задание.
2) Нажать кнопку  . Задание будет приостановлено в течении некоторого времени.

2.4.5. Описание периодических заданий Комплекса

Перечень типовых периодических заданий Комплекса:

1) Тип задания «evaluatetask» – задание для пересчета привилегий, выделенных пользователю через роль. Параметр задания «batch» – количество одновременно пересчитываемых ролей (например, 500). При назначении, отзыве или изменении роли у пользователя данные помечаются, как требующие пересчета привилегий. Задание просматривает эти данные и пересчитывает состав привилегий, при этом добавляя и/или удаляя их. Период запуска по умолчанию – 1 минута. Правила пересчета привилегий:
   - если назначение «Роль» и у пользователя нет УЗ, то ему выделяется УЗ с привилегиями выделяемой роли;
   - если назначение «Роль», уже выделена УЗ, и у УЗ нет каких-то привилегий, которая дает данные роль, то к УЗ добавляются данные привилегии;
   - если у привилегии типа назначения «Роль», то при отзыве роли, у УЗ удаляются привилегии, которые назначались данной ролью (при условии, что нет других ролей, которые бы давали данные привилегии);
   - если выделена роль и у УЗ есть привилегии с типом назначения не «Роль», то при выполнении задания у тех привилегий, которые даются ролью, тип назначения изменится на «Роль»;

- если «Дата начала» сегодня или в прошлом, а «Дата окончания» в будущем, то роль применяется к пользователю (выделяется или обновляется УЗ);
- если «Дата начала» в будущем, то роль не применяется к пользователю. Применение произойдет по наступлении «Дата начала»;
- если «Дата окончания» в прошлом, то привилегии, выделяемые по роли отзываются;
- если «Дата окончания» сегодня или в будущем, то привилегии, выделяемые по роли не отзываются. Отзыв произойдет по наступлении «Даты окончания».

2) Тип задания «reindextask» – задание для построения/обновления индекса объектов системы для поиска. Сейчас индекс создается только при создании нового объекта в Комплексе и обновлении существующего. Если в системе уже есть объекты, то для них индекс не создается. В случае сбоя/недоступности системы поиска, при создании нового объекта индекс также не будет создан, и данные, полученные через полнотекстовый поиск, будут неактуальными. Параметры задания отсутствуют.
3) Тип задания «failedsynctask» – задание для операций выделение/обновления/удаления УЗ в ЦС, ранее завершенные с ошибкой. При повторе операции обновления УЗ, если в форме ресурса определено поле пароль, то оно не будет изменено. Параметры задания отсутствуют.
4) Тип задания «workflowrequesttask» – задания для исполнения согласованных заявок. Если заявка была успешно согласована на всех этапах, то задание выполняет действия из данного типа заявки (пример, назначает пользователю ранее запрошенные роли). В случае успешного выполнения заявка переводится в статус «Исполнена». Параметр задания «batch» - количество одновременно обрабатываемых заявок (например, 500).

5) Тип задания «clearpasswordhistorytask» – задание для очищения истории паролей от неиспользуемых данных, относящихся к тем ресурсам, для которых нет в конфигурации парольной политики правила контроля истории паролей. Чтобы задание отработало, необходимо отключить историю паролей, убрав параметр «passwordHistoryCount» в файле passwordpolicy.json. Параметры задания отсутствуют.
6) Тип задания «recontask» – задание для синхронизации УЗ из ЦС или доверенного источника в режиме получения всех записей (полная синхронизация). Параметр задания «resAccName» – имя УЗР ЦС, для которой выполняется синхронизация УЗ (например, XMLResAccount).
7) Тип задания «livesynctask» – задание для синхронизации УЗ из ЦС или доверенного источника в режиме получения последних измененных записей (инкрементальная синхронизация). Параметр задания «resAccName» – имя УЗР ЦС, для которой выполняется синхронизация УЗ (например, XMLResAccount).
8) Тип задания «lookuprecon» – задание для синхронизации данных в справочники из ЦС или доверенного источника. Например, применяется для синхронизации справочника полномочий (групп, ролей, прав доступа) из ЦС. Параметры задания:
   – «resName» – имя ресурса, из которого выполняется синхронизация справочника. Например, XMLResource;
   – «connectorObjectType» – тип объекта коннектора данные которого будут синхронизироваться в справочник. Например, group;
   – «refBookType» – наименование справочника, в который будут синхронизироваться данные. Например, XML Groups;
   – «refBookCode» – атрибут объекта ЦС значение, которого будет записано в виде кода в справочник. Может содержать два значения: source – название поля объекта из ресурса и transform – правило, по которому надо преобразовать значение перед записью в справочник.

Например, {"source" : "id"};

– «refBookName» – атрибут объекта ЦС значение, которого будет записано в виде наименования в справочник. Может содержать два значения: source – название поля объекта из ресурса и transform – правило, по которому надо преобразовать значение перед записью в справочник. Например, {"source" : "groupName"};

– «deleteIfUnassigned» – признак удаления неактуальных справочных данных при синхронизации. Например, true. Неактуальными считаются те справочники, у которых имя ресурса («resName») и тип справочника («refBookType») отличаются от соответствующих параметров в периодическом задании.

Для того чтобы ограничить выборку данных из систем только необходимыми объектами или же заставить систему вернуть все поля объектов, следует указывать запросы для source/target систем. Для этого используются запросы sourceQueryFullEntry, targetQueryFullEntry, где source/targetQueryFullEntry - параметры для источника (ЦС) и получателя (Комплекс) соответственно:

– sourceQueryFullEntry - возможные значения true/false;

– targetQueryFullEntry - возможные значения true/false.

Если параметр принимает значение false, то Комплекс всегда считает полученный объект неполным и данный объект будет заново запрашиваться из системы, при необходимости. Если параметр принимает значение true, то Комплекс всегда считает полученный объект полным и данный объект будет всегда считываться из памяти. Для source/target систем можно задавать различные query-запросы. Из доступных по умолчанию:

– query-all: для получения всех объектов системы со всеми атрибутами одним запросом;

– query-all-ids: возвращает только идентификаторы объектов.
Если задать sourceQueryFullEntry = true, не указывая sourceQuery, то выполняется только запрос query-all-ids, а эти данные будут считаться полным объектом и никакие дополнительные запросы не будут отправляться.

9) Тип задания «warnbeforeexpiredaccountstask» – задание по уведомлению пользователя о просроченном пароле его УЗ ЦС в день просрочки и за несколько дней до момента наступления события. Параметры задания:
   – «warnBeforeExpired (days)» – количество дней до момента наступления события, когда будет направлено предупреждение о предстоящей просрочке времени действия пароля;
   – «emailSubject» – тема уведомления (например, «Требуется сменить пароль ваших УЗ»);
   – «emailBodyBeforeExpired» – текст уведомления, направляемого за несколько дней до момента наступления события (например, Ваш пароль для УЗ ${resAccName} устареет через ${expireDays} дн..);
   – «emailBodyExpired» – текст уведомления, направляемого в день просрочки (например, Ваш пароль для УЗ ${resAccName} устарел.).
10) Тип задания «disableinactiveaccountstask» – задание для блокировки УЗ пользователей в ЦС, если пользователь не входил в ЦС несколько дней. Параметры задания:
   – «resFormName» – имя формы ресурса ЦС, в которой присутствует атрибут «Дата последнего входа» (lastLogonDate). Например, ADResForm;
   – «inactivePeriod (days)» – период неактивности пользователя в днях. Задача блокирует учётную запись в заданной форме ресурса, если:

– статус УЗ active («Активна»);
– тип УЗ (accountType): primary (Основная) или other (дополнительная); «Дата последнего входа» (lastLogonDate) заполнена и с этой даты прошло не меньше заданного в задаче числа дней (inactivePeriod (days)). Для использования задания необходимо, чтобы ЦС хранила дату последнего входа/использования УЗ. Должна быть настроена синхронизация даты последнего входа/использования УЗ в служебный атрибут lastLogonDate (в формате yyyy-MM-dd). После разблокировки УЗ (например, администратором вручную или по заявке) пользователь должен осуществить под ней вход в ЦС до следующего запуска задания, иначе УЗ снова заблокируется.

11) Тип задания «rolemeningtask» – задача анализирует полномочия, назначенные пользователям. В результате анализа формируются предложения (отчет) по составу и содержанию новых ролей, с включенными в них полномочиями. Параметры задания:
– «miningAlgorith» – алгоритм анализа ролевой модели. По умолчанию поддерживается только алгоритм «HPRoleMinimization» (например, HPRoleMinimization);
– «wd», «wu», «wr», «wp», «wh» – весовые коэффициенты для использования алгоритмом. По умолчанию принимают значение 1.

12) Тип задания «usercertificationtask» – задание для формирования заявок на сертификацию ролей подчиненных сотрудников и направление заявок линейным руководителям. Для работы задания должен быть настроен рабочий поток «certificateUserAppRoles» для действия «verify» на объект «endpoint/user-certification». Параметры задания:
– «ifManagerNotDefined» – действие в случае, если линейный руководитель не найден. Может принимать значения: «do-nothing» и «assign-to-default-role». Если задано «do-nothing», то при отсутствии руководителя у пользователя, заявка на сертификацию не будет создаваться. Если задано «assign-to-default-role», то при отсутствии

руководителя у пользователя, заявка на сертификацию будет назначаться на роль, указанную в параметре задания «defaultRole»;

– «defaultRole» – роль по умолчанию, на которую будут назначаться заявки на сертификацию, в случае отсутствия линейного руководителя.

13) Тип задания «approlecertificationtask» – задание для формирования заявок на сертификацию настроек ролей определенной информационной системы (ИС) и направление заявок участникам роли сертификатора, заданной в ИС. Для работы задания должен быть настроен рабочий поток «certificateAppRoles» для действия «verify» на объект "endpoint/approle-certification/*". Параметр задания «targetInfoSystem» – имя ИС для ролей, которой выполняется сертификация. По умолчанию принимает значение «default-is», для которого сертификация не выполняется.

14) Тип задания «sodtask» – задание анализирует роли, для которых есть разрешение на запрос хотя бы у одной организации, и правила разграничения доступа. Нарушение выявляется при следующих условиях:

– роль назначена пользователю, при этом пользователь находится в другом подразделении, чем роль;

– полномочия роли назначены пользователю напрямую (в обход Комплекса), при этом пользователь находится в другом подразделении, чем роль;

– роль назначена пользователю, при этом не выполнены условия правила разграничения доступа;

– полномочия роли назначены пользователю напрямую (в обход Комплекса), при этом не выполнены условия правила разграничения доступа. По каждому выявленному неправомерному назначению создаётся инцидент с информацией о нарушении. Если в системе уже существует данный инцидент, он не будет создан повторно.

15) Тип задания «sodWorkflowTask» – задание анализирует новые инциденты, для которых ещё не были назначены на рассмотрение. Новый инцидент назначается на участников роли оператора SoD, задаваемой в атрибутах ИС, или в правиле разграничения доступа. В результате работы задания новые инциденты должны поступить на рассмотрение во входящие инциденты для операторов SoD. В процессе выполнения задания участникам роли оператор SoD направляется уведомление. Описание настроек уведомления о новом инциденте приведено в пункте 2.12.5.
16) Тип задания «email.request.processing.task» – задача читает почтовый ящик, указанный в файле конфигурации email.request.processing.json, и выполняет требуемые действия с заявкой. Если действие успешно выполнено, письмо удаляется.
17) Тип задания «approlesbyrulemessagetask» – задача назначает или отзывает роли пользователю по установленному для роли правилу автоназначения. Параметр задания «batch» – количество одновременно обрабатываемых задач по автоназначению ролей.
18) Тип задания «deferredchangestatustask» – задача изменяет статус (активация и/или блокировка) пользователя в зависимости от наступления даты:
    – «Дата принудительной активации»;
    – «Дата принудительной блокировки». Запуск осуществляется на первой секунде каждых суток.
19) Тип задания «incidentActualizationTask» – задание актуализирует название инцидента (после изменения имени роли или правила, на базе которого создан инцидент). В интерфейсе отображается сохраненной в кэш значение.
20) Тип задания «scriptexecution» – задача позволяет запускать скрипты JavaScript. Параметры задания:
    – «file» – путь к файлу скрипта;

– «input» – параметры, передаваемые скрипту (например, "first": 2, "second": 3);
– «source» – код скрипта, в случае если не используется файл (например, java.lang.System.out.println(“Test input:” + input.first + ’ ’ + input.second););
– «type» – тип скрипта, поддерживается только «text/javascript» (например, text/javascript).

2.4.6. Настройка службы запуска периодических заданий

Настройка службы запуска периодических заданий выполняется в файле conf/scheduler.json:

```
{
   "threadPool" : {
      "threadCount" : "10"
   },
   "scheduler" : {
      "executePersistentSchedules" : "&{openidm.scheduler.execute.persistent.schedules}"
   }
}
```

Настройка службы запуска периодических заданий включает следующие параметры:

– «threadCount» – количество используемых параллельных потоков;
– «executePersistentSchedules» – параметр определяющий возможность запуска периодических заданий. Может принимать значения «true» или «false» (автоматический запуск отключен). Значение параметра по умолчанию берется из файла конфигурации boot.properties.

2.4.7. Описание настроек расписания в формате cron

Расписание периодического задания задается в формате выражения cron состоит из следующих полей:

1) Секунды (0-59).
2) Минуты (0-59).
3) Часы (0-23).
4) День месяца (1-31).
5) Месяц (1-12 или JAN-DEC).

6) День недели (1-7 или MON-SUN).
7) Год (текущий-2099).

Поля могут содержать следующие специальные символы:

- символ косая черта «/» обозначает приращение значения. Например, «5/15» в поле «секунды» означает каждые 15 секунд, начиная с пятой секунды;
- знак вопроса «?» означает, что в поле не должно быть указанной величины. Таким образом, если вы устанавливаете день недели, вы можете вставить «?» в поле «день недели» для обозначения того, что значение «день недели» несущественно;
- знак астериска «*» обозначает, что любое возможное значение может быть принято для данного отдельного поля;
- знак дефиса «–» обозначает интервал значений.

Примеры:

1) 0 */15 * * * ? (запуск каждые 15 минут).
2) 0 0 0 */1 * ? (запуск каждый день).
3) 0 0 0 1 3 ? (запуск 01 марта в 00:00).
4) 0 0 11-13 * * ? (запуск с 11:00 по 13:00 часов).
5) 0 30 10 ? * * (запуск в 10:30 каждый день).

2.4.8. Просмотр списка периодических заданий

В списке задач в меню «Периодические задания» отображаются все периодические задачи в системе. Список отсортирован по столбцу «Название».

В списке отображаются следующие атрибуты периодической задачи:

1) «Название» – отображается имя задачи.
2) «Последний запуск» – отображается период времени и статус, с которым задача выполнялась в последний раз.
3) Статусы задач:
   - «Не выполнялась» – означает, что задача ни разу не запускалась;
   - «Успешно (<время с последнего выполнения>)» – означает, что задача была успешно выполнена;

– «Ошибка (<время с последнего выполнения>)» – означает, что задача завершена с ошибкой, а на вкладке «Информация» в блоке «Ошибка» выводится сообщение об ошибке;
– «Выполняется» – означает, что задача выполняется непосредственно в текущий момент просмотра.

4) «Включена» или «Выключена» – показывает состояние активности задания.

Возможные значения переменной <время с последнего выполнения>:

– ru:«Менее минуты назад»/en«Less minute ago» – означает, что последнее выполнение задачи производилось менее 60 секунд назад;
– <количество минут числом> ru:" мин. назад«/en:» min. ago" – означает, что последнее выполнение задачи производилось менее 60 минут назад, например: «43 мин. назад»/en:«43 min. ago»;
– <количество часов назад числом> ru:" ч. назад«/en:» hr. ago" – означает, что последнее выполнение задачи производилось менее 24 часов назад;
– <количество дней назад числом> ru:" дн. назад«/en:»d. ago" – означает, что последнее выполнение задачи производилось менее 30 дней назад;
– <количество месяцев назад числом> ru:«мес. назад»/en:«mo. ago» – означает, что последнее выполнение задачи производилось не более 12 месяцев назад;
– ru:«Больше года назад»/en:«Over year ago» – означает, что последнее выполнение задачи производилось более 12 месяцев назад.

Округление в меньшую сторону производится при < = 0.5 единицы измерения, а при > 0.5 единицы измерения округление производится в большую сторону.

2.5. Настройка справочников

Справочник (referencebook) – это набор значений с атрибутами определенного типа, поступающими из ЦС при синхронизации справочников. Используется при создании полей с предопределенным набором значений.

В меню «Админ. системы»/ «Справочники» администратор может выполнить следующие действия:

1) Создание справочника.
2) Удаление справочника.
3) Редактирование справочника.
4) Экспорт/импорт значений справочника из Excel.

Список справочников отсортирован по имени ресурса и содержит следующие данные:

– «Ресурс» – имя ресурса («resName») ЦС, для которой создан справочник.
– «Название справочника» – тип справочника («refBookType»).

Для одного ресурса в списке может быть указано несколько типов справочников, а один и тот же тип справочника может быть указан в нескольких ресурсах.

Детали справочников содержат имя и код. Администратор может производить поиск по деталям и по группе справочников. Поиск по деталям производится по коду и по имени. Поиск по группе справочников производится по ресурсу, по типу, по коду, по имени.

2.5.1. Создание справочника

Для создания справочника следует выполнить следующие действия:

1) Нажать кнопку «Создать». Откроется окно «Создание справочника».
2) Заполнить обязательные поля «Название справочника», «Ресурс», «Код/Имя».
3) Нажать кнопку «Создать». Появляется сообщение «Справочник успешно создан».

При заполнении полей следует учитывать следующие особенности:

1) Название справочника на один и тот же ресурс должно быть уникальным. В противном случае отобразится ошибка «Для выбранного ресурса уже существует справочник с таким названием».

2) Значение в поле «Код» должно быть уникальным. В противном случае отобразится ошибка «Такое значение уже существует».
3) Кнопка «Создать» не будет доступна пока все поля не заполнены корректно.

2.5.2. Удаление справочника

Для удаления справочника следует выполнить следующие действия:

1) Нажать кнопку «Удалить». Откроется окно «Удаление справочника».
2) Нажать кнопку «Удалить» для подтверждения. После удаления появится сообщение «Справочник успешно удален».

При удалении справочника удаляются все его значения.

2.5.3. Редактирование значений справочника

Редактирование значений справочника выполняется в правой части экрана на вкладке «Значения».

С помощью кнопки «Добавить записи» выполняется добавление нового значения. При добавлении записей в окне «Добавление записей» не отображаются существующие значения.

С помощью кнопки «Редактировать записи» выполняется редактирование полей «Код» и «Имя».

При изменении полей следует учитывать особенности их заполнения, которые приведены в пункте 2.5.1.

Для удаления записи следует отметить ее флагом и нажать кнопку «Удалить запись». Можно удалять несколько записей одновременно.

Строка поиска может быть использована как для проверки и редактирования значений, так и для их удаления. Результат поиска остается в списке «Выбранное» после сброса результатов поиска и его можно удалить в любой момент.

2.5.4. Экспорт/импорт значений справочника из Excel

2.5.4.1. Экспорт значений справочника из Excel

Для экспорта значений справочника необходимо в подразделе «Справочники» выполнить следующие действия:

1) Нажать на кнопку «Экспорт». Откроется окно «Экспорт справочников».
2) Ввести название ресурса в поле «Ресурс».
3) Выбрать при необходимости «Название справочника» для экспорта одного справочника. Для экспорта всех справочников следует оставить поле пустым.
4) Выбрать формат сохранения xls/xlsx.
5) Нажать на кнопку «Сохранить». Начнется стандартная загрузка файла. Отобразится сообщение «Справочники успешно экспортированы».

Файл экспорта содержит следующие столбцы:

- _rev – количество изменений записи;
- resource_id – id-ресурса;
- refBookType – название справочника;
- _id – id-записи справочника;
- refBookCode – код записи справочника;
- _ouid – числовой id-записи справочника;
- refBookName – имя записи справочника;
- status – статус записи «active»/«deleted».

Файл экспорта поддерживает пользовательские атрибуты (UDF-поля).

2.5.4.2. Импорт значений справочника из Excel

Для импорта значений справочника необходимо в подразделе «Справочники» выполнить следующие действия:

1) Нажать на кнопку «Импорт». Откроется окно «Импорт справочников».
2) Ввести название ресурса в поле «Ресурс».
3) Нажать кнопку «Добавить файл».
4) Нажать на кнопку «Импортировать». Отобразится сообщение «Файл загружен без ошибок».

После загрузки файла отображается статистика:

1) Новые справочники.

2) Обновленные справочники.

3) Удалённые справочники.

Нулевая статистика отображается при импорте файла, в который не было внесено изменений. При этом кнопка «Импортировать» недоступна.

На загрузку файла действуют следующие ограничения:

- можно импортировать только один файл;
- доступный тип файла: xls/xlsx.

Обязательные столбцы при импорте справочников:

- resource_id – id-ресурса;
- refBookType – название справочника;
- _id – id-записи справочника;
- refBookCode – код записи справочника;
- refBookName – имя записи справочника;
- status – статус записи «active»/«deleted».

При подготовке файла импорта следует учитывать следующее:

- при создании и редактировании записи справочника следует указывать значение «active» в столбце «status»;
- при удалении записи справочника необходимо указывать значение «deleted» в столбце «status».
- новая запись не должна содержать значение в столбце _id.

Файл импорта поддерживает пользовательские атрибуты (UDF-поля).

Перечень возможных предупреждений при импорте из-за ошибок в импортируемом файле:

1) Отсутствует лист «Справочники».

2) Требуемый заголовок отсутствует в файле – отсутствует обязательный столбец.

3) Не заполнены обязательные поля в файле.

4) Ошибка идентификатора справочника – файл содержит неверный id-записи справочника.

5) Ошибка идентификатора ресурса – значение «resource_id» в файле не совпадает с id-ресурса, указанного в поле «Ресурс» при импорте.

6) Дублирование записи в справочнике – файл содержит новые записи справочника с одинаковым значением в столбце refBookCode и разными значениями в столбце refBookName.
7) Справочник уже существует – файл содержит новую запись справочника, которая уже существует.
8) Неверный тип файла.
9) Неверный формат файла.
10) Ошибка в типе значения.

2.6. Настройка рабочих потоков

Рабочий поток (workflow) – это последовательность действий (бизнес-процессов, подпроцессов) прохождения электронной заявки.

Комплекс позволяет работать с рабочими потоками с помощью модуля «Activiti», с поддержкой стандарта BPMN 2.0. Более подробная информация о модуле «Activiti» изложена на сайте «http://www.activiti.org/».

В настоящем подразделе описаны действия, выполняемые администратором при подключении, отключении, создании и настройке рабочих потоков, а также дополнительные настройки для электронной заявки.

Комплекс поддерживает запуск рабочего потока для следующих действий:

1) Сертификация пользователей подразделения.
2) Сертификация ролей ИС.
3) Создание роли.
4) Удаление роли.
5) Изменение роли.
6) Отзыв роли у пользователя.
7) Сброс пароля пользователю администратором или оператором.
8) Активация пользователя.
9) Удаление пользователя.
10) Назначение роли пользователю.
11) Изменение пользователем собственных атрибутов.
12) Активация УЗ пользователя.

13) Создание пользователя.
14) Редактирование пользователя администратором или оператором.
15) Блокировка пользователя.
16) Блокировка УЗ пользователя.
17) Сброс пароля УЗ пользователя.
18) Удаление УЗ пользователя.
19) Создание подразделения.
20) Удаление подразделения.
21) Редактирование подразделения.
22) Изменение срока действия роли.
23) Создание записи делегирования.

При создании заявки происходит поиск подходящего бизнес-процесса согласования в следующей последовательности:

1) По заданным сущности, действию и ИС ищется настройка в объекте «managed/workflowsetting».
2) Если настройка найдена:
   – в случае заполненного поля workflowKey, используется указанный процесс согласования;
   – в случае незаполненного поля workflowKey, действие над объектом выполняется сразу, без запуска процесса согласования.
3) Если настройка не найдена, по заданным сущности и действию ищется настройка в файле конфигурации request.json:
   – в случае наличия настройки, используется указанный процесс согласования;
   – в случае отсутствия настройки, действие над объектом выполняется сразу, без запуска процесса согласования.

Комплекс поддерживает следующие основные возможности рабочих потоков «Activiti»:

1) Последовательное и параллельное согласования заявки.
2) Отображение и валидация полей формы заявки, настроенной на разных этапах согласования.

3) События, выполняемые по таймеру (например, Timer Boundary Event).
4) Уведомления на электронную почту участников согласования.

Комплекс также поддерживает возможность вложения файлов в электронную заявку.

Для просмотра списка бизнес-процессов Комплекса администратору следует перейти в меню «Админ. системы»/ «Бизнес-процессы».

Для создания бизнес-процесса в меню «Админ. системы»/ «Бизнес-процессы» администратору следует выполнить следующие действия:

1) Нажать кнопку «Создать». Откроется визуальный конструктор, который в центре окна отображает бизнес-процесс в виде графической схемы.
2) Заполнить уникальное значение параметра «Идентификатор» на вкладке «Основные». По умолчанию параметр «Идентификатор» принимает значение «Process_1» и не может быть отредактирован после сохранения бизнес-процесса. Если введено не уникальное значение, при сохранении отображается сообщение «Бизнес-процесс с таким идентификатором уже существует».
3) Выполнить действия по созданию схемы бизнес-процесса.
4) Нажать кнопку «Сохранить».

Для удаления бизнес-процесса в меню «Админ. системы»/ «Бизнес-процессы» администратору необходимо выполнить следующие действия:

1) Выбрать нужный бизнес-процесс из списка.
2) Нажать на кнопку «Удалить». Откроется окно подтверждения «Удаление бизнес-процесса».
3) Нажать на кнопку «Удалить» в окне подтверждения. В случае успешного удаления отобразится сообщение «Бизнес-процесс успешно удален». В случае ошибки отображается сообщение «Ошибка при удалении бизнес-процесса». Для отмены действия следует нажать кнопку «Отменить».

Типовые ошибки при попытке удаления бизнес-процесса: * бизнес-процесс связан по крайней мере с одной информационной системой; * в системе присутствуют незавершенные заявки (в процессе согласования), связанные с удаляемым бизнес-процессом.

Для редактирования бизнес-процесса в меню «Админ. системы»/ «Бизнес-процессы» администратору следует выбрать бизнес-процесс из списка и нажать кнопку «Редактировать». Откроется визуальный конструктор.

Графическая схема бизнес-процесса ориентирована в направлении слева направо.

Панель с инструментами и элементами графической схемы расположена в левой части экрана. Информация о каждом элементе бизнес-процесса отображается в правой части окна конструктора. Содержимое этой области меняется в зависимости от того, какой элемент был выбран.

Графические элементы определяют ход бизнес-процесса и делятся на следующие категории: * события; * действия; * шлюзы.

Чтобы установить новый элемент бизнес-процесса, следует выбрать элемент на панели и нажать на пустую область конструктора. После чего элемент будет установлен. Элемент можно передвигать по области конструктора.

Некоторые элементы имеют несколько типов. Для того чтобы выбрать нужный тип элемента, необходимо выделить элемент на схеме и нажать на инструмент «Изменить тип». Отобразится выпадающее меню для выбора типа.

Основные элементы конструктора приведены в таблице 2.16.

Таблица 2.16 – Основные элементы конструктора

Наименование элемента	**Описание**
Создать начальное событие	Событие, обозначающее начало процесса
Создать промежуточное/граничное событие	Промежуточное событие. Промежуточные события могут использоваться в качестве граничных событий для задач. В этом случае они могут быть прерывающими или непрерывными
Создать конечное событие	Событие, обозначающее конец процесса
Шлюз	Шлюз, который определяет вариант потока операций (эксклюзивный, параллельный, на основе событий и другие)
Создать задачу	Задача, обозначающее простое действие. Внутри элемента ввиде блока помещается наименование процесса

Наименование элемента	Описание
Создать развернутый подпроцесс	Несколько задач, выделенные в отдельную подзадачу или подпроцесс
Создать объект данных	Объект данных, представляющий информацию о том, какие действия необходимо выполнить или результат этих действий
Создать хранилище данных	Хранилище данных
Создать пул/участника	Пул, отображающий исполнителей (задач, организаций, пользователей). Используется для разграничения ответственности
Создать группу	Группа – элемент для группировки графических элементов
Добавить TextAnnotation	Примечание. Используется для добавления пояснения к элементу
Удалить	Используется для удаления элемента

Основные инструменты конструктора приведены в таблице 2.17.

Таблица 2.17 – Основные инструменты конструктора

Наименование инструмента	Описание
Использовать инструмент «Рука»	Инструмент, позволяющий захватить элемент схемы (всю схему) и передвигать по полю конструктора
Использовать инструмент «Лассо»	Инструмент, позволяющий выделить элемент схемы (всю схему)
Использовать инструмент «Сократить/Растянуть»	Инструмент, позволяющий позволяет «раздвинуть» или «сжать» схему
Использовать инструмент «Потоки и ассоциации»	Инструмент, представляющий собой соединяющие элементы

Комплекс поддерживает создание формы заявки заявителя и согласующего с помощью конструктора.

Для создания формы заявки заявителя следует выполнить следующие действия:

1) В конструкторе выбрать элемент «Создать начальное событие».
2) Перейти на вкладку «Формы» в правой части окна конструктора. Вкладка «Формы» содержит следующие поля:
   – ключ формы (необязательное поле);
   – поля формы – для создания полей формы заявки.
3) Нажать на кнопку «+» для создания нового поля формы заявки. Отобразятся следующие поля:

– «Идентификатор» – идентификатор поля;
– «Тип» – тип поля. Выбор доступен из выпадающего списка: «string», «long», «boolean», «date», «enum», «custom type» (только значение «text»);
– «Ярлык» – отображаемое имя поля в форме заявки;
– «Значение»по умолчанию"" – значение, которое автоматически будет отображаться в поле формы заявки;
– «Свойства» – свойство поля. Для добавления свойства нажать на кнопку «+» и заполнить поля «Идентификатор» и «Значение». У поля с типом «enum» идентификатор может начинаться только с латинских букв или символа нижнего подчеркивания, может содержать цифры.

Список свойств поля формы заявки приведены в таблице 2.18.

Таблица 2.18 – Список свойств поля формы заявки

Идентификатор	Описание	Значение
required	Признак обязательности поля	true/false
writable	Признак имеется ли возможность заполнить	true/false
readable	Признак отображается ли поле в форме	true/false
datePattern	Позволяет задать «маску» для поля с типом дата	например, dd-MM-yyyy

Для создания формы заявки согласующего следует выполнить следующие действия:

1) В конструкторе выбрать элемент «Создать задачу».
2) Нажать «Изменить тип» и выбрать «Задача, выполняемая пользователем».
3) Выполнить аналогичные действия приведенные для формы заявки заявителя в пунктах 2-3.

2.6.1. Настройка модуля «Activiti»

Модуль «Activiti» настраивается в файле, расположенном в каталоге ankey/conf/workflow.json. Если этот файл отсутствует в конфигурации, модуль

рабочего потока недоступен для использования. По умолчанию в Комплексе файл workflow.json имеет следующую базовую конфигурацию:

```
{
   "enabled" : true,
   "workflowDirectory" : "&{launcher.project.location}/workflow"
}
```

Перечень атрибутов в файле workflow.json:

- «enabled» – по умолчанию модуль рабочего потока находится во включенном состоянии «enabled: true». Администратор может отключить модуль, указав в параметре значение «false»;
- «workflowDirectory» – каталог, в котором размещаются настроенные рабочие потоки. По умолчанию Комплекс считывает настройки рабочих потоков в каталоге «ankey/workflow».

2.6.2. Настройка уровня истории модуля «Activiti»

Уровень истории модуля «Activiti» определяет объем исторической информации, которая сохраняется во время выполнения рабочего потока. Для настройки уровня истории администратору следует указать значение параметра «history» в файле workflow.json, например: "history" : "audit".

Могут быть настроены следующие уровни истории:

- «none» – архивирование истории не производится. Этот уровень обеспечивает наилучшую производительность выполнения рабочего потока, но не позволяет сохранять историю выполнения;
- «activity» – архивирование всех экземпляров рабочего потока и экземпляров «Activiti». Детали выполнения рабочего потока не архивируются;
- «audit» – это значение по умолчанию. Все экземпляры рабочего потока, экземпляры «Activiti» и представленные свойства форм архивируются, так что все взаимодействие с пользователем посредством форм прослеживается и может быть проверено;

– «full» – это самый высокий уровень архивирования истории, оказывающий наибольшее влияние на производительность. На этом уровне сохраняется вся информация, которая хранится на уровне «audit», а также значения любой переменной, используемой в рабочем потоке.

2.6.3. Настройка рабочего потока для заявки на назначение роли определенной информационной системы

Настройка рабочего потока для заявки на назначение роли определенной ИС может выполняться администратором Комплекса следующими средствами:

– в меню «Инф. системы» для связывания рабочего потока с ИС, описание приведено в руководстве пользователя в разделе 2.7;
– REST-запросом для объекта «managed/workflowsetting».

Параметры REST-запроса для объекта «managed/workflowsetting» приведены в таблице 2.19.

Таблица 2.19 – Параметры REST-запроса для объекта «managed/workflowsetting»

Атрибут	Описание	Обязательный	Поддерживаемые значения	Пример
Сущность (entity)	Объект системы	Да	Объект системы «managed/usrapprole»	managed/usrapprole
Действие (action)	Действие над объектом	Да	create, batchCreate, delete, batchDelete, patch, enable	batchCreate
Информационная система (is_id)	Идентификатор ИС	Нет	Идентификатор существующей ИС	2
Процесс согласования (workflowKey)	Идентификатор процесса согласования. В случае отсутствия значения, действие над объектом выполняется без согласования	Нет	Идентификатор существующего в системе процесса согласования	autoApproval

Пример содержимого REST-запроса для объекта «managed/workflowsetting»:

```
Method: POST
URL: ankey/managed/workflowsetting?_action=create
```

```
Body:
{
  "action": "batchCreate",
  "entity": "managed/usrapprole",
  "is_id": 2,
  "workflowKey": "autoApproval"
}
```

2.6.4. Создание рабочего потока

Для создания рабочего потока используется специальный плагин для интегрированной среды разработки Eclipse, который может быть использован для построения графической модели, тестирования и развертывания процессов BPMN 2.0. Инструкции по установке Eclipse Designer приведены в документации на сайте «http://www.activiti.org/».

Рабочий поток можно создать с помощью диаграмм, добавляя необходимые элементы. Нарисованная диаграмма преобразуется в XML-файл формата .bpmn20.xml. Рабочий поток может содержать в себе форму процесса (например, форма заявки), которая содержит в себе перечень атрибутов процесса, их типов и значений. В форме процесса могут присутствовать любые из следующих типов атрибутов:

- строка;
- текст;
- целочисленный тип;
- логический тип;
- дата;
- перечисляемый тип.

Правила работы с формой процесса приведены в документации на сайте «http://www.activiti.org/», в разделе «Forms».

Примеры готовых рабочих потоков расположены в каталоге ankey/samples/workflow.

2.6.5. Настройка рабочего потока для действий с объектами Комплекса

После создания рабочего потока xml-файл рабочего потока следует переименовать с расширением .bar и поместить в каталог ankey/workflow.

В файле request.json, расположенном в каталоге ankey/conf, указаны действия, производимые с идентификатором рабочего потока для конкретной сущности.

В нижеприведенном примере для сущности «managed/usrapprole» (назначение ролей пользователю) указаны действия создания («create»), массового создания («batchCreate») с использованием рабочего потока «managerApproval».

Пример содержимого файла request.json с демонстрационными рабочими потоками:

```
{
   "entities": {
      "managed/usrapprole": [
         {
            "action": "create",
            "workflowKey": "managerApproval"
         },
         {
            "action": "batchCreate",
            "workflowKey": "managerApproval"
         }
      ],
      "managed/user": [
         {
            "action": "patch",
            "workflowKey": "seflEdit"
         }
      ],
      "resform/*": [
         {
            "action": "enable",
            "workflowKey": "autoApproval"
         }
      ]
   },
   "resources": {
      "managed/user": {
         "title": "fullName",
         "fields": [
            "mail",
            "managerFullName",
            "organizationTree"
         ]
      },
      "managed/approle": {
         "title": "appRoleName",
         "fields": [
            "appRoleName",
            "appRoleDesc"
         ]
      },
      "repo/internal/user": {
         "title": "_id",
```

```
        "fields": [
          "_id"
        ]
      }
    }
}
```

В разделе *entities* для пар (сущность, действие) задаётся процесс согласования по умолчанию (используется в случае, если нет настройки в системе).

В разделе *resource* определяются заголовок и отображаемые атрибуты объектов в пользовательском интерфейсе. Для добавления/удаления атрибута необходимо отредактировать строку «fields».

При переключении рабочего потока на другой, необходимо указать в файле название нового рабочего потока в строках «workflowKey».

Рабочие потоки, которые поставляются в составе Комплекса в каталоге ankey/workflow приведены в таблице 2.20.

Таблица 2.20 – Рабочие потоки, которые поставляются в составе Комплекса в каталоге ankey/workflow

Идентификатор	Описание
autoApproval	Рабочий поток с автоматическим согласованием заявки. Форма заявки содержит только поле «Обоснование»
autoApprovalWithForm	Демонстрационный рабочий поток с автоматическим согласованием заявки
certificateAppRoles	Рабочий поток для согласования заявок на сертификацию ролей ИС
certificateUserAppRoles	Рабочий поток для согласования заявок на сертификацию ролей пользователей
dependentEnumApproval	Аналог managerApproval. Демонстрирует возможности фильтрации значений зависимого enum поля. В состав входит дополнительное enum поле в форме заявки, зависимое от другого поля
incidentResolution	Рабочий поток для обработки инцидентов нарушения правил разграничения доступа
managerApproval	Демонстрационный рабочий поток с согласованием заявки в два этапа: 1) Согласование руководителем бенефициара. 2) Согласование участниками роли с названием «Администратор ИБ»
multiplyApproval	Демонстрационный рабочий поток реализует последовательное согласование из двух этапов: 1) Этап «Прямой руководитель или его заместитель». На данном этапе два согласующих: руководитель сотрудника и его действующий заместитель. Если заместитель отсутствует, согласующий этапа только руководитель. Если руководитель отсутствует, заявку согласовывает ANKEY.

Идентификатор	Описание
	2) Этап «Согласование администраторами». На этапе две группы согласующих: «Администратор 1» и «Администратор 2». В случае, если одна из групп отсутствует в системе, согласуют только участники существующей группы. Если отсутствуют обе группы, заявку согласовывает системный пользователь «ANKEY». Для успешного согласования заявки требуется согласование обоих этапов. Заявка будет отклонена, если её отклонят на одном из этапов
newUserManagerApproval	Аналог managerApproval. Используется для создания пользователя по заявке
parallelApproval	Демонстрационный рабочий поток реализует параллельное согласование из двух этапов: 1) Этап «1-й этап»: согласуют сотрудники роли «Role 1» или пользователь ANKEY, в случае отсутствия роли в системе. 2) Этап «2-й этап»: согласуют сотрудники роли «Role 2» или пользователь ANKEY, в случае отсутствия роли в системе. Для успешного согласования заявки требуется согласование обоих этапов. Заявка будет отклонена, если её отклонят на одном из этапов. При отклонении на любом этапе оставшийся этап, если он ещё не был обработан, будет «удален» из списка заданий заявки, ожидающих обработки
threeStageApproval	Демонстрационный рабочий поток реализует согласование заявки в три этапа: 1) Согласование руководителем бенефициара. 2) Согласование участниками роли с названием «Администратор ИБ». 3) Согласование участниками роли с названием «Ankey Admins».

2.6.5.1. Включение сертификации пользователей по заявке

Для включения сертификации пользователей по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"endpoint/user-certification/*": [
  {
    "action": "verify",
    "workflowKey": "<Название БП>"
  }
]
```

2.6.5.2. Включение сертификации ролей по заявке

Для включения сертификации ролей по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"endpoint/approle-certification/*": [
  {
    "action": "verify",
```

```
   "workflowKey": "<Название БП>"
  }
]
```

2.6.5.3. Включение создания роли по заявке

Для включения создания роли по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"endpoint/approle/with-linked-entities": [
  {
    "action": "create",
    "workflowKey": "<Название БП>"
  }
 ]
```

2.6.5.4. Включение удаления роли по заявке

Для включения удаления роли по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/approle": [
  {
    "action": "batchDelete",
    "workflowKey": "<Название БП>"
  }
 ]
```

2.6.5.5. Включение редактирования роли по заявке

Для включения редактирования роли по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"endpoint/approle/with-linked-entities": [
        {
           "action": "update",
           "workflowKey": "<Название БП>"
        }
     ]
```

2.6.5.6. Включение отзыва ролей пользователя по заявке

Для включения отзыва ролей пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"internal/usrapprole": [{
   "action": "batchDelete",
   "workflowKey": "<Название БП>"
}]
```

2.6.5.7. Включение сброса пароля пользователя по заявке

Для включения сброса пароля администратором пользователю по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/user":
[{
   "action": "resetPassword",
   "workflowKey": "<Название БП>"
}]
```

2.6.5.8. Включение активации пользователя по заявке

Для включения активации пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/user":
[{
   "action": "enable",
   "workflowKey": "<Название БП>"
}]
```

2.6.5.9. Включение удаления пользователя по заявке

Для включения удаления пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/user":
[{
   "action": "delete",
   "workflowKey": "<Название БП>"
}]
```

2.6.5.10. Включение назначения ролей по заявке

Для включения процесса назначения ролей по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/usrapprole": [
        {
```

```
        "action": "batchCreate",
        "workflowKey": "<Название БП>"
    }
]
```

В рабочем потоке процесса назначения ролей по заявке можно назначить этап согласования как на пользователя, так и на согласующую роль.

В процессе запроса пользователем необходимых ему ролей и заполнении им заявки на согласование, запускается процесс согласования назначенных ролей, с выполнением следующих действий:

1) Система высчитывает текущего согласующего или согласующую роль с перечнем согласующих пользователей, назначает на него (на согласующую роль) заявку и уведомляет согласующего (согласующих) о поступлении заявки, требующей согласования.
2) Согласующий переходит в раздел ожидающих согласования заявок и видит список всех назначенных на согласование заявок.
3) Текущий согласующий или участник согласующей роли выполняет действия по согласованию заявки.
4) Система завершает текущий этап процесса согласования и выполняет предыдущие два действия необходимое количество раз, определенное преднастроенным алгоритмом. После прохождения последнего в процессе согласования этапа, система завершает процесс согласования и дает пользователю полномочия по запрашиваемым ролям.

2.6.5.11. Включение самостоятельного редактирования атрибутов пользователем по заявке

Для включения процесса самостоятельного редактирования атрибутов пользователем по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/user": [
        {
            "action": "patch",
            "workflowKey": "<Название БП>"
        }
    ]
```

2.6.5.12. Включение активации УЗ пользователя по заявке

Для включения процесса активации УЗ пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"resform/*": [
        {
          "action": "enable",
          "workflowKey": "<Название БП>"
        }
    ]
```

2.6.5.13. Включение создания пользователя по заявке

Для включения создания пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
 "internal/user/with-linked-entities":
 [{
      "action": "create",
      "workflowKey": "newUserManagerApproval"
 }]
```

2.6.5.14. Включение редактирования пользователя по заявке

Для включения редактирования пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"internal/user/with-linked-entities":

[{
    "action": "update",
    "workflowKey": "managerApproval"
}]
```

2.6.5.15. Включение блокировки пользователя по заявке

Для включения блокировки пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/user":
[{
   "action": "disable",
   "workflowKey": "<Название БП>"
}]
```

2.6.5.16. Включение блокировки УЗ пользователя по заявке

Для включения блокировки УЗ пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
   "resform/*":

[{
   "action": "disable",
   "workflowKey": "managerApproval"
}]
```

2.6.5.17. Включение сброса пароля УЗ пользователя по заявке

Для включения процесса сброса пароля УЗ пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
  "resform/*" :

[{
   "action": "resetPassword",
   "workflowKey": <Название БП>
}]
```

2.6.5.18. Включение удаления УЗ пользователя по заявке

Для включения процесса удаления УЗ пользователя по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
  "resform/*" :

[{
   "action" : "delete",
   "workflowKey" : "managerApproval"
}]
```

2.6.5.19. Включение создания подразделения по заявке

Для включения процесса создания подразделения по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/organization":
[{
      "action": "create",
```

```
    "workflowKey": "managerApproval"
}]
```

2.6.5.20. Включение удаления подразделения по заявке

Для включения процесса удаления подразделения по заявке в соответствии с указанным бизнес-процессом (параметр «workflowKey»), необходимо добавить в секцию «entities» следующий блок:

```
"managed/organization":

[{
   "action": "batchDelete",
   "workflowKey": "managerApproval"
}]
```

2.6.5.21. Включение редактирования подразделения по заявке

Для включения процесса редактирования подразделения по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/organization":
[{
  "action": "patch",
  "workflowKey": "managerApproval"
}]
```

2.6.5.22. Включение изменения срока действия роли

Для включения процесса изменения срока действия роли по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"internal/usrapprole":

[{
        "action": "prolongate",
        "workflowKey": "managerApproval"
}]
```

2.6.5.23. Включение создания записей делегирования

Для включения процесса создания записей делегирования по заявке в соответствии с указанным бизнес-процессом (параметр *workflowKey*), необходимо добавить в секцию *entities* следующий блок:

```
"managed/substitution":
```

```
[{
    "action": "create",
    "workflowKey": "managerApproval"
}]
```

Для назначения заместителя самому себе потребуется настройка с другим action:

```
"managed/substitution":

[{
"action": "selfCreate",
"workflowKey": "managerApproval"
}]
```

2.6.5.24. Включение добавления ролей на этапе согласования заявки

Комплекс позволяет настроить бизнес-процесс на добавление ролей на этапе согласования заявки. Для этого необходимо выполнить редактирование рабочего потока «Activiti».

Возможность добавления ролей определяется переменной «allowRolesAdding», которую требуется задать на одном или нескольких этапах согласования.

Для включения добавления ролей на этапе согласования заявки необходимо выполнить следующие действия:

1) Определить этап согласования заявки.
2) В начале этапа задать переменную «allowRolesAdding» и присвоить значение «true».
3) В конце этапа задать переменную «allowRolesAdding» и присвоить значение «false».

Настройка добавления ролей может быть выполнена на разных этапах как независимых друг от друга, так и следующих один за другим. Для незавивисимых этапов настройка выполняется на каждом этапе в отдельности. Во втором случае для переменной «allowRolesAdding» следует указать значение «true» на первом этапе, а значение «false» по завершению последнего.

Пример добавления ролей на этапе согласования руководителем для бизнес-процесса «managerApproval»:

```
<userTask id="managerTask" name="Прямой руководитель">
  <extensionElements>
    <activiti:taskListener class="org.activiti.engine.impl.bpmn.listener.ScriptTaskListener"
```

```
event="create">
                <activiti:field name="language">
                  <activiti:string><![CDATA[javascript]]></activiti:string>
                </activiti:field>
                <activiti:field name="script">
                  <activiti:string><![CDATA[task.setVariable("allowRolesAdding", true);]]></activiti:string>
                </activiti:field>
              </activiti:taskListener>
              <activiti:taskListener class="org.activiti.engine.impl.bpmn.listener.ScriptTaskListener"
event="complete">
                <activiti:field name="language">
                  <activiti:string><![CDATA[javascript]]></activiti:string>
                </activiti:field>
                <activiti:field name="script">
                  <activiti:string><![CDATA[task.setVariable("allowRolesAdding", false);]]></activiti:string>
                </activiti:field>
              </activiti:taskListener>
              <activiti:formProperty id="justification" name="Обоснование" type="text"
required="true"></activiti:formProperty>
              <activiti:formProperty id="contract" name="Договор КТ" type="string" required="true"
></activiti:formProperty>
              <activiti:formProperty id="position" name="Должность" type="string"></activiti:formProperty>
              <activiti:formProperty id="type_date" name="Дата" type="date" datePattern="dd-MM-yyyy"/>
              <activiti:formProperty id="flag" name="Флаг" type="boolean"></activiti:formProperty>
              <activiti:formProperty id="location" name="Размещение" type="enum">
                <activiti:value id="lvs" name="ЛВС"></activiti:value>
                <activiti:value id="internet" name="Интернет"></activiti:value>
              </activiti:formProperty>
              <activiti:taskListener event="create" expression="${request.assignToManagerOrInternalUser(task,
execution, 'ANKEY')}"></activiti:taskListener>
            </extensionElements>
          </userTask>
```

2.6.5.25. Включение изменения дат на этапе согласования заявки

Комплекс позволяет настроить бизнес-процесс на изменение дат на этапе согласования заявки при добавлении ролей. Для этого необходимо выполнить редактирование рабочего потока «Activiti».

Возможность изменения дат определяется переменной «allowDateChange», которую требуется задать на одном или нескольких этапах согласования. При отсутствии настроек для переменной «allowDateChange» возможность изменения дат будет заблокирована.

Для включения изменения дат на этапе согласования заявки необходимо выполнить следующие действия:

1) Определить этап согласования заявки.
2) В начале этапа задать переменную «allowDateChange» и присвоить значение «true».

3) В конце этапа задать переменную «allowDateChange» и присвоить значение «false».

Настройка изменения дат может быть выполнена на разных этапах как независимых друг от друга, так и следующих один за другим. Для незавивисимых этапов настройка выполняется на каждом этапе в отдельности. Во втором случае для переменной «allowDateChange» следует указать значение «true» на первом этапе, а значение «false» по завершению последнего.

Пример включения возможности изменения дат через прямое редактирование бизнес-процесса:

```
<activiti:taskListener class="org.activiti.engine.impl.bpmn.listener.ScriptTaskListener" event="create">
  <activiti:field name="language">
    <activiti:string>
      <![CDATA[javascript]]>
    </activiti:string>
  </activiti:field>
  <activiti:field name="script">
    <activiti:string>
      <![CDATA[task.setVariable("allowDateChange", true);]]>
    </activiti:string>
  </activiti:field>
</activiti:taskListener>
```

2.6.5.26. Добавление поля «Поиск объекта» («lookup») в форму заявки

Добавление поля «Поиск объекта» («lookup») в форму заявки осуществляется в xml-файле в каталоге ankey/workflow. В соответствии с указанным бизнес-процессом необходимо добавить в xml-файл поле type=«lookup».

Первоначальная настройка рабочего потока «managerApproval» в файле managerapproval.bpmn20.xml для объекта managed/organization содержит поле «Организация» типа «Поиск объекта» и приведена на следующем примере:

```
  <startEvent id="startevent" name="Start">
      <extensionElements>
        <activiti:formProperty id="justification" name="Обоснование" type="text"
required="true"></activiti:formProperty>
        <activiti:formProperty id="contract" name="Договор КТ" type="string"
required="true"></activiti:formProperty>
        <activiti:formProperty id="position" name="Должность" type="string"
default="${position}"></activiti:formProperty>
        <activiti:formProperty id="flag" name="Флаг" type="boolean" default="false"></activiti:formProperty>
        <activiti:formProperty id="location" name="Размещение" type="enum">
          <activiti:value id="lvs" name="ЛВС"></activiti:value>
          <activiti:value id="internet" name="Интернет"></activiti:value>
        </activiti:formProperty>
```

```
        <activiti:formProperty id="type_date" name="Дата" type="date" datePattern="dd-MM-yyyy"/>
        <activiti:formProperty id="org" name="Организация" type="lookup" required="false"
object="managed/organization" lookupField="_ouid" lookupDisplayFields="name" lookupDisplayFields2="code"
queryId="get-undeleted-organizations"></activiti:formProperty>
    </extensionElements>
    </startEvent>
```

Пример добавления двух полей типа «Поиск объекта» в форму заявки рабочего потока «managerApproval» для объектов managed/organization и managed/referencebook:

```
<startEvent id="startevent" name="Start">
      <extensionElements>
        <activiti:formProperty id="justification" name="Обоснование" type="text"
required="true"></activiti:formProperty>
        <activiti:formProperty id="contract" name="Договор КТ" type="string"
required="true"></activiti:formProperty>
        <activiti:formProperty id="position" name="Должность" type="string"
default="${position}"></activiti:formProperty>
        <activiti:formProperty id="flag" name="Флаг" type="boolean" default="false"></activiti:formProperty>
        <activiti:formProperty id="location" name="Размещение" type="enum">
          <activiti:value id="lvs" name="ЛВС"></activiti:value>
          <activiti:value id="internet" name="Интернет"></activiti:value>
        </activiti:formProperty>
        <activiti:formProperty id="type_date" name="Дата" type="date" datePattern="dd-MM-yyyy"/>
        <activiti:formProperty id="org" name="Организация" type="lookup" required="false"
object="managed/organization" lookupField="_ouid" lookupDisplayFields="name" lookupDisplayFields2="code"
queryId="get-undeleted-organizations"></activiti:formProperty>
        <activiti:formProperty id="pos_in_rb" name="Должность из справочников" type="lookup"
required="false" object="managed/referencebook" lookupField="refBookCode" lookupFieldType="string"
lookupDisplayFields="refBookName" lookupDisplayFields2="refBookCode" filter="refBookType eq
'Должность'" ></activiti:formProperty>
    </extensionElements>
    </startEvent>
```

Для отображения и возможности редактирования добавленного поля «Поиск объекта» («lookup») на других этапах прохождения заявки следует внести изменения в xml-файл по аналогии с настройками для поля «Организация».

2.6.5.27. Добавление полей с дополнительными свойствами в форму заявки

Для бизнес-процесса dependentEnumApproval в рабочем потоке согласования заявки в поле enum добавляются дополнительные свойства «dependsOn» и «url», где:

- dependsOn - имя основного поля, от значения которого будет меняться состав значений зависимого enum;
- url - адрес endpoint для возврата актуальных значений зависимого enum.

Пример dependsOn:

dependsOn="mainEnumName"

Пример url:

url="endpoint/requestform/enum/dependentEnumName?process=managerApproval&value=${mainEnumNam
e}"

Пример добавления зависимых полей в форму заявки рабочего потока dependentEnumApproval:

```
<startEvent id="startevent" name="Start">
        <extensionElements>
          <activiti:formProperty id="justification" name="Обоснование" type="text" required="true"/>
          <activiti:formProperty id="contract" name="Договор КТ" type="string" required="true"/>
          <activiti:formProperty id="position" name="Должность" type="string" default="${position}"/>
          <activiti:formProperty id="flag" name="Флаг" type="boolean" default="false"/>
          <activiti:formProperty id="room" name="Помещение" type="enum">
            <activiti:value id="230" name="230"/>
            <activiti:value id="237" name="237"/>
          </activiti:formProperty>
          <activiti:formProperty id="location" name="Размещение" type="enum" dependsOn="room"
url="endpoint/requestform/enum/location?process=dependentEnumApproval&amp;amp;value=${room}">
            <activiti:value id="one" name="1"/>
            <activiti:value id="two" name="2"/>
            <activiti:value id="three" name="3"/>
          </activiti:formProperty>
          <activiti:formProperty id="type_date" name="Дата" type="date" datePattern="dd-MM-yyyy"/>
          <activiti:formProperty id="org" name="Организация" type="lookup" required="false"
                        object="managed/organization" resultsDisplayAmount="3" lookupField="_ouid"
                        lookupDisplayFields="name" lookupDisplayFields2="code"
                        queryId="get-undeleted-organizations"/>
        </extensionElements>
      </startEvent>
```

Для настройки значений динамического enum:

1. Внутри бизнес-процесса для зависимого enum указываются все возможные значения (без учёта зависимостей от других полей).
2. В конфигурации requestform.enum.json указываются различные комбинации значений основных и зависимых enum.

Пример конфигурации requestform.enum.json:

```
{
  "mainEnumName": {
    "dependentEnumName": {
      "mainEnumValue#1": {"dependent": "зависимое", "value", "значение"},
      "mainEnumValue#2": {"yet": "еще", "another": "одно", "dependent": "зависимое", "value", "значение"}
    }
  }
}
```

Особенности:

- если в основном enum не совершен никакой выбор, то зависимый enum будет заблокирован;
- при некорректно заполненной конфигурации (пустой config или указаны несуществующие поля или значения), зависимый enum будет пуст;
- при выборе значения в основном enum последующий выбор в зависимом enum будет уже с учетом фильтрации;
- при выборе значения в зависимом enum и последующем выборе в основном enum, значение зависимого enum сбросится.

2.6.6. Удаление Бизнес Пакета (.bar)

Для полного и корректного удаления Бизнес Пакета (БП), следует выполнить следующие шаги:

- Шаг 1. Проверка связей БП с ИС;
- Шаг 2. Удаление БП на файловом уровне;
- Шаг 3. Удаление БП через REST API.

2.6.6.1. Проверка связей БП с информационными системами

Удалить БП можно лишь при полном отсутствии связей БП с ИС.

В противном случае при отправке REST запроса будет выброшена ошибка о неудачной попытке удаления. Ошибка содержит в себе информацию о количестве ИС, которые на момент удаления БП имеют с ним связь.

Для успешного удаления БП, данные ИС необходимо предварительно отредактировать через интерфейс IDM, очистив ссылку на БП. После этого можно приступать к удалению (см. следующие шаги).

2.6.6.2. Удаление БП на файловом уровне

Удалить БП на файловом уровне можно как на активном, так и на не запущенном программном комплексе Ankey IDM.

Для этого надо:

1) Перейти в директорию workflow: ankey/workflow.
2) Удалить необходимый БП (.bar файл).

2.6.6.3. Удаление БП через REST API

Удалить БП через REST API можно исключительно на уже запущенном программном комплексе Ankey IDM.

Для этого необходимо выполнить следующие действия:

1) Отправить REST запрос на получение списка идентификаторов всех Process Definition:

   GET: http://<hostname>:<port>/ankey/workflow/processdefinition?_queryId=query-all-ids

2) Выбрать подходящий идентификатор Process Denition (атрибут «_id»).
3) Удалить необходимый Process Denition:

   DELETE: http://<hostname>:<port>/ankey/workflow/processdefinition/<id>

Примечания

1) Каждый Process Definition имеет ссылку на «deploymentId». Это идентификатор развернутого БП.
2) Один БП может содержать в себе сразу несколько Process Definition.
3) Удаляя любой из Process Definition из пакета, уничтожается весь пакет целиком, в том числе все связанные с ним процессы Process Definition.

2.6.7. Настройка вложений в заявку

Настройка вложений в заявку выполняется в файле conf/document.json. Пример конфигурационного файла document.json:

```
{
 "ttl": 1800,
 "maxFileSize": 10485760,
 "mimeTypes": [
  "application/zip",
  "application/x-zip-compressed",
  "application/pdf",
  "image/jpeg",
  "image/tiff"
 ],
 "fileTypes": [
  "zip",
  "pdf",
  "jpeg",
  "jpg",
  "tiff"
 ]
}
```

Конфигурационный файл содержит следующие параметры:

- в параметре «ttl» указывается время жизни загруженного документа до его использования в секундах;
- в параметре «maxFileSize» указывается максимальный размер файла вложения в байтах;
- в параметре «mimeTypes» указывается список поддерживаемых MIME-типов;
- в параметре «fileTypes» указываются типы поддерживаемых файлов.

На каждый MIME-тип необходимо указывать свой тип файла, иначе произойдет ошибка «PolicyValidationFail».

2.6.8. Настройка отзыва заявки заявителем

Для корректной работы отзыва заявки в рабочем потоке потребуется добавить новый поток (sequenceFlow) от промежуточного шлюза «gateway» до завершающего шлюза «gateway» и дать название данному потоку, например, «отменено».

Пример графической настройки демонстрационного рабочего потока «managerApproval» приведен на рисунке 2.1.

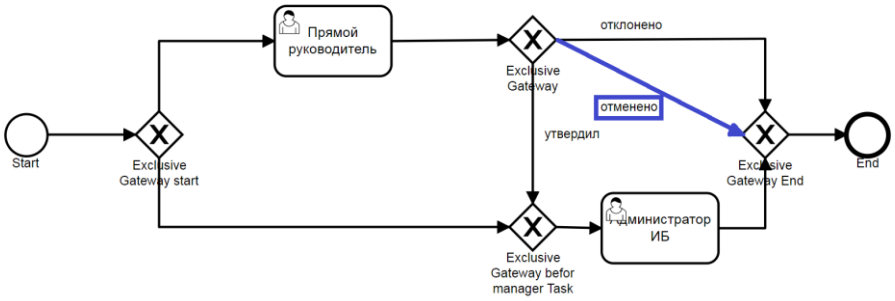

Рисунок 2.1 – Пример графической настройки демонстрационного рабочего потока «managerApproval»

Дополнительно необходимо задать условие срабатывания данного потока, которое зависит от переменной «ankey_outcome», и должно срабатывать если переменная принимает значение «cancelled». Пример настройки условия срабатывания для демонстрационного рабочего потока «managerApproval»:

```
<sequenceFlow id="sf_cancel_before_eg_end" name="отменено" sourceRef="eg_after_manager"
targetRef="eg_end">
   <conditionExpression xsi:type="tFormalExpression"><![CDATA[${ankey_outcome ==
'cancelled'}]]></conditionExpression>
</sequenceFlow>
```

2.6.9. Настройка отображения фотографии в деталях заявки

Для отображения фотографии пользователя в деталях заявки необходимо в каталоге ankey/conf в файле request.json добавить в блок «resources» поле «photoPreview».

Пример содержимого файла request.json с настройкой для отображения фотографии пользователя:

```
"resources": {
        "managed/user": {
                "title": "fullName",
                "fields": ["photoPreview"]
                ...
                }
        ...
        }
```

2.7. Настройка дополнительных атрибутов

Комплекс предоставляет возможность администратору добавлять новые атрибуты для любого управляемого объекта.

Для добавления нового атрибута администратору следует выполнить следующие действия:

1) Настроить дополнительные атрибуты в конфигурационном файле extend.json. Подробнее приведено в пункте 2.7.1.
2) Создать атрибуты в таблицы БД с помощью скриптов Liquidbase. Подробнее приведено в пункте 2.7.2.
3) Настроить локализацию дополнительных полей. Подробнее приведено в пункте 2.7.3.

2.7.1. Настройка дополнительного атрибута в конфигурационном файле extend.json

Для настройки дополнительных атрибутов администратору необходимо отредактировать конфигурационный файл extend.json в каталоге ankey/conf/.

При редактировании следует добавить атрибут в область управляемого объекта и заполнить параметры «repo», «policy», «managed»:

- в параметре «repo» указываются настройки хранения в БД;
- в параметре «policy» указываются политики валидации для данного атрибута;
- в параметре «managed» указывается параметры для отображения в интерфейсе.

Пример структуры файла extend.json:

```
{
   "managed/user": {
      "customAttr": {
         "repo": {},
         "policy": {},
         "managed": {}
      }
   }
}
```

, где:

- «managed/user» – управляемый объект «пользователь»;
- «customAttr» – название добавляемого атрибута.

Настройки параметра «repo» приведены в таблице 2.21.

Таблица 2.21 – Настройки параметра «repo»

Настройки параметра	**Описание**	**Пример**

column	Имя колонки в БД, которое будет создано с помощью скрипта Liquibase	"repo": {
		"column": "customattr",
		"type": "STRING",
		"init": {"file":
		"db/postgresql/liquibase/custom.xml",
		"type": "liquibase/xml"
		}

type	Тип пользовательского атрибута	
init	Указывается какой скрипт Liquibase будет использоваться для создания колонки в БД и состоит из параметров: – file – путь, где расположен скрипт Liquibase; – type – тип файла скрипта Liquibase	

Настройки параметра «policy» в таблице 2.22.

Таблица 2.22 – Настройки параметра «policy»

Настройки параметра	Описание	Пример

schema	Определение атрибута в JSON-SCHEMA	"policy ": {
		"schema": {
		"type": "string",
		"example": "My custom attr!",
		"required": false,
		"policies": [{
		"policyId": "maximumLength",
		"params": {
		"numChars": 64`
		}
		}]
		}

type	Тип атрибута	
example	Описание атрибута	
required	Признак обязательности поля (возможные значения true, false)	
policies	Список политик валидации дополнительного атрибута, где каждый элемент состоит из параметров: – policyId – имя политики; – params – параметры политики	

Настройки параметра «managed» в таблице 2.23.

Таблица 2.23 – Настройки параметра «managed»

Настройки параметра	Описание	Пример
required	Признак обязательности поля (возможные значения true, false). В интерфейсе отображается как символ «*», если указано значение true	"managed": { "required": false, "fieldType": "textField", "display": ["create", "update", "view", "selfEdit", "viewUserCard", "viewRequestCard", "viewMyProfile"]}

fieldType	Тип отображаемого поля, по умолчанию имеет значение textField. Возможные значения для типа поля «fieldType»:
	– textField;
		"managed/user": {
		"searchable_bijsr": {
		"managed": {
		"fieldType": "textField",
		"display": [

Настройки параметра	Описание	Пример

	– textArea;
	– date;
	– timestamp;
	– flag (поле типа boolean);
	– number;
	– lookup;
	– passwordField;
	– tree (поле типа иерархическая структура, например «организация»);
	– group;
	– status
		"view",
		"viewMyProfile",
		"create",
		"update"
		"searchable": true
		},
		"policy":
		],
		{ "schema":
		{ "example": "",
		"required": false,
		"type": "string" }
		},
		"repo":{
		"column": "searchable_bijsr",
		"type": "STRING",
		"init":{ "file": "conf/liquibase.json", "type": "liquibase/json"
		}
		}
		}

display	Массив с указанием, в каких формах отображать атрибут. Возможные значения:
	1) create: форма создания и детали заявки на создание роли для управляемого объекта «managed/approle», форма создания пользователя для управляемого объекта «managed/user», форма создания подразделения для управляемого объекта «managed/organization».
	2) update – форма редактирования и детали заявки на редактирование.
	3) view – форма просмотра.
	4) selfEdit – возможность самостоятельного редактирования поля пользователем в карточке пользователя.
	5) viewUserCard – карточка пользователя, отображаемая в меню «Пользователи».
	6) viewRequestCard – карточка пользователя, отображаемая в форме
		"managed/user": {
		"personnelNumber": { "managed": {
		"fieldType": "textField",
		"display":"viewModal"],
		"searchable": true,
		"search":["informationSystemUser"]
		}
		}
		}}

Настройки параметра	Описание	Пример
	заявки. 7) viewMyProfile – карточка пользователя, отображаемая в меню «Мой профиль». 8) viewSubordinates – карточка просмотра прямых подчиненных. 9) viewModal – карточка деталей пользователя, отображаемая при поиске пользователя. Отображается в свернутом виде. 10) viewRequestUserDetail – карточка деталей пользователя «Параметры пользователя», отображаемая в деталях заявки при создании пользователя по заявке. 11) certification_update – форма редактирования роли, отображаемая, когда сертификатор из заявки на сертификацию роли вызывает редактирование роли. 12) audit – карточка истории изменений пользователя, отображаемая в меню «Пользователи»	
searchable	Используется дополнительно для работы поиска. Для поддержки полнотекстового поиска указывается значение searchable: true. Свойство searchable указывается для всех типов полей, кроме passwordField. Если свойство не указано, то по умолчанию принимает значение false. После добавления, изменения, удаления свойства searchable из файла extend.json требуется запустить реиндексацию (тип задания reindextask)	

Настройки параметра	Описание	Пример
	для применения изменений	
search	Используется для настройки поиска. Указывается значение, которое определяет, где настраивается поиск. Значение «informationSystemUser» - вкладка «Пользователи» в деталях ИС, меню «Инф. системы»	

placeholder	Используется для отображения подсказки в поле. Указывается ключ локализации атрибута, для которого будет отображаться подсказка. Доступно для полей типа:
	– textField;
	– number;
	– lookup. Необходимо указать соответствующий перевод для текста подсказки в файлах локализации
		"managed": {
		"placeholder":
		"templates.managed.form.organization.lookup.placeholder.orgKurator"
		}

Список и описание политик валидации дополнительных атрибутов, настраиваемые в параметре «policy», приведены в подпункте 2.7.1.1.

Поддерживаемые типы атрибутов приведены в подпункте 2.7.1.2.

При отображении в форме заявки через поле viewRequestCard, требуется дополнительно прописать данное поле в конфигурационные файлы extrend.json и request.json.

Пример файла extrend.json:

```
"isName": {
  "managed": {
    "required": false,
    "display": [
       "view",
      "viewRequestCard"
    ],
    "fieldType": "textField"
  },
```

Пример файла request.json:

```
"resources" : {
   "managed/user" : {
    "title" : "fullName",
     "fields" : [
        "mail",
        "isName"
     ]
 },
```

Примечания

1) При настройки параметра «viewRequestCard» в «managed» →«display» требуется дополнительно прописать поля в request.json: «resources»→ «managed/user».
2) Для расширения видимости существующих атрибутов Комплекса достаточно прописать видимость атрибута только в «managed» →«display» (исключая «repo» и «policy»).

2.7.1.1. Политика валидации дополнительного атрибута

При настройке дополнительного атрибута в блоке <*объект*> → <*атрибут*> → policy → policies можно задать необходимые политики валидации.

Можно использовать как поставляемые с Комплексом политики, так и политики собственной разработки.

Пример настройки ограничения в 64 символа для дополнительного атрибута extendAttribute:

```
{
  "managed/user": {
    "extendAttribute": {
      "managed": {
        "fieldType": "textField",
        "display": [
          "view",
          "viewMyProfile",
          "create",
          "update"
        ]
      },
      "policy": {
        "schema": {
          "example": "extend value",
          "required": false,
          "type": "string"
        },
        "policies": [{
          "policyId": "maximumLength",
          "params": {
```

```
                "numChars": 64
              }
            }]

        },
        "repo": {
          "column": "searchable_bijsr",
          "type": "STRING",
          "init": {
            "file": "conf/liquibase.json",
            "type": "liquibase/json"
          }
        }
      }
    }
}
```

Для настройки политики нужно указать её идентификатор в атрибуте **policyId** и при необходимости задать ей конфигурацию в атрибуте **params**.

В примере выше на атрибут extendAttribute объекта managed/user настроена политика maximumLength, со значением 64 у параметра numChars.

Все политики, кроме required, разрешают отсутствие значения у проверяемого атрибута (null).

Список и описание политик валидации дополнительных атрибутов приведен в таблице 2.24.

Таблица 2.24 – Список и описание политик валидации дополнительных атрибутов

Имя политики	Описание	Параметры конфигурации
required	Проверка, что значение не null и не пустая строка после удаления всех начальных и конечных пробелов	–
unique	Проверка, что нет другого объекта с таким же значением валидируемого атрибута. В случае, если значение строкового типа, проверка осуществляется без учёта регистра	–

cannotContainCharacters	Проверка, что в значении атрибута нет запрещенных символов	forbiddenChars - массив запрещенных строк. Пример:
		
		"params": {
		"forbiddenChars": [
		"/"
		]
		}

Имя политики	Описание	Параметры конфигурации

maximumLength	Проверка максимальной длины	numChars - максимальная длина атрибута. Пример:
		
		"params": {
		"numChars": 255
		}

regexpMatches	Проверка соответствия значения регулярному выражению	1) regexp - регулярное выражение.
		2) flags - флаг (необязательный), допустимые значения i - регистровая независимость и m - многострочный режим. Пример:
		
		"params": {
		"regexp": "^([^\\s\\@])+\\@([^\\s\\@])+$"
		}

enum	Проверка, что значение среди множества допустимых	values — массив разрешённых значений. Пример:
		
		"params": {
		"values": [
		"active",
		"disabled"
		]
		}

resourceExists	Проверка существования ресурса с заданными числовым идентификатором (ouid)	resource - имя ресурса для проверки (обязательный параметр). Пример:
		
		"params": {
		"resource": "managed/approle"
		}

isInteger	Проверка целочисленности значения	–

2.7.1.2. Поддерживаемые типы пользовательских атрибутов

Тип пользовательского атрибута указывается в настройках параметра «repo» конфигурационного файла extend.json.

Для добавления доступны поля следующих типов:

- String;
- Integer;
- Object;
- Boolean.

Поддерживаемые типы пользовательских атрибутов представлены в таблице 2.25.

Таблица 2.25 – Поддерживаемые типы пользовательских атрибутов

Тип поля управляемого объекта (managed/fieldType)	**Тип поля (policy/schema/type)**	**Тип в БД (repo/type)**	**Описание**
textField	string	STRING	Строковое поле
textArea	string	STRING	Многострочное поле
flag	boolean	BOOLEAN	Boolean поле
number	integer	INTEGER	Числовое поле, поддерживаются значения от -2147483648 до 2147483647
passwordField	string	STRING	Поле для хранения пароля
tree	integer	BIGINT	Иерархия
date	string	DATETIME	Поле для хранения даты
enum	string	STRING	Перечисляемый тип
lookup	string, integer	STRING, BIGINT	Поле с возможностью поиска объекта

2.7.1.2.1 Настройка дополнительного поля типа «Строка»

Пример добавления поля «customAttr» типа string в карточку пользователя. Для добавления нового поля «customAttr» типа string в карточку пользователя администратору следует выполнить следующие действия:

1) Изменить конфигурационный файл extend.json в каталоге ankey/conf/, добавив в параметры «repo», «policy», «managed» следующие строки:

```
{
   "managed/user": {
   "customAttr" : {
     "repo": {
       "column": "customattr",
       "type": "STRING",
       "init" : {
         "file" : "db/postgresql/liquibase/custom.xml",
         "type" : "liquibase/xml"
       }
   },
     "policy" : {
       "schema" : {
         "type" : "string",
         "example" : "My super custom attr!",
```

```
        "required" : true
      },
      "policies": [
       {
        "policyId": "required"
       },
       {
        "policyId": "maximumLength",
        "params": {
         "numChars": 512
        }
       }
      ]
     },
     "managed" : {
      "required" : true,
      "fieldType" : "textField",
      "display" : ["create","update","view","selfEdit"]
     }
    }
   }
}
```

2) В каталоге ankey/db/postgresql/liquibase создать скрипт custom.xml и добавить в него следующие строки, оставив стандартный блок со схемами:

```
<?xml version="1.0" encoding="UTF-8"?>

<databaseChangeLog
      xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.1.xsd">
   <changeSet id="100" author="gis">

      <addColumn tableName="usr" schemaName="ankey">
         <column name="customattr" type="character varying(128)"/>
      </addColumn>
   </changeSet>
</databaseChangeLog>
```

3) Создать бандл локализации. В файл русской локализации translation_ru.properties добавить следующие строчки:

```
templates.managed.form.user.category=\\u041d\\u043e\\u0432\\u044b\\u0439\\u0020\\u043
0\\u0442\\u0440\\u0438\\u0431\\u0443\\u0442

templates.managed.card.user.category=\\u041d\\u043e\\u0432\\u044b\\u0439\\u0020\\u043
0\\u0442\\u0440\\u0438\\u0431\\u0443\\u0442
```

В файл английской локализации translation_en.properties добавить следующие строчки:

```
templates.managed.form.user.category=customAttr
templates.managed.card.user.category=customAttr
```

2.7.1.2.2 Настройка дополнительного поля «Перечисляемый тип»

Примеры конфигурационного файла extend.json для заведения полей с типом enum:

1) Пример дополнительного поля для объекта «Пользователь»:

```
{
   "managed/user": {
      "address": {
         "repo": {
         "column": "address",
         "type": "STRING",
            "init": {
            "file": "liquibase/custom_usr.xml",
            "type": "liquibase/xml"
        }
      },
         "policy": {
            "schema": {
            "type": "string",
              "example": "ул. Васи Алексеева д.3"
            },
       "policies": [{
         "policyId": "maximumLength",
         "params": {
            "numChars": 64
         }
       }]
    },
    "managed": {
      "required": false,
      "fieldType": "enum",
      "values": [
         "ул. Кронштадтская, д.10, литера А",
         "ул. Намёткина, д. 10А"
      ],
      "display": [
         "create",
         "update",
         "view",
         "viewUserCard"
           ]
        }
      }
   }
}
```

2) Пример расширения значений существующего поля типа enum.

```
{
  "managed/resaccount": {
    "beneficiarObject": {
      "managed": {
        "values": [
          "managed/approleresaccount"
        ]
      }
    }
  }
}
```

В файл translation_ru.properties для локализации нового значения требуется добавить перевод, например, «Привилегии роли»:

templates.managed/approleresaccount.title=\u041f\u0440\u0438\u0432\u0438\u043b\u0435\u0433\u0438\u0438\u0020\u0440\u043e\u043b\u0438

3) Пример расширения значений дополнительных полей типа enum для дальнейшей настройки фильтрации:

```
"managed/approle": {
    "landscape": {
        "managed": {
            "fieldType": "enum",
            "values": [
                "Продуктив",
                "Разработка",
                "Тестирование"
          ],
            "display": ["create",
                "update",
                "view"
            ],
          "search": [
          ],
            "searchable": false
        },
        "policy": {
            "schema": {
            "example": "",
            "required": false,
            "type": "string"
          }
      },
        "repo": {
            "column": "landscape",
            "type": "STRING",
            "init": {
              "file": "conf/liquibase.json",
              "type": "liquibase/json"
```

```
            }
          }
        },
        "risk": {
          "managed": {
             "fieldType": "enum",
             "values": [
                 "Высокий",
                 "Средний",
                 "Низкий"
              ],
               "display": ["create",
                 "update",
                 "view"
             ],
               "search": [
             ],
               "searchable": false
             },
             "policy": {
               "schema": {
                   "example": "",
                  "required": false,
                 "type": "string"
               }
            },
            "repo": {
              "column": "risk",
              "type": "STRING",
             "init": {
                   "file": "conf/liquibase.json",
                   "type": "liquibase/json"
               }
             }
          }
        }
```


Дополнительные поля, по которым будет производиться фильтрация необходимо добавить в конфигурационный файл ankey/conf/filter.json:


```
{
   "landscape": {"defaultValue": "Продуктив"
   },
   "risk": {
   }
}
```


Параметр «defaultValue» может быть указан, если необходимо установить для фильтра значение по умолчанию.

Пример настройки фильтрации для ролей и для ИС:

1) Настроить Extend-поле с типом enum и указать, что поле является поисковым: «searchable»:true

Пример конфигурационного файла extend.json для ИС

```
{
"managed/is": {
"isfield": {
 "managed": {
   "searchable": true,
   "required": false,
    "display": [
    "create",
    "update",
    "view",
    "selfEdit",
    "viewMyProfile"
 ],
   "fieldType": "enum",
   "values": [
   "ул. Кронштадтская, д.10, литера А",
   "ул. Намёткина, д. 11"
  ]
     },
"policy": {
 "schema": {
 "type": "string"
    }
 },
"repo": {
  "column": "isfield",
  "type": "STRING"
       }
     }
    }
   }
```

2) Указать правила фильтрации для ИС или ролей в filter. При этом следует учесть следующее:
   - (обязательно) при настройке фильтра требуется указать имя объекта фильтрации. Для ролей «approle», для ИС: «is»;
   - (опционально) в фильтре можно указать значение фильтрации по умолчанию.

Пример filter для ИС:

```
{
 "is": {
   "isfield": {}
 }
}
```

Пример filter для ИС с использованием фильтра по умолчанию:

```
{
 "is": {
  "isfield": {
   "defaultValue": "ул. Намёткина, д. 11"
  }
 }
}
```

Пример filter для ИС и ролей:

```
{
 "approle": {
  "rolefield": {}
 },
 "is": {
  "isfield": {}
 }
}
```

2.7.1.2.3 Настройка дополнительного поля «Поиск объекта»

Пример создания поля типа «Поиск объекта» («lookup») в карточке «Подразделения».

Для добавления нового поля «org_kurator» типа «lookup» в карточку «Подразделения» администратору следует выполнить следующие шаги:

Шаг 1. Изменить конфигурационный файл extend.json в каталоге ankey/conf/, добавив в параметры «repo», «policy», «managed» следующие строки:

```
{
  "managed/organization": {
    "orgKurator": {
      "managed": {
        "fieldType": "lookup",
        "display": [
          "create",
          "update",
          "view"
        ],
        "displayField": "orgKuratorName",
        "object": "managed/user",
        "searchStartLength": 3,
        "resultsDisplayAmount": 5,
        "lookupDisplayFields": [
          "userName"
        ],
        "lookupDisplayFields2": [
          "personnelNumber"
        ],
```

```
            "lookupField": "_ouid",
            "queryId": null,
            "filter": "position eq 'Начальник'"
         },
         "policy": {
            "schema": {
               "example": "MANAGER",
               "required": false,
               "type": "integer"
            }
         },
         "repo": {
            "column": "org_kurator",
            "init": {
               "file": "liquibase/custom_organization.xml",
               "type": "liquibase/xml"
            },
            "type": "BIGINT"
         }
      }
   }
}
```

При редактировании конфигурационного файла следует учесть следующие моменты:

1) Тип атрибута type должен обязательно совпадать с типом атрибута «lookupField». В данном примере lookupField=ouid, ouid имеет тип bigint, поэтому "type": "BIGINT".
2) Фильтр должен быть настроен так, чтобы для каждого lookupField он был уникальным. Например, имеется справочник referencebook для разных ресурсов (resource_id=1, resource_id=2) с одинаковыми значениями refBookType="Сотрудник". По фильтру "filter": "refBookType eq 'Сотрудник' and resource_id eq 1" возвращается одно значение для lookupField - из первого ресурса и это будет правильной настройкой. Настройка "filter": "refBookType eq 'Сотрудник'" не будет являться корректной, так как вернет сотрудников из разных ресурсов.
3) Поля типа «Поиск объекта» («lookup») не должны возвращать удаленные объекты. Для этого в качестве объекта атрибута («object») предпочтительнее использовать значение «object»: «enpoint/lookup/{name}» (Например, "endpoint/lookup/approle"). В

случае указания объекта атрибута («object») в виде «managed/{name}» следует пользоваться параметром «filter». Для запроса пользователей используется настройка "filter": "!(/accountStatus eq 'deleted') ", для остальных сущностей используется "filter": "!(/status eq 'deleted') ".

4) В качестве объекта атрибута («object») для корректного отображения пользователям с ролями «Оператор», «Администратор ролей ИС» необходимо указывать следующие значения:
   – "object": "endpoint/lookup/user" – для поиска по доступным для просмотра пользователям;
   – "object": "endpoint/lookup/organization" – для поиска по доступным для просмотра подразделениям;
   – "object": "endpoint/lookup/approle" – для поиска по доступным для просмотра ролям.
5) Следует указать уникальные между собой имя extend-поля и его «отображаемое» имя (displayName). В вышеупомянутом примере эти имена, как раз, отличаются: «orgKurator» и «orgKuratorName». Несоблюдение этого требования может повлечь за собой поломку отображения заявок на редактирование данного extend-поля.

Шаг 2. Добавить колонку в БД в которой будет храниться ouid выбранного объекта в lookup поле. Для этого в каталоге ankey/db/postgresql/liquibase создать скрипт custom_organization.xml и добавить в него следующие строки, оставив стандартный блок со схемами:

```
<?xml version="1.0" encoding="UTF-8"?>
<databaseChangeLog
      xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.1.xsd">

   <changeSet id="100" author="gis">
      <addColumn tableName="organization" schemaName="ankey">
         <column name="org_kurator" type="bigint"/>
      </addColumn>
   </changeSet>
</databaseChangeLog>
```

Шаг 3. Создать объект «handler» для заполнения виртуального поля. Поле «orgKuratorName», которое указанно в «displayField» не существует и необходимо создать виртуальное поле для объекта. Пример объекта «handler» на основе идентификатора из поля «ogrKurator», который предзаполняет поле «orgKuratorName» для корректного отображения в интерфейсе Комплекса:

```
package com.gis.idm.integration.handlers;
import com.gis.idm.api.config.ConfigurationHelper;
import com.gis.idm.api.managed.HandlerResult;
import com.gis.idm.api.managed.ManagedObjectHandler;
import com.gis.idm.api.model.Organization;
import com.gis.idm.api.service.data.UserService;
import org.apache.felix.scr.annotations.*;
import org.forgerock.json.JsonValue;
import org.forgerock.json.resource.Request;
import org.forgerock.json.resource.ResourceException;
import org.forgerock.openidm.util.ContextUtil;
import org.forgerock.services.context.Context;
import org.forgerock.util.promise.Promise;

import java.util.Map;

import static org.osgi.framework.Constants.SERVICE_DESCRIPTION;

/**
 * Handler для показа lookup UDF полей
 * Заполнение виртуальных полей, для отображения в UI
 */
@Service(ManagedObjectHandler.class)
@Component(
        immediate = true,
        name = OrganizationHandler.PID
)
@Properties({
        @Property(name = SERVICE_DESCRIPTION, value = "GiS.IDM :: OrganizationIntegrationHandler
Service"),
        @Property(name = ManagedObjectHandler.PROPERTY_RESOURCE, value = Organization.MANAGED),
        @Property(name = ManagedObjectHandler.PROPERTY_ORDER, value =
ManagedObjectHandler.DEFAULT_USER_LEVEL)
})
public class OrganizationHandler implements ManagedObjectHandler {
    static final String PID = ConfigurationHelper.DEFAULT_SERVICE_RDN_PREFIX +
"OrganizationIntegrationHandler";
    private static final String SUB_ORG_ID = "orgKurator";
    private static final String VIRTUAL_ORG_KURATOR = "orgKuratorName";

    @Reference
    private UserService userService;

    /**
     * При получении объекта из БД заполняем необходимые виртуальные поля
     *
     * @param context контекст сервсиа
```

```
    * @param request объект Request
    * @param object  полученный объект, в данном случае организация
    * @param args
    * @return
    */
   @Override
   public Promise<HandlerResult, ResourceException> onRetrieve(Context context, Request request, JsonValue
object, Map<String, Object> args) {
      return addSubOrgName(context, object).then(ignore -> HandlerResult.OK);
   }

   private Promise<JsonValue, ResourceException> addSubOrgName(Context context, JsonValue object) {
      return executeIf(ContextUtil.isExternal(context), () -> {
         Long orgKuratorOuid = object.get(SUB_ORG_ID).asLong();
         return executeIf(orgKuratorOuid != null, () -> userService.findByOuid(context, orgKuratorOuid)
               .then(user -> user != null
                     ? object.put(VIRTUAL_ORG_KURATOR, user.getUserName())
                     : object));
      });
   }
}
```

Необходимо проверять является ли контекст внешним, так как данные виртуальные поля используются только для отображения. В противном случаем данные поля будут заполняться при всех запросах, что влечет повышение нагрузки на систему.

Пример настройки поля «targetInfoSystem» типа «Поиск объекта» («lookup») в шаблоне периодического задания «approlecertificationtask».

```
 "org.forgerock.openidm.approlecertificationtask": {
   "id": "http://json-schema.org/draft-04/schema#",
   "$schema": "http://json-schema.org/draft-04/schema#",
   "description": "AppRole Certification",
   "type": "object",
   "properties": {
    "targetInfoSystem": {
      "type": "string",
      "example": "Сетевое рабочее место",
      "description": "Наименование ИС, роли которой подлежат сертификации",
      "fieldType": "lookup",
      "displayField": "name",
      "object": "endpoint/lookup/is",
      "managedObject": "managed/is",
      "searchStartLength": 3,
      "resultsDisplayAmount": 5,
      "placeholder": "templates.managed.form.approle.lookup.placeholder.targetInfoSystem",
      "lookupDisplayFields": [
        "name"
      ],
      "lookupDisplayFields2": [
        "description"
      ],
```

```
      "lookupField": "name",
      "queryId": null,
      "filter": "true"
    }
  },
  "required": [
    "targetInfoSystem"
  ],
  "additionalProperties": false
}
```

2.7.1.2.4 Настройка дополнительных атрибутов для автозаполнения

Автозаполнение можно использовать при создании и редактировании роли.

Для обеспечения возможности настройки дополнительных атрибутов для автозаполнения файл настройки расширенной конфигурации extend config для appRoleName и appRoleDesc в managed/approle дополнен полем autocomplete:{}, содержащим:

– defaultPattern – обязательное поле, шаблон построения значений для автозаполнения, а если не получится построить ничего подходящего, то используется defaultPattern;

– sourceFields – обязательное поле, содержащее значения которые могут использоваться в автозаполнении (можно использовать все поля доступные в ui.managed.config, включая extend поля);

– mappings – обязательное поле, маппинги для полей, используемые в том случае, когда значение sourceFields отсутствует или выбрано больше одного значения;

– organization – зависит от значений sourceFields, которых может быть несколько и необходимо заполнить маппинг для каждого из указанных в sourceFields полей:

- multiValue – если выбрано несколько организаций, то в pattern подставляется значение multiValue;
- emptyValue – если организации отсутствуют, то в pattern подставляется значение emptyValue;

– patterns – необязательное поле, которое заполняется в том случае, когда в зависимости от resType требуются разные паттерны:

– pattern – шаблон для ресурсов с resType = XML File;

– resTypeName – если заданы patterns, то при создании роли система предложит заполнить сначала УЗР, а затем уже остальные параметры.

Для автозаполнения можно использовать все поля вкладок «Параметры», «Подразделения», «Учетные записи ресурсов». Названия полей перечислены в ui.managed config для объекта approle. Также есть возможность обращаться к атрибутам объектов выбранных в lookup полях через точку. Например:

В объект appRole добавлено extend lookup поле, настроенное на объект «referencebook» с названием landscape.

Есть справочник с значениями:

refBookCode= «DEV», refBookName = «зона разработки»

Для того чтобы обратиться к атрибутам, необходимо настроить паттерн следующим образом:

– для имени роли:

```
pattern: "
{landscape.refBookCode}
"
```

– для описания роли pattern:

```
"landscape.refBookName"
```

Если атрибут отсутствует будет проставляться null.

Если autocomplete задан хотя бы для одного из полей appRoleName или appRoleDesc, то автозаполнение включается и поведение при создании роли изменяется:

1) На вкладке «Параметры» появится информационное сообщение «Включена настройка автозаполнения полей».
2) Над полями появится кнопка, при нажатии на которую в поле подставится значение, полученное с помощью шаблона автозаполнения.
3) Если, при необходимости использовать разный шаблон в зависимости от выбранного ресурса, заполнен блок patterns, то при открытии модального окна сразу будет осуществлен переход на вкладку

«Учетные записи ресурса» и появится информационное сообщение «Шаблон автозаполнения будет задан в зависимости от выбранного ресурса»:

– паттерн автозаполнения будет выбран в зависимости от ПЕРВОГО выбранного ресурса;
– если ресурс не будет выбран, то будет использован defaultPattern – это поле является обязательным к заполнению и не может быть пустым.

```
{
  "managed/approle": {
    "appRoleName": {
      "autoComplete": {
        "defaultPattern": "default_{organization}",
        "patterns": [
          {
            "pattern": "xml_{organization}",
            "resTypeName": "XML File"
          }
        ],
        "mappings": {
          "organization": {
            "multiValue": "MULTIPLEORG",
            "emptyValue": "Подразделение"
          }
        },
        "sourceFields": [
          "organization"
        ]
      }
    },
    "appRoleDesc": {
      "autoComplete": {
        "defaultPattern": "Роль_{is-id}",
        "patterns": [],
        "mappings": {
          "is-id": {
            "multiValue": null,
            "emptyValue": "NoIS"
          }
        },
        "sourceFields": [
          "is-id"
        ]
      }
    }
  }
}
```

2.7.1.2.5 Настройка отображения изменений пользователя в деталях

массовых операций

Для отображения изменений у пользователя в деталях массовых операций используется схема viewBulkOperation, имеющая следующий набор полей:

```
"viewBulkOperation": [
        "userName",
        "accountStatus",
        "enableDate",
        "disableDate",
        "lastName",
        "firstName",
        "middleName",
        "mail",
        "phone",
        "organization_id",
        "position",
        "personnelNumber",
        "usrOrgTabNum",
        "managerObjectId",
        "locked",
        "avatarId"
      ]
```

При добавлении поля в extend.json необходимо прописать указанную схему в поле display раздела managed для того чтобы пользовательский атрибут (UDF-поле) отображался в деталях массовых операций.

Например:

```
  ...
    "managed" : {
      "fieldType" : "textField",
      "display" : ["view", "viewBulkOperation"]
    }
```

Более подробное описание приведено в пункте 2.7.1.

2.7.1.2.6 Настройка отображения дополнительного атрибута в справочнике

Возможные типы udf-полей для отображения в справочниках: text, number, boolean, date, enum, lookup.

Настройка отображения атрибута в справочнике выполняется в конфигурационном файле extend.json для управляемого объекта «managed/referencebook». Общее описание настройки дополнительных атрибутов приведено в подразделе 2.7.

В параметре «managed»/«display» требуется указать два параметра «update» и «view».

Значение параметра «displayField» должно отличаться от названия самого атрибута.

Для настройки поиска параметр «searchable» должен иметь значение «true».

После выполнения настроек необходимо запустить периодическое задание «reindexTask».

Пример настройки отображения атрибута «customInformationSystemLookupM» типа «lookup» в справочнике:

```
{
  "customInformationSystemLookupM":{
    "repo":{
      "column":"custominformationsystemlookupm",
      "type":"BIGINT",
      "init":{
        "file":"conf/liquibase.json",
        "type":"liquibase/json"
      }
    },
    "policy":{
      "schema":{
        "type":"integer",
        "required":false
      }
    },
    "managed":{
      "searchable":true,
      "displayField":"customInformationSystemLookupMDf",
      "required":false,
      "fieldType":"lookup",
      "display":[
        "update",
        "view"
      ],
      "object":"managed/is",
      "searchStartLength":3,
      "resultsDisplayAmount":5,
      "lookupDisplayFields":[
        "name"
      ],
      "lookupDisplayFields2":[
        "description"
      ],
      "lookupField":"_ouid",
      "filter":"true"
    }
  }
}
```

2.7.2. Создание атрибутов в таблице БД с помощью скриптов Liquidbase

Для добавления атрибутов в таблицы БД администратору необходимо создать файл, на базе которого будет создана колонка для хранения значений дополнительного поля.

Файл со скриптом Liquidbase создается со следующими данными:

- стандартный блок XML Schema;
- «tableName» – название таблицы, в которой создается колонка;
- «schemaName» – по умолчанию принимает значение «ankey»;
- «column name» – название колонки, которое указывается в параметре «repo» в файле extend.json;
- «type» – тип, принятый для БД.

Администратору следует указать название таблицы, название колонки и тип при создании файла.

Пример скрипта, в котором создается колонка «orgunit» в таблице «organization»:

```
<?xml version="1.0" encoding="UTF-8"?>
<databaseChangeLog
      xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:ext="http://www.liquibase.org/xml/ns/dbchangelog-ext"
      xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.1.xsd
      http://www.liquibase.org/xml/ns/dbchangelog-ext http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-
ext.xsd">
   <changeSet id="100" author="gis">
      <addColumn tableName="organization" schemaName="ankey">
         <column name="orgunit" type="character varying(128)"/>
      </addColumn>
   </changeSet>
</databaseChangeLog>
```

2.7.3. Настройка локализации дополнительных полей

При создании дополнительного поля требуется задать для него локализацию. Локализация создается для отображения элементов интерфейса.

Комплекс позволяет настроить русскоязычную и англоязычную локализацию.

Администратору необходимо выполнить следующие шаги:

1) Шаг 1. Создать директорию localization.

2) Шаг 2. Создать директорию localization/i18n.

3) Шаг 3. В директории localization/i18n создать файлы для отображения названия создаваемого атрибута:

– файл translation_en.properties – для англоязычной локализации;

– файл translation_ru.properties – для русскоязычной локализации.

В файлах задаются следующие значения:

– ключ templates.managed.form – для окон редактирования и создания;

– ключ templates.managed.card – для страниц просмотра атрибутов.

Пример содержания файла для англоязычной локализации атрибута «customAttr»:

```
templates.managed.form.user.customAttr=Custom User Attribut

templates.managed.card.user.customAttr=Custom User Attribute
```

Для русскоязычной локализации требуется выполнить конвертацию текста для <Название атрибута в интерфейсе> в кодировку JS/JAVA.

Пример содержания файла русскоязычной локализации для атрибута «customAttr»:

```
templates.managed.form.user.customAttr=\\u041A\\u0430\\u0441\\u0442\\u043E\\u043C\\
u043D\\u044B\\u0439

\\u043F\\u043E\\u043B\\u044C\\u0437\\u043E\\u0432\\u0430\\u0442\\u0435\\u043B\\u044
C\\u0441\\u043A\\u0438\\u0439 \\u0430\\u0442\\u0440\\u0438\\u0431\\u0443\\u0442

templates.managed.card.user.customAttr=\\u041A\\u0430\\u0441\\u0442\\u043E\\u043C\\u
043D\\u044B\\u0439

\\u043F\\u043E\\u043B\\u044C\\u0437\\u043E\\u0432\\u0430\\u0442\\u0435\\u043B\\u044
C\\u0441\\u043A\\u0438\\u0439 \\u0430\\u0442\\u0440\\u0438\\u0431\\u0443\\u0442
```

В значениях локализации недопустимо использование круглых скобок.

4) Шаг 4. Создать Bundle локализации.

Bundle локализации представляет собой архив JAR. Внутри архива JAR содержатся следующие данные:

- каталог i18n с файлами translation_en.properties, translation_ru.properties;
- каталог META-INF с файлом MANIFEST.MF;
- файл manifest-addition.mf.

Создание Bundle локализации выполняется через системы сборки (например, maven, gradle, ant и другие) или утилиту jar.

При создании Bundle локализации необходимо находиться в папке localization.

Для создания Bundle администратору следует выполнить следующие действия:

а) Создать файл manifest-addition.mf с дополнительными настройками для манифест-файла следующего содержания:

```
Bundle-Name: Localization bundle
Bundle-SymbolicName: localization
Bundle-Version: 0.0.1
I18N: i18n.translation
```

Пример создания файла с дополнительными параметрами к манифест-файлу:

```
cd ../../
echo 'Bundle-Name: Localization bundle
Bundle-SymbolicName: localization
Bundle-Version: 0.0.1
I18N: i18n.translation' > manifest-addition.mf
```

б) Создать jar-файл localization.jar с дополнительными настройками для манифест-файла, выполнив команду *jar cfm localization.jar manifest-addition-path localization*, где «manifest-addition-path» – путь к файлу с дополнительными параметрами к манифесту, а localization – название директории, откуда будет собираться Bundle. Для выполнения команды в среде Windows следует использовать абсолютный путь к файлу, выполнив команду: *jar cfm localization.jar manifest-addition.mf localization.*

После выполнения команды, получен Bundle локализации: localization.jar.

Каталог META-INF создается автоматически и содержит в себе манифест MANIFEST.MF, созданный на основе manifest -addition.mf. Перед загрузкой Bundle в Ankey следует убедиться, что манифест MANIFEST.MF содержит в себе строку «I18N: i18n.translation».

5) Шаг 5. Загрузить Bundle локализации в Ankey.

Для загрузки файла localization.jar необходимо сохранить его в директорию ankey/extensions и перезагрузить Ankey. Подробная информация о перезапуске Комплекса приведена в документе «Руководство по инсталляции» (72410666.00054-02 94 01).

2.7.4. Настройка локализации существующего поля

Для настройки локализации существующего поля необходимо выполнить шаги, описанные в пункте 2.7.3.

Настройка локализации вкладки «Информация» возможна для следующих объектов в соответствующем конфигурационном файле:

1) «Входящие»/«Заявки», «Входящие»/«История», «Мои заявки», «Журнал заявок» - templates.request.detail.tabs.info.
2) «Подразделения» - templates.modules.organizations.detail.tabs.info.
3) «Инциденты» - templates.modules.incident.detail.tabs.info.
4) «Инф.системы» - templates.modules.is.detail.tabs.info.
5) «Админ.системы»:
   - «Ресурсы» - templates.modules.administration.resource.tab.detail.info;
   - «Правила» - templates.modules.administration.rules.detail.tabs.info;
   - «Парольные политики» - templates.modules.administration.passwordPolicies.tab.detail.info.

В файлах локализации следует указать конфигурационный файл объекта и его новое значение, как показано на примерах ниже.

Пример англоязычной локализации для вкладки «Информация» в файле translation_en.properties:

```
templates.request.detail.tabs.info=About
templates.modules.organizations.detail.tabs.info=Info
templates.modules.incident.detail.tabs.info=My info
```

templates.modules.is.detail.tabs.info=Custom info
templates.modules.administration.resource.tab.detail.info=My custom info
templates.modules.administration.passwordPolicies.tab.detail.info=Info about object
templates.modules.administration.rules.detail.tabs.info=New name info

Пример русской локализации для вкладки «Информация» в файле translation_ru.properties:

templates.request.detail.tabs.info=\u041e\u0431 \u043e\u0431\u044a\u0435\u043a\u0442\u0435
templates.modules.organizations.detail.tabs.info=\u0418\u043d\u0444\u043e
templates.modules.incident.detail.tabs.info=\u041c\u043e\u044f
\u0438\u043d\u0444\u043e\u0440\u043c\u0430\u0446\u0438\u044f
templates.modules.is.detail.tabs.info=\u043f\u043e\u043b\u044c\u0437\u043e\u0432\u0430\u0442\u0435\u043b\u044c\u0441\u043a\u0430\u044f \u0438\u043d\u0444\u043e\u0440\u043c\u0430\u0446\u0438\u044f
templates.modules.administration.resource.tab.detail.info=\u043c\u043e\u044f
\u043f\u043e\u043b\u044c\u0437\u043e\u0432\u0430\u0442\u0435\u043b\u044c\u0441\u043a\u0430\u044f
\u0438\u043d\u0444\u043e
templates.modules.administration.passwordPolicies.tab.detail.info=\u0418\u043d\u0444\u043e \u043e\u0431
\u043e\u0431\u044a\u0435\u043a\u0442\u0435
templates.modules.administration.rules.detail.tabs.info=\u041d\u043e\u0432\u043e\u0435 \u0438\u043c\u044f
\u043f\u043e\u043b\u044f \u0438\u043d\u0444\u043e

2.8. Настройка безопасности

2.8.1. Настройка безопасного подключения к web-консолям

Для настройки безопасного подключения к web-консолям администратору необходимо выполнить следующие действия:

1) В конфигурационном файле jetty.xml закомментировать или удалить блок кода «Call name = "addConnector";», который включает свойство openidm.port.http.
2) Оставить блоки кода «<Call name = "addConnector">», которые содержат свойства openidm.port.https и openidm.port.mutualauth. Значения для этих свойств задаются в файле conf/boot/boot.properties.
3) В конфигурационном файле config.properties установить для свойства «org.osgi.service.http.enabled» значение «false», как показано в следующем примере:

# Enable pax web http/https services to enable jetty
org.osgi.service.http.enabled=false
org.osgi.service.http.secure.enabled=true

2.8.2. Настройка схем аутентификации

Поддерживаемые системой схемы аутентификации пользователя в порядке очередности проверки содержатся в файле authentication.json в каталоге ankey/conf. В случае, когда схема аутентификации была применена для пользователя, остальные по порядку схемы не применяются.

Пример настроек файла authentication.json:

```
{
   "serverAuthContext" : {
      "sessionModule" : {
         "name" : "JWT_QUERY_STATUS_WRAPPER",
         "properties" : {
            "keyAlias" : "openidm-localhost",
            "privateKeyPassword" : "&{openidm.keystore.password}",
            "keystoreType" : "&{openidm.keystore.type}",
            "keystoreFile" : "&{openidm.keystore.location}",
            "keystorePassword" : "&{openidm.keystore.password}",
            "maxTokenLifeMinutes" : "540",
            "tokenIdleTimeMinutes" : "30",
            "sessionOnly" : true,
            "status" : {
               "queryId" : "for-userName",
               "queryOnResource" : "managed/user",
               "authenticationId" : "uid",
               "expired": [
                  {
                     "allowedUrl": "/endpoint/user",
                     "allowedAction": "changePassword"
                  },
                  {
                     "allowedUrl": "/endpoint/policy/user/password",
                     "allowedAction": "validate"
                  }
               ]
            }
         }
      },
      "authModules" : [
         {
            "name" : "STATIC_USER",
            "properties" : {
               "queryOnResource" : "repo/internal/user",
               "username" : "anonymous",
               "password" : "anonymous",
               "defaultUserRoles" : [ "ankey-reg" ]
            },
            "enabled" : true
         },
         {
            "name" : "TRUSTED_ATTRIBUTE",
            "properties" : {
               "queryOnResource" : "managed/user",
```

```
      "propertyMapping" : {
         "authenticationId" : "userName",
         "userRoles" : "roles"
      },
      "defaultUserRoles" : [ ],
      "authenticationIdAttribute" : "X-Ankey-AuthenticationId"
   },
   "enabled" : false
},
{
   "name" : "MANAGED_USER",
   "properties" : {
      "queryId" : "credential-query",
      "queryOnResource" : "managed/user",
      "propertyMapping" : {
         "authenticationId" : "username",
         "userCredential" : "password",
         "expired": [
            {
               "allowedUrl": "/endpoint/user"
            },
            {
               "allowedUrl": "/endpoint/policy/user/password"
            }
         ],
         "userRoles" : "roles"
      },
      "defaultUserRoles" : [ ]
   },
   "enabled" : true
},
{
   "name" : "INTERNAL_USER",
   "properties" : {
      "queryId" : "credential-internaluser-query",
      "queryOnResource" : "repo/internal/user",
      "propertyMapping" : {
         "authenticationId" : "username",
         "userCredential" : "password",
         "expired": [
            {
               "allowedUrl": "/endpoint/user"
            },
            {
               "allowedUrl": "/endpoint/policy/user/password"
            }
         ],
         "userRoles" : "roles"
      },
      "defaultUserRoles" : [ ]
   },
   "enabled" : true
},
{   "name": "IWA",
   "properties": {
      "servicePrincipal": "HTTP/idm.example.loc",
      "keytabFileName": "/etc/idm.keytab",
```

```
            "kerberosRealm": "EXAMPLE.LOC",
            "kerberosServerName": "AD.example.loc",
            "queryOnResource": "connector/Active_Directory/account",
            "augmentSecurityContext" : {
               "type" : "text/javascript",
               "file" : "auth/populateAsManagedUser.js"
            },
            "propertyMapping":
            {
               "authenticationId": "sAMAccountName",
               "groupMembership": "memberOf"
            },
            "groupRoleMapping":
            {
               "openidm-admin": []
            },
            "groupComparisonMethod": "ldap",
            "defaultUserRoles": ["ankey-authorized"]
         },
         "enabled": true
      }
   ]
  }
}
```

Схемы аутентификации, поддерживаемые Комплексом приведены в таблице 2.26.

Таблица 2.26 – Схемы аутентификации, поддерживаемые Комплексом

Наименование	**Назначение**
JWT_QUERY_STATUS_WRAPPER	Схема аутентификации по токену сессии JSON Web Tokens (JWT)
STATIC_USER	Схема использует механизм аутентификации для доступа анонимных пользователей
TRUSTED_ATTRIBUTE	Схема использует механизм сквозной аутентификации по заголовку HTTP-запроса, в котором передается логин пользователя системы
MANAGED_USER	Схема использует механизм аутентификации, который ведет поиск по БД, в частности, запрашивает объекты managed/user и разрешает аутентификацию, если учетные данные совпадают
INTERNAL_USER	Схема используется для аутентификации внутренних(служебных) пользователей Комплекса, в частности пользователя «ANKEY»
IWA	Схема используется для аутентификации пользователей с помощью механизма Kerberos, например, в домене службы каталога Active Directory

Для настройки схем аутентификации администратору необходимо в конфигурационном файле authentication.json выполнить следующие действия:

1) Добавить схемы аутентификации в секцию «authModules» файла настроек, если они отсутствуют.
2) Установить порядок применения схем аутентификации, указав их последовательно в секции «authModules».
3) Активировать схему аутентификации, установив параметр «enabled» в значение «true».

2.8.2.1. Настройка JSON Web Token аутентификации

Конфигурация JSON Web Token (JWT) аутентификации задаётся в файле authentication.json в блоке «sessionModule» со следующими параметрами:

- «keyAlias» – алиас для используемой пары ключ/сертификат;
- «privateKeyPassword» – пароль от keystore;
- «keystoreType» – тип keystore;
- «keystoreFile» – путь к keystore-файлу;
- «keystorePassword» – пароль от keystore-файла;
- «maxTokenLifeMinutes» – максимальное время жизни выданного токена (в минутах);
- «maxTokenLifeSeconds» – максимальное время жизни выданного токена (в секундах);
- «tokenIdleTimeMinutes» – время простоя, после которого токен признаётся недействительным (в минутах);
- «tokenIdleTimeSeconds» – время простоя, после которого токен признаётся недействительным (в секундах).

Необходимо учитывать следующие условия:

1) Время жизни токена:
   - нельзя указывать одновременно оба параметра: maxTokenLifeSeconds и maxTokenLifeMinutes;
   - если не указан ни один из параметров, время жизни токена считается 0 секунд.

2) Время простоя:

- нельзя указывать одновременно оба параметра: tokenIdleTimeSeconds и tokenIdleTimeMinutes;
- если не указан ни один из параметров, время простоя считается 0 секунд;
- если выполняемый пользователем запрос происходит в течение минуты после выдачи токена, период простоя не сбрасывается.

Пример настройки JSON Web Token аутентификации:

```
"sessionModule" : {
   "name" : "JWT_QUERY_STATUS_WRAPPER",
   "properties" : {
      "keyAlias" : "openidm-localhost",
      "privateKeyPassword" : "&{openidm.keystore.password}",
      "keystoreType" : "&{openidm.keystore.type}",
      "keystoreFile" : "&{openidm.keystore.location}",
      "keystorePassword" : "&{openidm.keystore.password}",
      "maxTokenLifeMinutes" : "120",
      "tokenIdleTimeMinutes" : "30",
      "sessionOnly" : true,
      "status" : {
         "queryId" : "for-userName",
         "queryOnResource" : "managed/user",
         "authenticationId" : "uid",
         "expired": [
            {
               "allowedUrl": "/endpoint/user",
               "allowedAction": "changePassword"
            },
            {
               "allowedUrl": "/endpoint/policy/user/password",
               "allowedAction": "validate"
            }
         ]
      }
   }
}
```

2.8.2.2. Настройка схемы аутентификации IWA

Параметры для настройки схемы аутентификации IWA приведены в таблице 2.27.

Таблица 2.27 – Параметры для настройки схемы аутентификации IWA

Наименование параметра	Описание атрибута	Пример
servicePrincipal	Идентификатор для	"HTTP/idm.example.loc"

Наименование параметра	Описание атрибута	Пример
	аутентификации сервера приложений Комплекса в следующем формате: HTTP/host.domain@DC-DOMAIN-NAME	
keytabFileName	Полный путь к файлу keytab, с ключом Keberos сервера приложений Комплекса	"/etc/idm.keytab"
kerberosRealm	Адрес службы центра распространения ключей Kerberos (KDC). Для службы Windows Kerberos это имя домена	"EXAMPLE.LOC"
kerberosServerName	Полное доменное имя сервера центра распространения ключей Kerberos, например, сервера контроллера домена	"AD.example.loc"
queryOnResource	Запрос для выборки УЗ из ресурса, с которым будет производиться сравнение в формате «connector//account». Где имя ресурса должно соответствовать ресурсу для службы каталога, например, Active Directory	"connector/Active_Directory/account"

augmentScriptContext	Скрипт, который выполняется при успешной аутентификации для обновления контекста безопасности (например, контекст был заполнен дынными из внешней системы (AD) и необходимо на основе этих данных заполнить контекст (или вызвать исключение) данными пользователя	{
		"type" : "text/javascript",
		"file" : "auth/populateAsManagedUser.js"
		}

propertyMapping	Атрибуты ресурса службы каталога, значения которых используются в процессе аутентификации внутри Комплекса:«authenticationId» - идентификатор пользователя, по которому будет выполняться поиск УЗ в ресурсе службы каталога. Например, если задан атрибут «sAMAccountName» в качестве идентификатора пользователя, то в процессе аутентификации в службе каталога Active Directory будет искаться пользователь, у которого значение «sAMAccountName»	{"authenticationId": "sAMAccountName","groupMembership": "memberOf"}

Наименование параметра	Описание атрибута	Пример
	совпадает со значением полученным в процессе Керберос аутентификации.«groupMembership» - атрибут в службе каталога, который содержит присвоенные пользователю группы	
groupRoleMapping	Атрибут описывает какие группы службы каталога присвоенные пользователю в случае успешной аутентификации в Комплексе предоставят внутренние административные роли (приведены в пункте 2.8.5)	{"ankey-admin": ["cn=Administrators,cn=Users,dc=example, dc=loc"]}
groupComparisonMethod	Метод сравнения групп пользователя в службе каталога. По умолчанию принимает значение «ldap»	"ldap"
defaultUserRoles	Внутренние административные роли Комплекса (приведены в пункте 2.8.5), присваиваемые пользователю, который успешно прошел аутентификацию	["ankey-authorized"]

Для работы схемы аутентификации IWA администратор должен выполнить следующие действия:

1) На контроллере домена создать УЗ компьютера для сервера приложений Комплекса.
2) На контроллере домена сформировать ключ аутентификации в файле keytab для компьютера сервера приложений Комплекса в домене. При создании ключа задается идентификатор для аутентификации сервера приложений Комплекса «servicePrincipalName».
3) Настроить клиента для аутентификации Kerberos в файле /etc/krb5.conf на сервере приложений Комплекса.
4) Настроить объекты «Коннектор», «Тип ресурса» и «Ресурс» для сервера контроллера домена службы каталога, в которой будет выполняться аутентификация Kerberos.
5) Задать необходимые значения для параметров в файле authentication.json.

6) Настроить web-браузеры на встроенную доменную аутентификацию для адреса компьютера сервера приложений Комплекса.

При настройке схемы аутентификации IWA необходимо учитывать следующие условия:

1) Сетевые адреса компьютера сервера приложений, контроллера домена и рабочей станции пользователя должны корректно определяться по короткому и длинному доменному имени.
2) Если необходимо игнорировать регистр при поиске идентификатора пользователя в службе каталога (использование фильтра eqic), то при создании объекта «Коннектор» для службы каталога необходимо указать следующий параметр: «enableCaseInsensitiveFilter»: true.
3) Для работы с логинами в службе каталога, заведенными в кириллице, необходимо добавить значение в переменную окружения JAVA_OPTS «-Dsun.security.krb5.msinterop.kstring=true». Переменная может задаваться для пользователя из-под которого запускается сервер приложений Комплекса или в сам скрипт запуска сервера приложений.
4) Должен быть открыт сетевой доступ с сервера Ankey IDM к следующим портам службы центра распространения ключей Kerberos (KDC): 88 и 464.
5) Если пользователю назначено много групп Active Directory, то возможно возникновение ошибки, связанно с размером заголовка запроса (Request Header). При возникновении ошибки необходимо в файле настроек conf/jetty.xml увеличить значение параметра «requestHeaderSize», например, до значения 32768 (по умолчанию значение выставлено в 8192).

2.8.2.2.1 Пример настройки схемы аутентификации IWA для одного домена

Сетевые адреса компьютера сервера приложений, контроллера домена и рабочей станции пользователя должны корректно определяться по короткому и длинному доменному имени. Пример, для локального файла hosts:

1) Сервер приложений Комплекса. Содержимое файла "/etc/hosts: "10.0.101.55 exch.lab.local exch", где exch доменное имя контроллера домена.
2) Рабочая станция пользователя, включенная в домен. Содержимое файла "hosts": "10.0.101.26 idm.lab.local idm", где idm доменное имя сервера приложений Комплекса.

На контроллере домена администратору необходимо выполнить следующие действия:

1) Создать УЗ компьютера, в примере «idm». Для службы каталога «Active Directory пользователи и компьютеры» перейти в домен/Компьютеры.
2) Выполнить операцию «Создать» для объекта «Компьютер», указав имя «idm». Пользователя с именем «idm» не должно быть в этом домене.
3) Создать ключ keytab, выполнив команду: "ktpass -princ HTTP/idm.gmt.loc@GMT.LOC -mapuser idm$@GMT.LOC -crypto ALL -ptype KRB5_NT_SRV_HST +rndpass -out c:\idm.keytab". Аргументы команды содержат:
   – "<idm$@GMT.LOC>" – имя компьютера в службе каталога;
   – "+rndpass" – пароль, который будет сгенерирован для УЗ компьютера.
4) Выбрать «y» в диалоге о сбросе пароля для объекта компьютер "Reset IDM$'s password [y/n]?". Контейнер Organization Unit (OU), где будет создан компьютер, и верхние OU должны называться латиницей.
5) Проверить привязку spn (service principal name) к УЗ компьютера можно используя команду: "setspn -Q HTTP/idm.gmt.loc". Если HTTP/idm.gmt.loc привязана к нескольким компьютерам или пользователям, то аутентификация Kerberos работать не будет.

На сервере приложений Комплекса администратору необходимо выполнить следующие действия:

1) Настроить клиента для аутентификации Kerberos в файле /etc/krb5.conf. Пример содержимого файла:

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
dns_lookup_kdc = true
dns_lookup_realm = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
rdns = false
default_realm = GMT.LOC
default_ccache_name = KEYRING:persistent:%{uid}

[realms]
GMT.LOC = {
kdc = 10.0.101.17
admin_server = 10.0.101.17
}

[domain_realm]
.gmt.loc = GMT.LOC
gmt.loc = GMT.LOC
```

2) Проверить корректность настройки клиента для аутентификации Kerberos, выполнив последовательно команды:
   - "kinit <administrator@GMT.LOC>" – проверяет успешность подключения доменного пользователя по Kerberos;
   - "klist" – выводит список текущих тикетов аутентификации;
   - "kdestroy" – удаляет тикеты аутентификации.
3) Проверить корректность сгенерированного ключа из файла «keytab» следующей командой: "klist -ke /etc/idm.keytab".
4) Проверить корректность аутентификации доменного пользователя с помощью файла «keytab» следующей командой: "kinit -V -k -t /etc/idm.keytab HTTP/idm.gmt.loc@GMT.LOC".
5) Настроить схему аутентификации IWA в файле authentication.json. Пример содержимого файла:

```
"serverAuthContext": {
.....
   "authModules" : [
   ..............
      {    "name": "IWA",
         "properties": {
            "servicePrincipal": "HTTP/idm.gmt.loc",
```

```
        "keytabFileName": "/etc/idm.keytab",
        "kerberosRealm": "GMT.LOC",
        "kerberosServerName": "AD.gmt.loc",
        "queryOnResource": "connector/Active_Directory/account",
        "augmentSecurityContext" : {
          "type" : "text/javascript",
          "file" : "auth/populateAsManagedUser.js"
      },
        "propertyMapping":
        {
        "authenticationId": "sAMAccountName",
        "groupMembership": "memberOf"
      },
      "groupRoleMapping":
      {
        "openidm-admin": []
      },
      "groupComparisonMethod": "ldap",
      "defaultUserRoles": ["ankey-authorized"]
    },
    "enabled": true
  }
 ]
}
}
```


На рабочей станции пользователя должен быть настроен web-браузер на встроенную доменную аутентификацию для адреса компьютера сервера приложений Комплекса. Например, для web-браузера Internet Explorer адрес подключения к серверу приложений Комплекса должен быть добавлен в сайты местной интрасети. Для web-браузера Chrome при запуске должны быть установлены следующие параметры: "--auth-server-whitelist="*.gmt .loc"", "--auth-negotiate-delegate-whitelist="*.gmt.loc"".

Детальное описание команд для настройки службы каталога, клиента аутентификации Kerberos и web-браузеров приведено в документации производителя.

2.8.2.2.2 Пример настройки схемы аутентификации IWA для нескольких доменов в одном лесу

В случае аутентификации пользователей из нескольких доменов в одном лесу настройка схемы аутентификации IWA может выполняться в следующих режимах:

1) Каждый домен леса задается отдельно в файле authentication.json. Аналогично в каждом домене создается компьютер для сервера приложений Комплекса, отдельный файл с ключом keytab. Пример содержимого файла файла authentication.json:

```
{
   "serverAuthContext": {
.....
       "authModules" : [
          .................
          {    "name": "IWA",
             "properties": {
                "servicePrincipal": "HTTP/idm.domain1.loc",
                "keytabFileName": "/etc/idm.keytab",
                "kerberosRealm": "DOMAIN1.LOC",
                "kerberosServerName": "AD.domain1.loc",
                "queryOnResource": "connector/Active_Directory1/account",
                "augmentSecurityContext" : {
                   "type" : "text/javascript",
                   "file" : "auth/populateAsManagedUser.js"
                },
                "propertyMapping":
                {
                   "authenticationId": "sAMAccountName",
                   "groupMembership": "memberOf"
                },
                "groupRoleMapping":
                {
                   "openidm-admin": []
                },
                "groupComparisonMethod": "ldap",
                "defaultUserRoles": ["ankey-authorized"]
             },
             "enabled": true
          },
          {    "name": "IWA",
             "properties": {
                "servicePrincipal": "HTTP/idm.domain2.loc",
                "keytabFileName": "/etc/idm.keytab",
                "kerberosRealm": "DOMAIN2.LOC",
                "kerberosServerName": "AD.domain2.loc",
                "queryOnResource": "connector/Active_Directory2/account",
                "augmentSecurityContext" : {
                   "type" : "text/javascript",
                   "file" : "auth/populateAsManagedUser.js"
                },
                "propertyMapping":
                {
                   "authenticationId": "sAMAccountName",
                   "groupMembership": "memberOf"
                },
                "groupRoleMapping":
                {
                   "openidm-admin": []
                },
                "groupComparisonMethod": "ldap",
                "defaultUserRoles": ["ankey-authorized"]
             },
             "enabled": true
          }
       ]
```

```
    }
}
```

2) В качестве сервера службы каталога используется сервер глобального каталога в лесе доменов Active Directory. В параметре объекта «Ресурс» для службы каталога Active Directory для такого сервера указывается порт «3269» и в качестве базового контекст («baseContext») каталога указывается весь домен "dc=<domain>,dc=<domain>", по которому выполняется поиск.

2.8.2.2.3 Настройка скрипта предзаполнения контекста безопасности augmentScriptContext

Настройка скрипта предзаполнения контекста безопасности augmentScriptContext выполняется для корректного поиска пользователя в Комплексе, в случае его успешной аутентификации по выбранной схеме аутентификации.

По умолчанию используется скрипт расположенный в каталоге сервера приложений Ankey "./ankey/bin/defaults/script/auth/populateAsManagedUser.js". Содержимое файла populateAsManagedUser.js из дистрибутива следующее:

```
var params = {
   "_queryId": "for-userName",
   "uid": security.authenticationId
};
var updatedSecurityContext;
var user;
var result = openidm.query("managed/user", params);
if (result.result && result.result.length === 1) {
   user = result.result[0];
   if(user){
      updatedSecurityContext =  {
         "authorization": {
            "id": user._id,
            "component": "managed/user",
            "roles": user.roles.split(",")
         },
         "authenticationId": user.userName
      };
   }
}
if(!user){
   throw {
      "code" : 404,
      "message" : "No matches found with principal name " + security.authenticationId
   };
}
updatedSecurityContext;
```

Скрипт выполняет поиск соответствующего пользователя в Комплексе по полю «userName» и значению, полученному в процессе аутентификации «security.authenticationId». Например, для схемы аутентификации IWA в качестве «authenticationId» возвращается атрибут службы каталога «sAMAccountName».

При необходимости выполнять поиск внутри Комплекса по другим атрибутам имеется возможность задать собственный фильтр в данном скрипте или создать новый, указав ссылку на него в файле «authentication.json». Пример содержимого файла для поиска по атрибуту должность «title»:

```
var filter = "title eq " + security.authenticationId;
var params = {
   "_queryFilter": <filter>
};
var updatedSecurityContext;
var user;
var result = openidm.query("managed/user", params);
if (result.result && result.result.length === 1) {
   user = result.result[0];
   if(user){
      updatedSecurityContext =  {
         "authorization": {
            "id": user._id,
            "component": "managed/user",
            "roles": user.roles.split(",")
         },
         "authenticationId": user.userName
      };
   }
}
if(!user){
   throw {
      "code" : 404,
      "message" : "No matches found with principal name " + security.authenticationId
   };
}
updatedSecurityContext;
```

2.8.2.3. Настройка схемы аутентификации openID connect

Конфигурация oauth2.0 и openid connect аутентификации задаётся в файле authentication.json в блоке «authModule» со следующими параметрами: * «name» - уникальное имя провайдера; * «type» - OAUTH или OPENID_CONNECT, зависит от настройки сервиса аутентификации; * «clientId» - уникальный идентификатор приложения, выдаваемый сервером аутентификации; * «clientSecret» - секретный ключ, выдаваемый сервером аутентификации; * «authorizationEndpoint» - адрес для

отправки запроса с целью получения уникального кода; * «tokenEndpoint» - адрес для отправки запроса с целью получения токена; * «userInfoEndpoint» - адрес для получения информации о пользователе.

Для отображения иконок на странице login Ankey IDM существует файл identityProviders.json.

Параметры: * «name» - уникальное имя провайдера, должно совпадать с именем провайдера в authentication.json; * «icon» - путь к иконке (обязательный); * «scope» - запрашиваемые разрешения, которые будут выданы приложению после аутентификации; * «enabled» - если true, то на главной странице отображается иконка, а если false, то иконка не отображается; * «clientId» - уникальный идентификатор приложения; * «authorizationEndpoint» - URL для запроса данных от провайдера; * «redirectUri» - URL, на который по умолчанию будет переадресован пользователь после прохождения авторизации; обратный адрес должен совпадать с redirectUri, указанным в приложении; * «popoverText» - текст иконки, который при наведении мышкой на иконку будет отображаться; * «responseType» - response type, которые будут доступны приложению при обращении к URL авторизации (authorization endpoint). Не обязательный атрибут, но если в приложении нет специальных указаний, то они должны совпадать.

Пример настройки openid connect аутентификации

«resolvers» - блок с описанием сервисов аутентификации

```
{
      "name" : "oidc",
      "type" : "OAUTH",
      "authenticationId" : "username",
      "enabled" : true,
      "clientId" : "ankeyidm",
      "clientSecret" : "***",
      "authorizationEndpoint" : "https://10.10.110.175/blitz/oauth/ae",
      "tokenEndpoint" : "https://10.10.110.175/blitz/oauth/te",
      "userInfoEndpoint" : "https://10.10.110.175/blitz/oauth/me"
}
```

Пример настройки identityProviders

```
{
  "providers" : [
   {
    "name" : "oidc",
```

```
      "icon" : "/assets/images/icons/login/icon_blitz_identity.svg",
      "scope" : [
         "profile"
      ],
      "enabled" : true,
      "clientId" : "ankeyidm",
      "authorizationEndpoint" : "https://10.10.110.175/blitz/oauth/ae",
      "redirectUri": "https://10.10.110.226:8080/",
      "popoverText" : "Blitz",
      "responseType" : "code"
   }
 ]
}
```

2.8.3. Настройка политики блокировки УЗ

По умолчанию для любого пользователя Ankey IDM включено неограниченное количество попыток ввода пароля при входе в систему. Такая настройка дает возможность злоумышленнику подобрать пароль пробным путем. Ankey IDM позволяет настроить отслеживание числа недопустимых попыток входа и может быть настроен на реагирование на этот вид возможной атаки путем отключения УЗ на заданный период времени. Параметры политики блокировки УЗ контролируют пороговое значение количества последовательных попыток входа с неверным паролем, а также действия, выполняемые по достижении порогового значения.

Параметры политики блокировки УЗ можно настроить в файле authenticated.lock.json, расположенном в директории ankey/config.

Настройка политики блокировки УЗ по умолчанию (политика блокировки УЗ отключена) в файле authenticated.lock.json:

```
{
  "badPwdCountLimit": 0,
  "lockoutTimePeriod": 0
}
```

Политика блокировки УЗ управляется двумя параметрами badPwdCountLimit и lockoutTimePeriod. Описание и возможные значения данных параметров приведены в таблице 2.28.

Таблица 2.28 – Параметры политики блокировки УЗ

Параметр	Название	Возможные значения	Описание
badPwdCountLimit	Порог блокировки УЗ	от 0 до 999999	Параметр политики «Порог блокировки учетной записи»

Параметр	Название	Возможные значения	Описание
			определяет количество последовательных неудачных попыток входа, которое приведет к блокировке УЗ пользователя. Допустимые значения: от 1 до 999999 попыток. Значение 0 указывает, что настроено неограниченное количество неудачных попыток входа; Пример: "badPwdCountLimit": 4 (четвертая неудачная попытка входа приведет к блокировке УЗ пользователя)
lockoutTimePeriod	Продолжительность блокировки УЗ	от 0 до 999999	Параметр политики Продолжительность блокировки УЗ определяет число минут, в течение которых УЗ остается заблокированной до ее автоматической разблокировки. Допустимые значения: от 1 до 999999 минут. Значение 0 указывает, что УЗ будет сразу разблокирована. Если для параметра Порог блокировки УЗ задано значение больше нуля, значение параметра Продолжительность блокировки УЗ должно быть больше или равно 1. Пример: "lockoutTimePeriod": 15 (УЗ будет оставаться заблокированной в течении 15 минут и автоматически разблокируется по истечении данного времени)

Примечания

1) Политика блокировки УЗ не распространяется на системные (тип internal) УЗ Ankey IDM.
2) Политику блокировки УЗ можно также настраивать REST-запросом. Ниже представлен пример REST-запроса настройки политики методом PUT.

Пример использования REST-запроса для настройки политики блокировки УЗ методом PUT:

```
C:\> curl
--cacert self-signed.crt ^
--header "Content-Type: application/json" ^
--header "X-Ankey-Username: ankey" ^
--header "X-Ankey-Password: ankey" ^
--request PUT ^
--data"

{

    "badPwdCountLimit": 3,
    "lockoutTimePeriod": 15

}" ^

'http://localhost:port/ankey/config/authenticated.lock'
```

В данном примере порог блокировки УЗ равен трем и продолжительность блокировки УЗ устанавливается равной 15 минутам. Для аутентификации используются учетные данные системного пользователя Ankey.

2.8.4. Настройка повторного использования идентификатора удалённого пользователя

По умолчанию в Ankey IDM включена возможность повторного использования идентификатора удалённого пользователя.

Возможность повторного использования идентификатора удалённого пользователя можно настроить в файле uniqueLogin.json, расположенном в директории ankey/config.

Ниже представлено содержание файла по умолчанию, возможность повторного использования идентификатора удалённого пользователя включена.

Пример настройки повторного использования идентификатора удалённого пользователя по умолчанию в файле uniqueLogin.json:

```
{
  "reuseUserLogin": true}
```

Параметр «reuseUserLogin» определяет возможность повторного использования идентификатора удалённого пользователя и может принимать значения «true» или «false». Значение «true» указывает, что возможность повторного использования идентификатора удалённого пользователя включена. Значение «false» указывает, что возможность повторного использования идентификатора удалённого пользователя отключена.

Примечание: Создание и изменение идентификатора пользователя с использованием идентификаторов удалённых пользователей невозможно.

2.8.5. Административные роли

Административные роли Комплекса предназначены для разграничения доступа к объектам и действиям с объектами. Административные роли назначаются пользователю системой автоматически при наступлении определенных условий.

Присвоенные административные роли перечисляются в атрибуте «roles» объекта «Пользователь» («managed/user»). Административные роли являются внутренним механизмом Комплекса, поэтому не отображаются в web-интерфейсе Комплекса.

Перечень административных ролей Комплекса приведен в таблице 2.29.

Таблица 2.29 – Перечень административных ролей Комплекса

Идентификатор роли	**Способ назначения**	**Полномочия роли**
ankey-admin	Присвоение роли «Ankey Admins»	Полные полномочия на все объекты системы
ankey-authorized	Присваивается всем пользователям в случае успешной аутентификации	Полномочия пользователя с доступом к пунктам меню «Входящие», «Мои заявки», «Мой профиль»
ankey-adminsod	Присваивается пользователям при следующих условия: – одна из ролей пользователя, указана в параметре «Роль оператора SoD» ИС; – одна из ролей пользователя, указана в параметре «Оператор SoD» правила разграничения доступа	Доступ к меню «Инциденты» и подменю «Входящие»/«Инциденты»
ankey-operator	Присваивается пользователям роли, указанной в параметре «Роль оператора» подразделения	Доступ к меню: – «Пользователи»; – «Подразделения»; – «Журнал заявок». Права на выполнение действий: – все действия с пользователями организации, для которой указан в качестве оператора; – все действия с подразделениями, которых входят в структуру организации, для которой указан в качестве оператора; – переназначение заявок для бенефициаров из организации, для которой указан в качестве оператора

Идентификатор роли	Способ назначения	Полномочия роли
ankey-approlesadmin	Присваивается пользователям при следующих условия: – одна из ролей пользователя, указана в параметре «Администратор ролей» ИС; – присвоена роль «Ankey Roles Managers»	Доступ к меню «Роли». Все права на действия с ролями в рамках своей ИС
ankey-assignmentsadmin	Присваивается пользователям роли, указанной в параметре «Администратор назначений» ИС	Доступ к меню «Пользователи» и к меню «Инф. системы» и права на просмотр деталей ИС, списка ролей и пользователей, с возможностью отзыва ролей в рамках ИС

2.8.6. Настройка сервиса проверки входящих соединений

В Комплексе используется балансировщик нагрузки с поддержкой взаимной аутентификации Transport Layer Security (TLS). При взаимной аутентификации TLS выполняется проверка сертификатов как клиента, так и сервера. Проверка подлинности клиента через взаимную TLS требует, чтобы сертификат включал в себя параметр Client Authentication. Эта схема проверки подлинности позволяет клиенту подтвердить свой сертификата в центре сертификации.

Когда балансировщик нагрузки обращается к Комплексу на определенный адрес, выполняется проверка переданного им сертификата. Сертификат содержит в себе имя субъекта, которое проверяется на наличие в белом списке. Белый список содержится в конфигурационном файле authenticated.whitelist.json. Файл расположен в каталоге ankey/conf. Когда имя субъекта найдено, балансировщику возвращается положительный ответ о том, что сервер доступен. В противном случае возвращается ответ о недоступности сервера.

Для проверки входящих соединений необходимо создать белый список с именами субъектов.

Администратору следует отредактировать конфигурационный файл authenticated.whitelist.json и задать параметр «values» типа массив, например:

```
{
"values": [
"CN=Administrator 01, CN=Users, DC=gis, DC=loc"
]
}
```

2.9. Настройка журналов работы

Журналирование системы производится отдельно от аудита. Для настройки системных журналов администратору необходимо отредактировать файл logging.properties, расположенный в каталоге ankey/conf.

2.10. Настройка аудита событий

Служба аудита Комплекса может записывать всю системную активность одной или нескольких ЦС, включая активности с локальными файлами данных, хранилищем Комплекса. Комплекс записывает данные о следующих событиях аудита:

1) События доступа.
2) События системной активности.
3) События аутентификации.
4) События изменений конфигурации.
5) События синхронизации.
6) События, связанные с заявками:
   - claim – взять в работу заявку;
   - unclaim – вернуть заявку из работы;
   - approved – согласовать заявку;
   - rejected – отклонить заявку;
   - delegate – вернуть заявителю для уточнения;
   - resolve – отправить возвращенную заявителю заявку на согласование;
   - reassign – переназначить согласующего заявки. Фиксируется от кого и на кого выполняется переназначение;
   - cancel – отменить заявку.

2.10.1. Настройка службы аудита

Комплекс предоставляет конфигурацию журналов аудита в файле, расположенном в каталоге ankey/conf/audit.json.

Чтобы настроить службу аудита для записи события, администратору необходимо включить его в список событий для обработчика событий аудита,

используемого для запросов. Администратору следует выбрать один любой обработчик событий аудита для управления запросами на журналах аудита. По умолчанию обработчиком запросов аудита является хранилище Комплекса.

Для того чтобы указать обработчик событий аудита, который должен использоваться для запросов, администратору необходимо указать параметр «handlerForQueries» в файле audit.json следующим образом:

```
{
  "auditServiceConfig" : {
    "handlerForQueries" : "repo",
    "availableAuditEventHandlers" : [
      "org.forgerock.audit.events.handlers.csv.CSVAuditEventHandler",
      "org.forgerock.openidm.audit.impl.RepositoryAuditEventHandler",
      "org.forgerock.openidm.audit.impl.RouterAuditEventHandler"
```

Параметр «availableAuditEventHandlers» содержит массив обработчиков событий аудита, доступных для Комплекса.

Обработчик событий аудита управляет событиями аудита, отправляет результат аудита в хранилище событий и контролирует их формат. По умолчанию Комплекс использует следующие типы обработчиков событий аудита:

1) «org.forgerock.audit.events.handlers.csv.CSVAuditEventHandler» – обработчик для хранения аудита в файлах формата CSV.
2) «org.forgerock.openidm.audit.impl.RepositoryAuditEventHandler» – обработчик для хранения аудита в БД Комплекса.
3) «org.forgerock.openidm.audit.impl.RouterAuditEventHandler» – обработчик для маршрутизации данных аудита во внешнюю систему.

Каждый обработчик событий аудита может включать в себя несколько параметров:

- «class» – имя класса, используемое для создания обработчика;
- «config» – объект JSON, который используется для настройки обработчика;
- «name» – имя на выбор;
- «logDirectory» – каталог с журналами;
- «topics» – события аудита, настроенные с этим обработчиком. В то время как свойство «config» определяет конфигурацию обработчика

событий, этот же параметр также используется как тема события аудита.

2.10.2. Настройка обработчика событий аудита в CSV-файле

CSV-обработчик событий аудита записывает события в CSV-файл. Пример настройки файла audit.json для обработчика событий аудита в CSV-файл:

```
"eventHandlers" : [
{
   "class" : "org.forgerock.audit.events.handlers.csv.CSVAuditEventHandler",
   "config" : {
      "name" : "csv",
      "logDirectory" : "&{launcher.working.location}/audit",
      "topics" : [ "access", "activity", "recon", "sync", "authentication", "config" ]
   }
}
```

Свойство «logDirectory» указывает имя каталога, в котором журналы должны быть записаны.

Для того чтобы направить журналы в другой файл, администратору необходимо использовать свойство подстановки значений. Пример настройки файла audit.json для записи аудита событий в пользовательский каталог:

```
{
   "logTo" : [
      {
         "logType" : "csv",
         "location" : "&{user.home}/audit"
      }
   ]
}
```

2.10.3. Описание типов журналов аудита

2.10.3.1. Журнал доступа

В журнал доступа Комплекс записывает сообщения, касающиеся доступа к Rest-интерфейсу.

Файл журнала доступа по умолчанию расположен в каталоге ankey/audit/access.csv.

Параметры журнала доступа представлены в таблице 2.30.

Таблица 2.30 – Параметры журнала доступа

Параметр	Описание
roles	Роли Комплекса, связанные с запросом
_id	Идентификатор для объекта сообщения, например «0419d364-1b3d-4e4f-b769-555c3ca098b0»
timestamp	Время, когда Комплекс зарегистрировал сообщение
eventName	Имя события аудита
transactionId	Идентификатор транзакции
userId	Идентификатор пользователя
trackingIds	Уникальное значение для отслеживаемого объекта
server.ip	IP-адрес для сервера Комплекса
server.port	Номер порта, используемый сервером Комплекса
client.ip	IP-адрес клиента
client.port	Номер порта клиента
request.protocol	Протокол для запроса, как правило, Rest
request.operation	Типичная Rest -операция
request.detail	Типичные детали для запроса действия
http.request.secure	Логическое значение для запроса безопасности
http.request.method	HTTP-метод, запрошенный клиентом
http.request.path	Путь HTTP-запроса
http.request.queryParameters	Параметры, отправленные в HTTP-запросе
http.request.headers	HTTP-заголовки для запроса (необязательно)
http.request.cookies	HTTP-куки для запроса (необязательно)
http.response.headers	HTTP-заголовки ответа (необязательно)
response.status	Normally, SUCCESSFUL, FAILED или null
response.statusCode	SUCCESS в response.status приводит к нулевому значению response.statusCode, FAILURE в response.status приводит к ошибке 400 уровня
response.detail	Сообщение, связанное с ответом response.statusCode, таким как Not Found или Internal Server Error
response.elapsedTime	Время для выполнения события доступа
response.elapsedTimeUnits	Единица времени отклика

2.10.3.2. Журнал активности

В журнал активности Комплекс записывает операции на внутренних и внешних объектах, включая события по работе с заявками.

Файл по умолчанию расположен в каталоге ankey/audit/activity.csv.

Параметры журнала активности представлены в таблице 2.31.

Таблица 2.31 – Параметры журнала активности

Параметр	Описание
passwordChanged	True/False запись об изменениях пароля
_id	Идентификатор для объекта сообщения, например, «0419d364-1b3d-4e4f-b769-555c3ca098b0»
timestamp	Время, когда Комплекс зарегистрировал сообщение
eventName	Имя события аудита
transactionId	Идентификатор транзакции
userId	Идентификатор пользователя
trackingIds	Уникальное значение для отслеживаемого объекта
runAs	Идентификатор пользователя, под УЗ которого выполнялось действие
objectId	Идентификатор объекта, например, /managed/user/0419d364-1b3d-4e4f-b769-555c3ca098b0
operation	Типичная Rest-операция
before	JSON-представление объекта перед выполнением действия
after	JSON-представление объекта после выполнения действия
changedFields	Поля, которые были изменены
revision	Число ревизий объекта
status	Результат (например, SUCCESS)
message	Сообщение о действии

2.10.3.3. Журнал синхронизации с ЦС

В журнал синхронизации с ЦС Комплекс записывает результаты выполнения синхронизации.

Файл по умолчанию расположен в каталоге ankey/audit/recon.csv.

Параметры журнала синхронизации с ЦС представлены в таблице 2.32.

Таблица 2.32 – Параметры журнала синхронизации с ЦС

Параметр	Описание
reconId	UUID для операции синхронизации
_id	Идентификатор для объекта сообщения, например, «0419d364-1b3d-4e4f-b769-555c3ca098b0»
transactionId	Идентификатор транзакции
timestamp	Время, когда Комплекс зарегистрировал сообщение
eventName	Имя события аудита

Параметр	Описание
userId	Идентификатор пользователя
trackingIds	Уникальное значение для отслеживаемого объекта
action	Действие синхронизации, отображено как Rest-действие
exception	Трассировка стека исключения
linkQualifier	Определитель связи, примененный к действию
mapping	Имя маппинга, используемое для операций синхронизации, заданных в файле ankey/conf/recon.json
message	Описание действия синхронизации
messageDetail	Детали запуска синхронизации, представленные в виде Rest -запроса
situation	Ситуация, возникающая при синхронизации
sourceObjectId	Идентификатор объекта в исходной системе, например, «managed/user/d03a29e8-c95f-4637-82f6-cc9b94d0dfb7»
status	Статус результата синхронизации, например, «SUCCESS» или «FAILURE»
targetObjectId	Идентификатор объекта в ЦС, например, «system/xmlfile/account/dkruglov»
reconciling	Значение того, что Комплекс синхронизировал: «source» для первого этапа и «target» - для второго
ambiguousTargetObjectIds	Когда параметр «situation» имеет значение «AMBIGUOUS» или «UNQUALIFIED», и Комплекс не может отличить более одного объекта-получателя, Комплекс регистрирует идентификаторы таких объектов
reconAction	Действие синхронизации (как правило «recon» или «null»)
entryType	Тип записи журнала синхронизации (как правило «start», «entry» или «summary»)

2.10.3.4. Журнал аутентификации

Комплекс записывает результаты операций аутентификации в журнал аутентификации.

Файл по умолчанию расположен в каталоге ankey/audit/authentication.csv.

Параметры журнала аутентификации представлены в таблице 2.33.

Таблица 2.33 – Параметры журнала аутентификации

Параметр	Описание
entries	JSON-представление сессии аутентификации
_id	Идентификатор для объекта сообщения, например, «0419d364-1b3d-4e4f-b769-555c3ca098b0»
timestamp	Время, когда Комплекс зарегистрировал сообщение

Параметр	Описание
eventName	Имя события аудита
transactionId	Идентификатор транзакции
userId	Идентификатор пользователя
trackingIds	Уникальное значение для отслеживаемого объекта
result	Результат транзакции («SUCCESSFUL» или «FAILED»)
principal	Массив УЗ, используемых для аутентификации, например, «ankey»
context	Полная безопасность операции аутентификации, включая аутентификацию ID, целевой конечной точки, ролей, IP-адресов, с которых был сделан запрос аутентификации

2.10.3.5. Журнал конфигурации

В журнал конфигурации Комплекс записывает изменения, выполненные в настройках Комплекса.

Файл по умолчанию расположен в каталоге ankey/audit/config.csv.

Параметры журнала конфигурации представлены в таблице 2.34.

Таблица 2.34 – Параметры журнала конфигурации

Параметр	Описание
revision	Число ревизий объекта
_id	Идентификатор для объекта сообщения, например, «0419d364-1b3d-4e4f-b769-555c3ca098b0»
timestamp	Время, когда Комплекс зарегистрировал сообщение
eventName	Имя события аудита
transactionId	Идентификатор транзакции
userId	Идентификатор пользователя
trackingIds	Уникальное значение для отслеживаемого объекта
runAs	Идентификатор пользователя, под УЗ которого выполнялось действие
objectId	Идентификатор объекта, например, «ui»
operation	Типичная Rest -операция
before	JSON-представление объекта до выполнения действия
after	JSON-представление объекта после выполнения действия
changedFields	Поля, которые были изменены

2.10.3.6. Журнал инкрементальной синхронизации с ЦС

В журнал инкрементальной синхронизации с ЦС Комплекс записывает результаты выполнения инкрементальной синхронизации.

Файл по умолчанию расположен в каталоге ankey/audit/sync.csv.

Параметры журнала инкрементальной синхронизации с ЦС представлены в таблице 2.35.

Таблица 2.35 – Параметры журнала инкрементальной синхронизации с ЦС

Параметр	Описание
targetObjectId	Идентификатор объекта в ЦС, например, «uid=dkruglov,ou=People,dc=example,dc=com»
_id	Идентификатор для объекта сообщения, например, «0419d364-1b3d-4e4f-b769-555c3ca098b0»
transactionId	Идентификатор транзакции
timestamp	Время, когда Комплекс зарегистрировал сообщение
eventName	Имя события аудита
userId	Идентификатор пользователя
trackingIds	Уникальное значение для отслеживаемого объекта
action	Действие синхронизации, отображено как Rest-действие
exception	Трассировка стека исключения
linkQualifier	Определитель связи, примененный к действию
mapping	Имя маппинга, используемое для операций синхронизации, заданных в файле ankey/conf/sync.json
message	Описание действия синхронизации
messageDetail	Детали запуска синхронизации, представленные в виде Rest -запроса
situation	Ситуация синхронизации
sourceObjectId	Идентификатор объекта в исходной системе, например, «managed/user/d03a29e8-c95f-4637-82f6-cc9b94d0dfb7»
status	Статус результата синхронизации (например, «SUCCESS» или «FAILURE»)

2.10.4. Настройка обработчика событий аудита «маршрутизатор»

Обработчик событий аудита «маршрутизатор» записывает события на любой внешней или пользовательской конечной точке, такой как connector/ad.

Пример конфигурации обработчика событий аудита «маршрутизатор» в audit.json:

```
"eventHandlers" : [
      {
         "class": "org.forgerock.openidm.audit.impl.RouterAuditEventHandler",
         "config": {
            "name": "router",
            "topics" : [ "access", "activity", "recon", "sync", "authentication", "config" ],
```

```
        "resourcePath" : "connector/auditdb"
      }
    }
  ]
```

Параметр «resourcePath» в конфигурации указывает на то, что журналы должны быть направлены в конечную точку connector/auditdb. Для каждого события конечная точка формируется в формате «resourcePath/событие». Например, для события access и resourcePath «connector/auditdb» конечная точка будет иметь следующий вид: «connector/auditdb/access»

2.10.4.1. Пример настройки маршрутизации аудита во внешнюю БД

Пример настройки маршрутизации аудита во внешнюю БД приведен для универсального коннектора SQL и включает следующие действия:

1) Настройка типа ресурса.
2) Настройка объекты «connector».
3) Настройка ресурса.
4) Настройка файла запроса в БД.

Пример настройки типа ресурса:

```
{
  "resTypeName": "dbtType",
  "resTypeDesc": "DBT Type",
  "resTypeBody": {
    "properties": {
      "user": {
        "type": "string",
        "default": ""
      },
      "password": {
        "type": "string",
        "default": ""
      },
      "jdbcDriver": {
        "type": "string",
        "default": ""
      },
      "url": {
        "type": "string",
        "default": ""
      },
      "configurationFileName": {
        "type": "string",
        "default": ""
      },
      "logColumnName": {
        "type": "string",
```

```
            "default": ""
          }
        },
        "private": ["password"],
        "order": ["user", "password", "jdbcDriver", "url", "configurationFileName", "logColumnName"]
    }
}
```

Пример настройки объекта «connector»:

```
{
   "connectorName": "auditdb",
   "connectorDesc": "org.identityconnectors.databasetable.DatabaseTableConnector",
   "connectorBody": {
      "name": "databaseTable",
      "objectTypes": {
         "access": {
            "_id": "ACCESS",
            "type": "object",
            "$schema": "http://json-schema.org/draft-03/schema",
            "nativeType": "ACCESS",
            "properties": {
               "_id": {
                  "type": "string",
                  "required": false,
                  "nativeName": "OBJECTID",
                  "nativeType": "string"
               },
               "roles": {
                  "flags": ["MULTIVALUED"],
                  "type": "string",
                  "required": false,
                  "nativeName": "ROLES",
                  "nativeType": "string"
               },
               "timestamp": {
                  "type": "JAVA_TYPE_DATE",
                  "required": false,
                  "nativeName": "TIMESTAMP",
                  "nativeType": "JAVA_TYPE_DATE"
               },
               "userId": {
                  "type": "string",
                  "required": false,
                  "nativeName": "USERID",
                  "nativeType": "string"
               },
               "transactionId": {
                  "type": "string",
                  "required": false,
                  "nativeName": "TRANSACTIONID",
                  "nativeType": "string"
               },
               "eventName": {
                  "type": "string",
                  "required": false,
```

```
            "nativeName": "EVENTNAME",
            "nativeType": "string"
          },
          "client": {
            "type": "object",
            "required": false,
            "nativeName": "CLIENT",
            "nativeType": "object"
          },
          "server": {
            "type": "object",
            "required": false,
            "nativeName": "SERVER",
            "nativeType": "object"
          },
          "http": {
            "type": "object",
            "required": false,
            "nativeName": "HTTP",
            "nativeType": "object"
          },
          "request": {
            "type": "object",
            "required": false,
            "nativeName": "REQUEST",
            "nativeType": "object"
          },
          "response": {
            "type": "object",
            "required": false,
            "nativeName": "RESPONSE",
            "nativeType": "object"
          }
        }
      }
  },
  "connectorRef": {
    "bundleName": "com.gis.openicf.database",
    "bundleVersion": "0.2-SNAPSHOT",
    "connectorName": "com.gis.openicf.database.DatabaseConnector"
  },
  "operationOptions": {},
  "operationTimeout": {
    "GET": -1,
    "SYNC": -1,
    "TEST": -1,
    "CREATE": -1,
    "DELETE": -1,
    "SCHEMA": -1,
    "SEARCH": -1,
    "UPDATE": -1,
    "VALIDATE": -1,
    "AUTHENTICATE": -1,
    "SCRIPT_ON_RESOURCE": -1,
    "SCRIPT_ON_CONNECTOR": -1
  },
  "poolConfigOption": {
```

```
        "maxIdle": 1,
        "maxWait": 600000,
        "minIdle": 1,
        "maxObjects": 1,
        "minEvictableIdleTimeMillis": 120000
      },
      "producerBufferSize": 100,
      "syncFailureHandler": {
        "maxRetries": 5,
        "postRetryAction": "logged-ignore"
      },
      "connectorPoolingSupported": true
   }

}
```

Пример настройки ресурса:

```
{
   "restype_id": ${restypeId},
   "connector_id": ${connectorId},
   "resName": "auditdb",
   "resDesc": "Внешний аудит",
   "resBody": {
      "user": ${user},
      "password": ${password},
      "jdbcDriver": ${jdbcDriver},
      "url": ${url},
      "configurationFileName": "audit.txt",
      "logColumnName": ${logColumnName}
   }
}
```

Пример настройки файла запроса в БД (конфигурация DBT коннектора (audit.txt)):

```
CREATE_ACCESS
   Query="call ANKEY_AUDIT.AUDITPACKAGE.createAccess(?,?,?,?,?,?,?,?,?,?,?,?,?)"

Parameters=[OBJECTID:String:IN,ROLES:String:IN,TIMESTAMP:String:IN,USERID:String:IN,TRANSACTIO
NID:String:IN,CLIENT:String:IN,SERVER:String:IN,HTTP:String:IN,REQUEST:String:IN,RESPONSE:String:I
N,uid:String:OUT,error code:NUMBER:OUT,error message:String:OUT]

TEST
   Query="SELECT 1 FROM DUAL"
   Parameters=[no parameters]
```

2.10.5. Настройка обработчика событий аудита в БД Комплекса

Обработчик событий аудита в БД передает информацию в хранилище Комплекса.

В БД Комплекс хранит записи журнала в следующих таблицах:

– auditaccess;
– auditactivity;
– auditrecon;
– auditsync.

Пример настройки файла audit.json для обработчика событий аудита хранилища:

```
"eventHandlers" : [
      {
    "class": "org.forgerock.openidm.audit.impl.RepositoryAuditEventHandler",
   "config": {
      "name": "repo",
      "topics" : [ "access", "activity", "recon", "sync", "authentication", "config" ]
   }
]
```

2.10.6. Настройка буферизации записей аудита

Комплекс поддерживает буферизацию, чтобы уменьшить число записей в системе. Для настройки буферизации администратору необходимо отредактировать файл «audit.json» и задать следующие параметры:

– «enabled» – включает/выключает буферизацию (True/false);
– «autoFlush» – определяет, будет ли служба аудита автоматически удалять события после их записи на диск (True/false).

Пример настройки буферизации:

```
 "eventHandlers" : [
    {
     "config" : {
       ...
       "buffering" : {
         "autoFlush" : false,
         "enabled" : false
       }
     }
```

Комплекс записывает данные в журналы аудита асинхронно, в то время как функция «autoFlush» позволяет службе аудита записывать данные в журналы на регулярной основе.

2.10.7. Дополнительная настройка обработчика событий аудита

Для конфигурации обработчика событий аудита администратору необходимо перейти в каталог ankey/conf и отредактировать файл audit.json в подразделе «config» раздела «eventHandlers»:

```
"eventHandlers" : [
    {
      …
      "config" : {
            ……
    }
```

Свойства конфигурации для обработчика событий аудита CSV приведены в таблице 2.36.

Таблица 2.36 – Свойства обработчика событий аудита CSV

Свойство	Описание
delimiterChar	Символы разделения CSV полей
endOfLineSymbols	Форматирование: символ конца строки, например, «» или «
formatting	–
logDirectory	Каталог с файлами обработчика событий аудита CSV
quoteChar	Форматирование: символ, используемый вокруг поля CSV
buffering	Конфигурация для дополнительной буферизации событий
autoFlush	Управляет автоматической очисткой буфера
enabled	True или false для буферизации
fileRetention	Определяет, как долго хранить файл аудита
maxDiskSpaceToUse	Максимальный объем дискового пространства для файлов аудита
maxNumberOfHistoryFiles	Максимальное количество резервных копий файлов аудита
minFreeSpaceRequired	Минимальное дисковое пространство, необходимое системе с файлами аудита
fileRotation	Параметры ротации файлов аудита
maxFileSize	Максимальный размер файла в байтах перед ротацией
rotationEnabled	Включение и выключение файла ротации
rotationFilePrefix	Префикс файла после ротации
rotationFileSuffix	Суффикс добавляется в конце имен файлов аудита
rotationInterval	Период времени между ротацией журнала (ротация через интервал времени). Могут использоваться следующие значения: 5 seconds, 5 minutes, 5 hours, disabled
rotationRetentionCheckInterval	Интервал периодической проверки файла ротации и политик хранения. По умолчанию один раз в 5 секунд

Свойство	Описание
rotationTimes	Время после 00:00, когда будет вызвана ротация журналов (ротация в заданное время)
security	Использование подписи для событий аудита
enabled	True или false для использования подписи для событий аудита
filename	Путь к файлу хранилища ключей JAVA
password	Пароль для хранилища ключей JAVA
keystoreHandlerName	Имя хранилища ключей
signatureInterval	Интервал генерации подписей. По умолчанию 1 час. Допустимые значения: – days, day, d; – hours, hour, h; – minutes, minute, min, m; – seconds, second, sec, s. Например, 1 hour, 3 sec

Пример настройки конфигурации для обработчика событий аудита CSV:

```
"eventHandlers": [{
        "class": "org.forgerock.audit.handlers.csv.CsvAuditEventHandler",
         "config": {
              "name": "csv",
              "logDirectory": "&{launcher.working.location}/audit",
              "topics": ["access", "activity", "recon", "sync", "authentication", "config"],
              "fileRetention": {
                        "maxDiskSpaceToUse": 1000000,
                        "maxNumberOfHistoryFiles": 3,
                        "minFreeSpaceRequired": 0
                   },
              "fileRotation": {
                        "maxFileSize": 0,
                        "rotationEnabled": true,
                        "rotationFilePrefix": "prefix-",
                        "rotationFileSuffix": "_yyyy-MM-dd_kk:mm:ss",
                        "rotationInterval": "5 m",
                        "rotationTimes": [
                              "10 m"
                        ]
                   }
              }
        }
```

2.11. Настройка парольной политики

Настройка парольной политики выполняется в меню «Админ. системы»/«Парольные политики».

Администратор выполняет следующие действия:

– создание парольной политики. Описание приведено в пункте 2.11.1;

– редактирование парольной политики. Описание приведено в пункте 2.11.2;
– применение парольной политики. Описание приведено в пункте 2.11.3;
– удаление парольной политики. Описание приведено в пункте 2.11.4.

Парольная политика создается для следующих объектов:

1) «Пользователи».
2) «Учетная запись ресурсов».

Для объекта «Пользователи» создается одна парольная политика, которая автоматически применяется ко всем пользователям для входа в Комплекс. Политика будет действовать для всех новых пользователей. Для ранее созданных пользователей политика будет действовать только на смену пароля.

Для объекта «Учетная запись ресурсов» можно создавать разные варианты парольных политик:

1) Политика, которая автоматически применяется только к УЗ выбранных УЗР.
2) Политика, которая автоматически применяется ко всем УЗ существующих УЗР. Кроме тех, на которые действуют более приоритетные политики.
3) Политика, которая применяется к конкретной УЗ в УЗР.

Особенности применения парольных политик приведены в пункте 2.11.3.

Вариант парольной политики определяется с помощью флагов следующим образом:

1) Если флаги не установлены, будет создана парольная политика для применения к конкретным УЗ в выбранных УЗР.
2) Если установлен флаг «Применить ко всем УЗ пользователя в выбранных УЗ ресурсов», будет создана парольная политика для применения ко всем УЗ в выбранных УЗР.
3) Если установлен флаг «Применить ко всем УЗ ресурсов», будет создана общая парольная политика на все УЗР Комплекса.

Варианты парольных политик приведены в таблице 2.37.

Таблица 2.37 – Варианты парольных политик

Назначение политики	Объекты, к которым применяется политика	Флаги	Приоритет
Пользователи	Все пользователи	Отсутствуют	–
Учетная запись ресурса	Конкретные УЗ в выбранных УЗР	Не установлены	Высокий
Учетная запись ресурса	Все УЗ в выбранных УЗР	Флаг «Применить ко всем УЗ пользователя в выбранных УЗ ресурсов»	Средний
Учетная запись ресурса	Все УЗР Комплекса	Флаг «Применить ко всем УЗ ресурсов». Устанавливаются оба флага	Низкий

Список парольных политик содержит следующую информацию:

– название политики;

– назначение.

Детали парольных политик содержат следующие сведения:

1) «Данные»:
   – название политики;
   – назначение: «Пользователь» или «Учетные записи ресурсов»;
   – флаги «Применить ко всем УЗ пользователя в выбранных УЗР», «Применить ко всем УЗ ресурсов» установлен/нет. Отображаются только для парольной политики с назначением «Учетные записи ресурсов»;
   – число последних использованных паролей. Отображается, если было задано значение;
   – максимальное время действия пароля. Отображается, если было задано значение;
   – минимальное время действия пароля. Отображается, если было задано значение;
2) «Настройки политики» – отображаются выбранные значения политики из таблицы 2.38;
3) «Объекты» – отображаются объекты, для которых создана политика. Объект «Пользователь» с признаком «Все» или список УЗР.

Список правил для поддерживаемых парольных политик приведен в таблице 2.38.

Таблица 2.38 – Список поддерживаемых правил для парольных политик

Параметр	Назначение	Примечание
minLength	Минимальная длина пароля	–
maximumLength	Максимальная длина пароля	–
disallowUserLogin	Пароль не должен содержать логин пользователя	–
minAlphabetChars	Минимальное число букв в пароле	–
minNumericChars	Минимальное число цифр в пароле	–
minSpecialChars	Минимальное число специальных символов в пароле	–
minLowercaseChars	Минимальное число символов в нижнем регистре	–
minUppercaseChars	Минимальное число символов в верхнем регистре	–
startWithAlphabet	Пароль должен начинаться с буквы	–
cannotContainCharacters	Пароль не должен содержать перечисленные символы	Массив отображаемых символов, в кавычках, перечисленных через запятую. Например, ["@", "$", "1", "Я"]
minPwdAge	Минимальное время действия пароля (в сутках)	В случае смены пароля администратором для пользователя или УЗ ЦС пользователя (операция сброса пароля), политика не применяется
expirePeriod	Максимальное время действия пароля	Если срок действия пароля равен 0 или не установлен, считается, что срок его действия неограничен
minNewChars	Минимальное количество измененных символов при создании нового пароля	В случае пакетной смены пароля, политика применяется к каждой УЗ. В случае смены пароля администратором для пользователя или УЗ ЦС пользователя (операция сброса пароля), политика не применяется
passwordHistoryCount	Число последних использованных паролей	В случае смены пароля администратором для пользователя или УЗ ЦС пользователя (операция сброса пароля), политика не

Параметр	Назначение	Примечание
		применяется

Перед созданием/редактированием парольной политики следует учесть следующие моменты:

1) Нельзя создать более одной парольной политики по умолчанию на одну и ту же УЗР.
2) Нельзя создать более одной парольной политики с назначением «Пользователи». Назначение «Пользователи» будет недоступно при наличии в Комплексе такой политики.
3) Нельзя создать более одной парольной политики с флагом «Применить ко всем УЗ ресурсов». Флаг будет недоступен при наличии в Комплексе такой политики.
4) Название парольной политики должно быть уникальное. В противном случае отобразится подсказка «Такое значение уже существует».
5) Для всех числовых полей допустимо вводить целые значения от 1 до 2147483647 (включительно). Не допускается вводить формулы, отрицательные и дробные числовые значения. В противном случае в зависимости от формата вводимых значений они будут либо сброшены, либо числовое поле будет подсвечено красным цветом.

2.11.1. Создание парольной политики

Важные моменты, которые следует учесть перед созданием парольной политики, приведены в подразделе 2.11.

Для создания парольной политики с назначением «УЗР» администратору необходимо выполнить следующие действия:

1) Нажать кнопку «Создать». Откроется окно «Создание парольной политики».
2) Перейти на обязательную вкладку «Параметры» и выполнить:
   – заполнить обязательное поле «Название парольной политики». В противном случае отобразится подсказка «Не заполнено обязательное поле»;

– выбрать в обязательном поле «Назначение» значение «Учетные записи ресурсов»;
– заполнить при необходимости остальные поля.

3) Установить при необходимости один из двух флагов «Применить ко всем аккаунтам в выбранных УЗР» или «Применить ко всем УЗ ресурсов», чтобы политика применялась по умолчанию. Подробнее о политике по умолчанию приведено в пункте 2.11.3.
4) Перейти при необходимости на вкладку «Правила политики» и выполнить действия, приведенные в подпункте 2.11.1.1.
5) Перейти при необходимости на вкладку «Пользовательские правила» и выполнить действия, приведенные в подпункте 2.11.1.2.
6) Перейти на обязательную вкладку «Список УЗ ресурсов» и выбрать нужные УЗР. В противном случае отображается сообщение «Ошибка при создании парольной политики из-за отсутствия УЗР». Вкладка недоступна, если был установлен флаг «Применить ко всем УЗ ресурсов».
7) Нажать кнопку «Создать».

Следует учитывать следующее поведение:

1) Смена флага «Применить ко всем УЗ ресурсов» приводит к сбросу выбранных УЗР в списке. После чего требуется повторно выбрать УЗР на вкладке «Список УЗ ресурсов».
2) Поле «Назначение» и флаги будут недоступны при редактировании.

Для создания парольной политики с назначением «Пользователи» администратору необходимо выполнить следующие действия:

1) Нажать кнопку «Создать». Откроется окно «Создание парольной политики».
2) Перейти на обязательную вкладку «Параметры» и выполнить:
   – заполнить обязательное поле «Название парольной политики». В противном случае отобразится подсказка «Не заполнено обязательное поле»;

- выбрать в обязательное поле «Назначение» значение «Пользователи». При этом вкладка «Список УЗ ресурсов» станет недоступна;
- заполнить при необходимости остальные поля.

3) Перейти при необходимости на вкладку «Правила политики» и выполнить действия, приведенные в пункте 2.11.1.1.
4) Перейти при необходимости на вкладку «Пользовательские правила» и выполнить действия, приведенные в пункте 2.11.1.2.
5) Нажать кнопку «Создать».

2.11.1.1. Добавление правила политики для парольной политики

Добавление правил политики выполняется при создании или редактировании парольной политики.

Для добавления правил политики необходимо:

1) Нажать кнопку «Добавить правило».
2) Выбрать нужное правило из выпадающего списка. При добавлении правила «Пароль не должен содержать перечисленные символы» добавлять можно только по одному символу. Для ввода каждого следующего символа нужно нажимать кнопку «Ввод».
3) Ввести при необходимости в поле «Значение» числовое значение.
4) Нажать кнопку «Добавить».
5) Нажать кнопку «Сохранить».

2.11.1.2. Добавление пользовательских правил для парольной политики

Комплекс позволяет расширить существующие правила парольной политики.

Для этого требуется создать пользовательский java-класс. После этого выполнить добавление пользовательского правила, в котором следует описать свой алгоритм проверки пароля.

Добавление пользовательских правил выполняется при создании или редактировании парольной политики.

Для добавления пользовательских правил необходимо:

1) Перейти на вкладку «Пользовательские правила».

2) Нажать «Добавить правило».
3) Заполнить идентификатор правила. Указывается значение созданного пользовательского java-класса.
4) Заполнить блок с параметрами со своим алгоритмом проверки пароля.
5) Нажать кнопку «Добавить».

Возможные сообщения об ошибках при сохранении политики:

– «Невалидный JSON в пользовательских правилах» – при заполнении блока с параметрами была допущена синтаксическая ошибка;
– «Ошибка в пользовательских правилах» – идентификатор правила не соответствует созданному java-классу или не существует.

Пример создания java-класса для пользовательских правил парольной политики:

```
public class CannotBeNeighborsCharactersPolicy implements PasswordPolicy
```

Пример блока параметров со своим алгоритмом проверки пароля:

```
{code:java}
{
  "neighbourChars": [
  "qwerty",
  "123456"
        ],
  "numChars": 5
}
{code}
```

2.11.2. Редактирование парольной политики

Важные моменты, которые следует учесть перед редактированием парольной политики, приведены в подразделе 2.11.

Для редактирования парольной политики администратору необходимо выполнить следующие действия в меню «Админ. системы»/«Парольные политики»:

1) Выбрать парольную политику.
2) Нажать кнопку «Редактировать». Откроется окно «Редактирование парольной политики».
3) На вкладке «Параметры» выполнить редактирование поля «Название парольной политики». Поле «Назначение» недоступно.

4) На вкладке «Правила политики» выполнить редактирование, удаление или добавление правил:
   - нажать кнопку «Редактировать» для редактирования правила. Внести изменения и нажать кнопку «Изменить»;
   - нажать кнопку «Удалить» для удаления правила;
   - нажать кнопку «Добавить правило» для добавления нового правила. Выполнить действия, приведенные в подпункте 2.11.1.1.
5) На вкладке «Пользовательские правила» выполнить добавление пользовательских правил:
   - нажать кнопку «Редактировать» для редактирования правила. Внести изменения и нажать кнопку «Изменить»;
   - нажать кнопку «Удалить» для удаления правила;
   - нажать кнопку «Добавить правило», откроется окно «Пользовательское правило». Выполнить действия, приведенные в подпункте 2.11.1.2.
6) На вкладке «Список УЗ ресурсов» выполнить редактирование УЗР. Вкладка недоступна, если был установлен флаг «Применить ко всем УЗ ресурсов».
7) Нажать кнопку «Сохранить».

2.11.3. Применение парольной политики

Способы применения парольных политик:

1) Вручную.
2) По умолчанию (автоматически).

Комплекс применяет парольные политики в зависимости от их приоритета.

Способы применения парольных политик и приоритет приведены в таблице 2.39.

Таблица 2.39 – Способы применения парольных политик и приоритет

Назначение политики	**Объекты, к которым применяется политика**	**Способ применения**	**Приоритет**
Пользователи	Все пользователи	По умолчанию	–

Назначение политики	Объекты, к которым применяется политика	Способ применения	Приоритет
Учетная запись ресурса	Конкретные УЗ в выбранных УЗР	Вручную	Высокий
Учетная запись ресурса	Все УЗ в выбранных УЗР	По умолчанию	Средний
Учетная запись ресурса	Все УЗР Комплекса	По умолчанию	Низкий

В Комплексе может применяться только одна парольная политика на все УЗР Комплекса. Она автоматически будет применяться и к новым УЗР.

Парольная политика так же будет применяться для тех паролей, которые приходят через синхронизацию из ЦС.

Парольную политику, которая создана для конкретных УЗ в выбранных УЗР необходимо вручную связать с УЗ пользователей. Действия подробно описаны в пункте 2.3.6.

Для применения более строгой политики для конкретной УЗР следует:

1) Создать строгую политику на эту же УЗР с выключенными флагами.
2) Сменить политику по умолчанию для УЗ на более строгую. Смена парольной политики описана в пункте 2.3.6. В противном случае будет действовать парольная политика по умолчанию.

2.11.4. Удаление парольной политики

Для удаления парольной политики администратору необходимо выполнить в меню «Админ. системы»/«Парольные политики» следующие действия:

1) Выбрать из списка парольную политику.
2) Нажать кнопку «Удалить». Появится окно «Удаление парольной политики».
3) В окне «Удаление парольной политики» отображается информация:
   – название парольной политики и область ее применения;
   – информационное сообщение «При удалении парольной политики будут удалены все ее связки с аккаунтами».
4) Нажать кнопку «Удалить». Для отмены действий следует нажать кнопку «Отменить».

2.12. Настройка уведомлений

Комплекс позволяет настраивать уведомления на электронную почту пользователей следующими средствами:

1) Настройка уведомлений на события (создание, изменение, удаление) управляемых объектов в файле managed.json. Отправка уведомлений осуществляется скриптом (JavaScript).
2) Настройка уведомлений на события (создание, изменение, удаление) УЗ в объекте «Маппинг». Отправка уведомлений осуществляется скриптом (JavaScript).
3) Настройка уведомлений в рабочих потоках с помощью модуля «Activiti». Отправка уведомлений осуществляется задачей рабочего потока «Mail Task».
4) Настройка уведомлений при согласовании заявки из письма.

Поддерживаемые атрибуты письма приведены в таблице 2.40.

Таблица 2.40 – Поддерживаемые атрибуты письма

Атрибут	**Описание**	**Признак обязатель-ности**	**Значение по умолчанию**	**Пример**	**Комментарии**
subject	Тема письма	Нет	<no subject>	Ankey IDM: создание УЗ	–
from	От кого	Нет	Значение атрибута from из файла настроек external.email.json	ankeynoreply@gaz-is.ru	Если атрибут from не заполнен и не указано значение по умолчанию в файле настроек external.email.json, произойдёт ошибка
to	Кому	Да	–	aib@gaz-is.ru,ankeyadmin@gaz-is.ru	В случае нескольких получателей их адреса указываются через запятую
cc	Копия	Нет	–	user@gaz-is.ru,manager@gaz-is.ru	В случае нескольких получателей их адреса указываются через запятую
bcc	Скрытая копия	Нет	–	user@gaz-is.ru,manager@gaz-is.ru	В случае нескольких

Атрибут	Описание	Признак обязатель-ности	Значение по умолчанию	Пример	Комментарии
					получателей их адреса указываются через запятую
type	Тип письма	Нет	text/html	text/plain	Возможные значения: text/plain, text/html, text/xml. В случае указания не поддерживаемого типа произойдёт ошибка
body	Тело письма	Нет	<empty message>	В системе MS Active Directory создана УЗ aib@gaz-is.ru	–

2.12.1. Настройка почтового сервера

Настройка почтового сервера, через который будут осуществляться уведомления выполняется в следующих файлах:

1) Файл ankey/conf/workflow.json (в случае уведомлений из рабочего потока).
2) Файл ankey/conf/external.email.json (в случае уведомлений для управляемых объектов и объекта «Маппинг»).

Настройка почтового сервера в файле workflow.json выполняется в секции «mail», как показано в следующем примере:

```
{
   "enabled": true,
   "workflowDirectory": "&{launcher.project.location}/workflow",
   "mail": {
      "host": "localhost",
      "port": 25,
      "username": "ankeynoreply@domain.local",
      "password": "******",
      "starttls": false,
      "defaultFrom": "ankeynoreply@domain.local"
   }
}
```

Перечень параметров настройки в секции «mail» в файле workflow.json:

– «host» – сетевой адрес сервера почтовой службы;
– «port» – порт для отправки электронной почты почтовой службой;

- «username» – идентификатор пользователя для аутентификации на сервере почтовой службы. В случае анонимного доступа для отправки почты параметр не указывается;
- «password» – пароль пользователя для аутентификации на сервере почтовой службы. При считывании файла системой, указанное в поле в открытом виде значение пароля преобразуется в закрытое значение. В случае анонимного доступа для отправки почты параметр не указывается;
- «starttls» – параметр подключения к почтовой службе по протоколу SSL, принимает значения «true» или «false»;
- «defaultFrom» – параметр с указанием адреса отправителя по умолчанию, используется если явно не задан отправитель в рабочем потоке в задачах отправки писем «Mail Task».

Настройка почтового сервера в файле external.email.json выполняется, как показано в следующем примере:

```
{
   "host": "localhost",
   "port": "25",
   "debug": false,
   "from": "ankeynoreply@domain.local",
   "auth": {
      "enable": true,
      "username": "ankeynoreply@domain.local",
      "password": "******"
   },
   "starttls": {
      "enable": false
   }
}
```

Перечень параметров в файле external.email.json:

- «host» – сетевой адрес сервера почтовой службы;
- «port» – порт для отправки электронной почты почтовой службой;
- «debug» – параметр для вывода отладочной информации в журнал работы, принимает значения «true» или «false»;
- «auth» – секция содержит параметры для аутентификации на сервере почтовой службы;

- «auth»/«enable» – параметр выполнения аутентификации на сервере почтовой службы, принимает значения «true», если на сервере почтовой службы требуется аутентификация, или «false», если не требуется;
- «auth»/«username» – идентификатор пользователя для аутентификации на сервере почтовой службы;
- «auth»/«password» – пароль пользователя для аутентификации на сервере почтовой службы. При считывании файла системой, указанное в поле в открытом виде значение пароля преобразуется в закрытое значение;
- «starttls» – секция содержит параметры подключения к почтовой службе по протоколу SSL;
- «starttls»/«enable» – параметр подключения к почтовой службе по протоколу SSL, принимает значения «true» или «false»;
- «from» – параметр с указанием адреса отправителя по умолчанию, используется если явно не задан отправитель.

2.12.2. Настройка скрипта отправки уведомлений

Отправка уведомления может выполняться предварительно разработанным скриптом (например, JavaScript). По умолчанию скрипты хранятся в каталоге ankey/script.

Пример содержимого скрипта отправки уведомления:

```
var email = {
    from : "ankeynoreply@domain.local",
    to : "user@domain.local",
    cc : "admin@domain.local",
    subject : "Account created",
    type : "text/html"
  };

if (openidm.read("config/external.email")) {
```

```
    email.body = "Found created account in target system:<br>"
    + "<table border='1'><tbody>"
    + "<tr><td>Account Name</td><td>" + source.username + "</td></tr>"
    + "<tr><td>Last Name</td><td>" + source.lastname + "</td></tr>"
    + "<tr><td>First Name</td><td>" + source.firstname + "</td></tr>"
    + "<tr><td>Middle Name</td><td>" + source.middlename + "</td></tr>"
    + "</tbody></table><br><br>"
    + "Account created in ankey.";

    openidm.action("external/email", "sendEmail", email);

} else {
    console.log("Email service not configured; report not generated. ");
}
```

В примере скрипта выше вызываются следующие функции:

1) openidm.read ("config/external.email") – функция считывает настройки подключения к почтовому серверу из файла external.email.json.
2) openidm.action ("external/email", "sendEmail", email) – функция отправки почты. Процедура получает три аргумента на вход: идентификатор объекта (пример, «external/email»), действие («sendEmail») и объект, в котором хранятся необходимые для выполнения действия значения (пример, email).

Объект, в котором хранятся необходимые для выполнения отправки уведомления данные, включает следующие параметры:

1) «from» – почтовый адрес отправителя.
2) «to» – почтовые адреса получателей, разделенные запятой.
3) «cc» – почтовые адреса получателей, разделенные запятой, в копии письма.

4) «bcc» – почтовые адреса получателей, разделенные запятой, в скрытой копии письма.
5) «subject» – тема письма.
6) «body» – содержимое письма.
7) «type» – формат содержимого письма (MIME type). Может принимать значения «text/plain», «text/html», or «text/xml».

2.12.3. Отправка уведомления с помощью Rest-запроса

Для отправки уведомления используется следующий Rest-запрос:

```
ANKEY@NODE1:~$ curl
--header "Content-Type: application/json" ^
--header "X-ankey-Username: ankey" ^
--header "X-ankey-Password: ankey" ^
--request POST ^
--data '{"from":"ankeynoreply@domain.local","to":"recipient@domain.local", "subject":"Test","body":"Test"}'
```

«http://localhost:8080/ankey/external/email?_action=send»

2.12.4. Настройка согласования заявки из уведомления

Комплекс поддерживает возможность направления заявки по электронной почте и согласованию ее в ответном письме.

Настройка решения по согласованию заявки из письма выполняется в следующих объектах:

1) Файл настроек параметров подключения к почтовому серверу для отправки сообщений external.email.json . Описание приведено в пункте 2.12.1.
2) Файл настроек параметров подключения к почтовому серверу для получения входящих сообщений email.request .processing.json.
3) Периодическое задание с типом «email.request.processing.task» для обработки действий с заявками через электронную почту. Описание приведено в пункте 2.4.5.

Параметры конфигурационного файла email.request.processing.json приведены в таблице 2.41.

Таблица 2.41 – Параметры конфигурационного файла email.request.processing.json

Атрибут	Тип	Описание
enabled	Флаг	Включение возможности согласования через письмо. Значение по умолчанию: false
mail:	–	Родительский атрибут, который содержит набор настроек подключения к почтовому серверу
storeType	Строка	Протокол чтения электронной почты. Поддерживаемые значения: – imap и pop3 - без SSL соединения; – imaps и pop3s - для SSL соединения. Значение по умолчанию: «imap»
host	Строка	Адрес почтового сервера
port	Число	Порт для соединения с почтовым сервером. Значение по умолчанию: 143
user	Строка	Пользователь для соединения с почтовым сервером
password	Строка	Пароль пользователя. После старта Комплекса значение пароля шифруется
email	Строка	Почтовый ящик для обработки письма
folder	Строка	Директория почтового ящика из которой читаются письма. Значение по умолчанию: «INBOX»
properties	Строка	Свойство, которое позволяет установить SSL соединение. Не обязательно, если не установлено SSL.Для подключения по ipmap с включенным безопасным TLS-подключением надо указать следующие параметры: 1) Для протокола «imaps» указать: port: 993 "properties": {"mail.imaps.ssl.trust": "ваш хост","mail.imap.starttls.enable": "true"} 2) Для протокола «pop3s» указать: «port»: 995 "properties": {"mail.pop3s.ssl.trust": "ваш хост"}
templates:	–	Родительский атрибут, который содержит набор настроек шаблонов писем для настроенных рабочих потоков (бизнес процессов (БП))
<имя БП>	Объект	Описание шаблона отправляемого письма. Должны быть указаны 2 вложенных свойства: – subject – строка, шаблон темы письма; – body – строка, шаблон тела письма

Пример настройки без SSL соединения:

```
{
"mail": {
   "storeType": "pop3",
   "host": "exchange.gaz-is.ru",
   "port": 110,
   "user": "GIS\\request",
   "password": "********",
   "email": "request@gaz-is.ru",
   "folder": "INBOX",
```

```
    "properties": {}
    }
  }
```

Пример настройки для SSL соединения по протоколу imaps:

```
  "mail": {
    "storeType": "imaps",
   "host": "10.0.101.999",
   "port": 993,
   "user": "GIS\\request",
   "password": "********",
   "email": "request@gaz-is.ru",
   "folder": "INBOX",
   "properties": {"mail.imaps.ssl.trust": "10.0.101.999"
     }
  }
```

Пример настройки для SSL соединения по протоколу pop3s:

```
  "mail": {
  "storeType": "pop3s",
  "host": "10.0.101.999",
  "port": 995,
  "user": "GIS\\request",
  "password": "********",
  "email": "request@gaz-is.ru",
  "folder": "INBOX",
  "properties": {
  "mail.pop3s.ssl.trust": "10.0.101.999"
     }
  }
```

Пример заполнения шаблона отправляемого письма:

```
"templates": {
   "managerApproval": {
       "subject": "Ankey IDM: Заявка №${requestId} на доступ к информационному ресурсу.",
       "body": "Уважаемый пользователь, в системе Ankey IDM создана заявка №${requestId} на доступ к
информационной системе:<br><br><table border='1' style='border-
collapse:collapse'><tbody><tr><td><b>Запрашиваемые
роли:</b></td><td>${appRolesWithIsNames}</td></tr><tr><td><b>Фамилия Имя
Отчество:</b></td><td>${beneficiary.fullName}</td></tr><tr><td><b>Подразделение:</b></td><td>${benefic
iary.organizationName}</td></tr><tr><td><b>Обоснование:</b></td><td>${requestForm.justification}</td></tr
><tr><td><b>Договор
КТ:</b></td><td>${requestForm.contract}</td></tr><tr><td><b>Должность:</b></td><td>${requestForm.posit
ion}</td></tr></tbody></table><br>Предыдущие согласующие:</b><br>${previousApprovers}<br><br>Этап
'<b>${taskName}</b>'.<br>Вам необходимо рассмотреть ее.<br>Доступные действия: <a
href='${approveLink}'>Согласовать</a> <a href='${rejectLink}'>Отклонить</a><br>Данные по заявке
доступны по ссылке: <a href='${baseUrl}'>Ankey IDM</a>"
   }
}
```

Параметры, доступные для использования в шаблоне письма приведены в таблице 2.42.

Таблица 2.42 – Параметры, доступные для использования в шаблонах

Параметр	Описание	Пример
beneficiary	Данные по бенефициару. В шаблоне можно использовать только строковые атрибуты	1) ${beneficiary.fullName}, результат «Глазырин Сергей Игоревич». 2) ${beneficiary.organizationName}, результат «Группа разработки java-решений №1». 3) ${beneficiary.userName}, результат «glazyrin-s»
appRolesNames	Имена согласуемых ролей, по одному имени роли на строку	${appRolesNames}, результат: 1) Доступ в Confluence. 2) Доступ в Jira
appRolesWithIsNames	Имена согласуемых ролей и информационных систем. В случае групповой заявки все роли и ИС перечисляются в виде «Название роли [Инф.система]» по одному имени роли на строку	${appRolesWithIsNames}, результат: 1) «Бизнес аналитик [Группа разработки]». 2) «Бухгалтер [Сетевое рабочее место]». 3) «Главный бухгалтер [Сетевое рабочее место]»
requestForm	Данные формы заявки	$requestForm.justification}, результат «Выполнение работ по Ankey IDM» ${requestForm.contract}, результат «№ 430»
requestId	Номер заявки	${requestId}, результат «314»
baseUrl	Ссылка на стартовую страницу Ankey IDM. Задаётся через java-параметр baseUrl. (например, export JAVA_OPTS=-DbaseUrl=https://www.gaz-is.ru/ankey $JAVA_OPTS	${baseUrl}, результат «http://ankey»
approveLink	Письмо для согласования заявки	–
rejectLink	Письмо для отклонения заявки	–
taskName	Имя задания согласования	${taskName}, результат «Прямой руководитель»
previousApprovers	Предыдущие согласующие с информацией о дате согласования, ФИО согласующего и названии этапа согласования, по одному этапу на строчку	${previousApprovers}, результат: – «04.10.2019, Глазырин Сергей Игоревич [Прямой руководитель]»; – «05.10.2019, Белов Эдуард Владимирович [Начальник отдела]»

2.12.5. Настройка уведомления о новом инциденте

Для настройки уведомления о новом инциденты необходимо выполнить следующие действия:

1) Настройки параметра почтового сервера в файле ankey/conf/workflow.json согласно пункту 2.12.1.
2) Убедиться, что хотя бы у одного оператора SOD, для которого будет назначено рассмотрение инцидента, указан почтовый адрес.
3) Убедиться, что в системе установлена переменная JAVA_OPTS со значением -DbaseUrl=<адрес ankey>, например -DbaseUrl=http://www.gaz-is.ru/ankey. В случае отсутствия этого параметра будет использован адрес по умолчанию: http://localhost:8080/

Шаблон уведомления о новом инциденте приведен в таблице 2.43.

Таблица 2.43 – Шаблон уведомления о новом инциденте

Атрибуты письма	Текст	Пример
Тема	Новый инцидент №<номер инцидента>	Новый инцидент №17
Сообщение	Для Вашей информационной системы <имя информационной системы> имеется новый инцидент №<номер инцидента>: Правило (роль): <имя роли>; Нарушитель: <ФИО нарушителя>. Ознакомиться с деталями инцидента можно по ссылке <адрес ankey>	Для Вашей информационной системы Сетевое рабочее место имеется новый инцидент №17: Правило (роль): Администратор домена; Нарушитель: Барбашев Олег Андреевич. Ознакомиться с деталями инцидента можно по ссылке https://www.gaz-is.ru/ankey

2.12.6. Настройка уведомлений в рабочих потоках с помощью модуля «Activiti»

Для настройки уведомлений в рабочих потоках используется типовая задача рабочего потока «Mail Task» модуля «Activiti». Основные рекомендации по настройке «Mail Task»:

– задача «Mail Task» должна выполняться параллельно другим задачам рабочего потока (например, с помощью компонента «Parallel Gateway»), для исключения задержек в обработке заявки или ошибочного завершения рабочего потока;

– возникающие исключения (например, не определен адрес получателя) при работе задачи «Mail Task» должны игнорироваться, для этого в файл конфигурации рабочего потока добавлен параметр «ignoreException»:

```
<serviceTask id="mailtask1" name="Mail Task" activiti:type="mail">
  <extensionElements>
              ........
    <activiti:field name="ignoreException">
      <activiti:string>
        <![CDATA[true]]>
      </activiti:string>
    </activiti:field>
  </extensionElements>
</serviceTask>
```

2.12.7. Перенос на новую строку во всплывающих сообщениях

Для настройки переноса на новую строку во всплывающих сообщениях Комплекса необходимо выполнить следующие действия:

1. Добавить «» в файл ankey/ui/default/ng/public/i18n/translation_ru.properties
2. Перезагрузить Комплекс.

Пример настройки переноса на новую строку во всплывающем сообщении для сообщения при создании подразделения (см. «Руководство пользователя», п. 2.6.1).

```
templates.organization.create.success.message=\u041F\u043E\u0434\u0440\u0430\u0437\u0434\u0435\u043B\u0435\u043D\u0438\u0435\u000A\u0443\u0441\u043F\u0435\u0448\u043D\u043E\u000A\u0441\u043E\u0437\u0434\u0430\u043D\u043E
```

2.13. Настройка производительности системы

Факторы влияющие на производительность системы включают:

1) Количество конкурентных (одновременно работающих) пользователей.
2) Количество конкурентных заданий на синхронизацию.
3) Количество конкурентных изменений объектов (атрибутов, заявок, ролей и т.д.), обрабатываемых и регистрируемых системой (аудит).
4) Объем индексируемых данных об объектах в службе поиска.

Настройка производительности выполняется на следующих компонентах:

1) Сервер приложений Ankey IDM.
2) СУБД.

3) Коннектор сервер.
4) Служба поиска.

2.13.1. Настройка производительности сервера приложений Ankey IDM

Настройка производительности сервера приложений включает следующие настройки:

1) Настройка выделяемой памяти Java процессу сервера приложений.
2) Настройка пула соединений к СУБД.
3) Отказ от создания точек восстановления перед каждым запросом в БД.
4) Настройка асинхронности выполнения операций распространения данных.
5) Настройка количества потоков выполнения периодических заданий.

2.13.1.1. Настройка выделяемой памяти Java процессу сервера приложений

По умолчанию размер выделяемой памяти процессу Java сервера приложений Комплекса задается в скрипте ankey/startup.sh в переменной окружения: ANKEYIDM_OPTS="-Xmx1024m -Xms1024m".

При необходимости данные параметры могут быть изменены в соответствии с документацией Java.

Дополнительно для оптимизации работы с памятью рекомендуется выбирать подходящий под текущую нагрузку Комплекса сборщик мусора Java.

2.13.1.2. Настройка пула соединений к СУБД

Настройка пула соединений к СУБД выполняется в конфигурационном файле conf/repo.jdbc.json в секции «pool». Описание параметров секции «pool» приведено в таблице 2.44.

Таблица 2.44 – Описание параметров секции «pool»

Параметр	Значение по умолчанию	Описание параметра
initialSize	0	Первоначальный размер пула при старте сервера приложений Комплекса
maxTotal	20	Максимальное число подключений к БД
maxIdle	20	Максимальное число простаивающих соединений к БД

Параметр	Значение по умолчанию	Описание параметра
maxWaitMillis	5000	Максимальное время ожидания подключения к БД в миллисекундах. Значение по умолчанию «-1» означает, что нет ограничения по времени

Дополнительные параметры, настройка которых может потребоваться, если есть проблема с периодическим отключением соединений к БД. Описание дополнительных параметров приведено в таблице 2.45.

Таблица 2.45 – Описание дополнительных параметров

Параметр	Значение по умолчанию	Описание параметра
testOnBorrow	true	Проверка соединения на работоспособность при каждом запросе из пула. При значении «true» должно быть указано значение validationQuery
testOnCreate	false	Проверка на работоспособность перед созданием соединения
testOnReturn	false	Проверка перед возвращением соединения в пул. Если после очередного цикла проверки соединения в пуле: – maxIdle, то при попытке возврата соединения в пул оно будет закрыто; – < minIdle, то фоновый процесс создаст дополнительные соединения. Объекты, которые не могут быть проверены, будут удалены из пула
testWhileIdle	false	Проверка неиспользуемых соединений. При значении «true» должно быть указано значение validationQuery
timeBetweenEvictionRunsMillis	-1	Время ожидания в миллисекундах между проверками работоспособности и очистки неактивных соединений
numTestsPerEvictionRun	3	Количество неактивных соединений, проверяемых за раз
minEvictableIdleTimeMillis	1000 * 60 * 30	Минимальное время между проверками доступности соединения
removeAbandonedOnBorrow	false	Удаление некорректных соединений при выделении из пула. Соединение считается разорванным и может быть удалено, если оно не использовалось дольше, чем значение параметра removeAbandonedTimeout
removeAbandonedOnMaintenance	false	Удаление из пула некорректных соединений после проверки доступности соединения

Параметр	Значение по умолчанию	Описание параметра
removeAbandonedTimeout	300	Время ожидания в секундах, через которое любое простаивающее соединение будет считаться неактивным
validationQuery	–	SQL запрос проверки соединения. Запрос должен использовать оператор SELECT, который возвращает хотя бы одну строку, например, «SELECT 1»
connectionInitSqls	NULL	SQL запрос инициализации соединения

Пример настройки параметров соединений к СУБД в конфигурационном файле conf/repo.jdbc.json в секции «pool»:

```
"pool" : {
        "initialSize": 0,
        "maxTotal": 8,
        "maxIdle": 8,
        "minIdle": 0,
        "maxWaitMillis": -1,
        "testOnBorrow": true,
        "testOnCreate": true,
        "testOnReturn": true,
        "testWhileIdle": true,
        "timeBetweenEvictionRunsMillis": 400,
        "numTestsPerEvictionRun": 4,
        "minEvictableIdleTimeMillis": -1,
        "removeAbandonedOnBorrow": true,
        "removeAbandonedOnMaintenance": true,
        "validationQuery": "SELECT 1",
        "connectionInitSqls": ["SELECT 1", "SELECT 2"]
    }
```

2.13.1.3. Отказ от создания точек восстановления перед каждым запросом в БД

После запуска комплекса в постоянную эксплуатацию (фиксирование дополнительных атрибутов объектов системы) можно отказаться от создания точек восстановления перед каждым запросом в БД во избежание ошибки «cached plan must not change result type», возникающей после изменения структуры таблиц (добавление/изменение/удаление колонок) при использовании пула соединений в БД PostgreSQL. Это позволит избежать потенциальных задержек при взаимодействии с БД.

Для отключения функции создания точек восстановления необходимо в конфигурационном файле repo.jdbc.json убрать параметр autosave=conservative из переменной connection/dataSource/url.

Например, вместо:

```
"url"  : "jdbc:postgresql://localhost:5432/ankey?autosave=conservative"
```

будет

```
"url"  : "jdbc:postgresql://localhost:5432/ankey"
```

При необходимости поменять дополнительные атрибуты после отключения параметра, необходимо выполнить следующие действия:

1) Вернуть первоначальное значение параметра.
2) Перезагрузить сервер приложений Комплекса.
3) Внести в extend.json изменения и дождаться их применения.
4) Отключить функции создания точек восстановления.
5) Перезагрузить сервер приложений Комплекса.

2.13.1.4. Настройка асинхронности выполнения операций распространения данных

Настройка асинхронности выполнения операций распространения данных может выполняться в случае долгого выполнения provision-операций (для объекта «Маппинг» параметр «mapType» установлен в «provision»). При включении режима асинхронного выполнения на стороне Комплекса объекты будут создаваться или обновляться мгновенно, вне зависимости выполнилась ли операция или еще нет фактически. Настройка асинхронного выполнения операций делается в конфигурационном файле conf/sync.json. Пример настроек в файле conf/sync.json:

```
{
    "async": true,
    "threadPoolCount": 4
}
```

Настройки в файле conf/sync.json включают следующие параметры:

– «async» – флаг активации режима асинхронного исполнения. Значение по умолчанию «false»;

– «threadPoolCount» – количество одновременно выполняемых операций в асинхронном режиме. Обязательный параметр, если «async» установлен в значение «true».

2.13.1.5. Настройка кол-ва потоков выполнения периодических заданий

При большом количестве периодических заданий рекомендуется увеличить количество потоков выполнения «threadCount» в файле conf/scheduler.json, описание приведено в пункте 2.4.6.

2.13.2. Настройка производительности СУБД

Комплекс Ankey IDM можно отнести к транзакционным системам (Online Transaction Processing - OLTP). Комплекс может выполнять большое число чтений из БД (например, работа пользователей в интерфейсе), так и большое число записей в БД (например, синхронизация данных, пересчет полномочий (задача evaluate), аудит).

Общие рекомендации по повышению производительности СУБД:

1) Для построения отчетов использовать отдельную реплику СУБД (например, Standby реплика в режиме только чтения), чтобы уменьшить нагрузку на оперативные данные Комплекса.
2) Для записи и чтения событий аудита использовать отдельную инстанцию СУБД. Описание настройки приведено в пункте 2.10.4.
3) Выполнять мониторинг производительности СУБД и повышение ее общей производительности в соответствии с документацией разработчика СУБД.
4) Настраивать количество допустимых подключений на стороне СУБД необходимо с учетом настройки пула соединений на стороне сервера приложений подпункта 2.13.1.3.

Для колонок таблиц, используемых в правилах связывания УЗР и правилах корреляции в маппингах синхронизации, рекомендуется создавать индексы в БД. Для создания индекса создается файл настройки «liquibase» и помещается в каталог ankey\db\postgresql\liquibase. Пример файла настройки индекса ankey\db\postgresql\liquibase\snn.xml для дополнительного поля «Табельный номер SNN» объекта «Пользователь» (managed/user):

```
<changeSet id="123" author="gis">
  <createIndex tableName="usr"
               indexName="idx_snn"
               schemaName="ankey"
               tablespace="pg_default"
               unique="true">
    <column name="snn" type="character varying(64)"/>
  </createIndex>
</changeSet>
```

2.13.3. Настройка производительности службы поиска

Настройка производительности службы поиска Opensearch выполняется в соответствии с документацией разработчика.

Комплекс не содержит дополнительных настроек производительности в части службы поиска.

2.14. Настройка интеграции с внешними системами

Интеграция с внешними системами осуществляется с использованием программного интерфейса REST API Комплекса. Описание поддерживаемых функций REST API Комплекса доступно в веб-консоли «Swagger UI», которая доступна по умолчанию по адресу: http://localhost:8080/api.

В веб-консоли «Swagger UI» доступны следующие действия с объектами Комплекса:

1) Просмотр объектов и их деталей. Выполняется запросом «GET» для соответствующих объектов.
2) Создание новых объектов. Выполняется запросом «POST» или «PUT» для соответствующих объектов.
3) Изменение объектов. Выполняется запросом «PATCH» или «PUT» для соответствующих объектов.
4) Удаление объектов. Выполняется запросом «DELETE» для соответствующих объектов.

Все действия выполняются из под служебной УЗ пользователя «ankey» с паролем по умолчанию «ankey». Действия могут быть выполнены из под другого пользователя, для этого необходимо указать имя пользователя и его пароль в

параметрах «X-Ankey-Username» и «X-Ankey-Password» соответственно. Описание остальных параметров приведено в веб-консоли «Swagger UI».

Дополнительно администратор Комплекса может настраивать интеграцию со следующими внешними системами:

1) Внешние системы формирования отчетов. Интеграция производится за счет публикации внешней ссылки в интерфейсе Комплекса на веб-консоль системы формирования отчетов.
2) Внешние системы, которым необходимо проверять роли, назначаемые пользователю, на предмет нарушения правил разграничения доступа, настроенных в Комплексе.

2.14.1. Настройка внешней ссылки

Комплекс позволяет добавлять внешнюю ссылку и отображать ее в качестве пункта меню. Пункт меню с внешними ссылками по умолчанию называется «Документы и отчеты» и доступен Администратору всегда, даже когда внешние ссылки не заданы. Пользователь может видеть и получать доступ к этому пункту меню только тогда, когда есть одна или более внешних ссылок.

Администратору доступна кнопка «Редактировать», недоступная пользователям. Кнопка «Редактирование» открывает окно «Управление внешними ссылками», в котором Администратор может добавлять новые ссылки, изменять название внешней ссылки, редактировать и удалять заданные ранее ссылки.

Администратор может изменять название страницы с внешними ссылками в окне «Управление внешними ссылками», но название страницы в списке «Ссылки» изменится только при следующем входе в систему, о чем сообщает информационная надпись в окне.

Добавление новой ссылки начинается с нажатия кнопки «Добавить ссылку» в окне «Управление внешними ссылками». Открывается окно «Создание внешней ссылки». Поля «Название» и «Адрес ссылки» - это обязательные поля. Кнопка «Создать» будет заблокирована до заполнения обязательных полей.

Название внешней ссылки уникально. Адрес ссылки проходит валидацию: при попытке ввести некорректные данные появится сообщение «Адрес ссылки должен начинаться с http, https, ftp».

Не обязательные поля - это поля «Ссылка будет доступна следующим ролям» и «Ссылка будет доступна следующим ролям по маске». В поле «Область видимости ролей» Администратор может выбрать существующую в системе роль. В поле «Ссылка будет доступна следующим ролям по маске» действует всплывающая подсказка о том, какие роли пользователей могут получить доступ по введенной маске. Маска - это часть названия роли, она не зависит от регистра. Например, при введении в это поле значения «бухгалтер» отображаются %% внутри которых можно задать вручную значение маски.

Нажатие кнопки «Сохранить» после заполнения обязательных и, при необходимости, не обязательных полей сохраняет вновь созданную внешнюю ссылку, что подтверждается появлением сообщения «Внешние ссылки обновлены». После этого в списке «Ссылка» появится новая ссылка, доступная пользователям.

Существующую ссылку можно редактировать в окне «Управление внешними ссылками», которое открывается после нажатия кнопки «Редактировать». В этом окне для каждой ссылки есть своя кнопка «Редактировать», нажатие которой открывает для выбранной ссылки окно «Редактирование внешней ссылки», в котором можно добавить несколько ролей (добавленная роль не отображается в списке) или удалить как одну, так и все роли. В этом же окне можно редактировать обязательные поля «Название» и «Адрес ссылки», добавлять, изменять или удалять маски.

Для того, чтобы удалить внешнюю ссылку необходимо в окне «Управление внешними ссылками» нажать на кнопку «Корзина», расположенную рядом с удаляемой ссылкой, а затем нажать кнопку «Сохранить». Удалять ссылки может только Администратор.

2.14.2. Проверка нарушений правил разграничения доступа

Для проверки нарушений правил разграничения доступа из внешней системы необходимо отправить запрос REST API со следующими параметрами:

– метод «POST»;

– адрес «/endpoint/preliminary/incident»;
– тело запроса:

```
{
   "userName": "<Идентификатор пользователя>",
   "appRoleNames": ["<Имя роли 1>", "<Имя роли 1>"]
}
```

В теле запроса в качестве идентификатора пользователя используется значение атрибута «userName» объекта «Пользователь», в качестве списка имен ролей «appRoleNames» значение атрибута «appRoleName» объекта «Роль». Если передается несколько ролей, то они разделяются запятыми.

В случае выявления нарушения Комплекс возвращает следующий ответ:

```
{
   "hasIncidents" : true,
   "manual": ["<Имя правила 1>","<Имя правила 2>"],
   "role": ["<Имя роли 1>","<Имя роли 2>"]
}
```

Содержимое ответа включает:

– «hasIncidents» – флаг наличия («true») или отсутствия («false») нарушения.
– «manual» – список правил разграничения доступа, на которых выявлено нарушение.
– «role» – список ролей, на которых выявлено нарушение, в случае если пользователь находится в другом подразделении чем роль.

2.15. Настройка службы сервера коннекторов

Настройка службы сервера коннекторов ICF Connector Server выполняется следующими средствами:

1) Конфигурационный файл <Путь к каталогу с ICF Connector Server>/conf/ConnectorServer.properties.
2) Скрипт <Путь к каталогу с ICF Connector Server>/bin/ConnectorServer.sh.

Файл ConnectorServer.properties включают следующие параметры настройки:

1) Порт службы сервера коннекторов, на котором будут приниматься подключения «connectorserver.port». По умолчанию имеет значение: «8759».
2) Наименование каталога на сервере коннектора, где располагаются внешние библиотеки необходимые для работы коннекторов «connectorserver.libDir». По умолчанию имеет значение: «lib»
3) Параметр активирующий режим подключения к службе сервера коннекторов по протоколу SSL «connectorserver.usessl». По умолчанию имеет значение: «false». Для включения режима необходимо установить параметр в значение «true».
4) Наименование каталога на сервере коннектора, где располагаются библиотеки коннекторов «connectorserver.bundleDir». По умолчанию имеет значение: «bundles».
5) Наименование используемой библиотеки журналирования работы сервера коннекторов «connectorserver.loggerClass». По умолчанию имеет значение: org.forgerock.openicf.common.logging.slf4j.SLF4JLog.
6) Ключ аутентификации при подключении к серверу коннекторов «connectorserver.key». Хранится в виде хеш значения, задаваемого параметром /setKey скрипта ConnectorServer.sh.

Скрипт ConnectorServer.sh имеет следующие параметры запуска:

1) Параметр «/run <параметры Java>». Запускают службу сервера коннекторов. Пример выполнения: «./ConnectorServer.sh /run» или «./ConnectorServer.sh /run «-J-Djavax.net.ssl.keyStore=mykeystore.jks» «-J-Djavax.net.ssl.keyStorePassword=changeit»», в случае подключения к службе сервера коннекторов по протоколу SSL.
2) Параметр «/setKey <значение ключа>». Задает значение ключа аутентификации при подключении к серверу коннекторов. Пример выполнения: «./ConnectorServer.sh /setKey changeit».
3) Параметр «/setDefaults». Сбрасывает настройки файла ConnectorServer.properties на значения по умолчанию. Пример выполнения: «./ConnectorServer.sh /setDefaults».

2.16. Настройка правил

Комплекс содержит следующие типы правил:

1) Правило автоназначения роли. Настраивается при создании/редактировании роли.
2) Правило связывания. Настраивается при создании/редактировании УЗР, для связывания новых УЗ ЦС с их владельцами (бенефициарами).
3) Правило корреляции. Настраивается в объекте «Маппинг» с типом «recon» для синхронизации объектов из доверенного источника. В таком режиме отсутствует форма ресурса и правило связывание (п.2.) не применимо.
4) Правило разграничения доступа (далее – правила SoD). Настраивается в подменю «Правила» меню «Админ. системы», для выявления нарушений прав доступа в дополнительных сценариях использования. Дополняют автоматически настраиваемые правила разграничения доступа при установке связи роли и подразделения.

2.16.1. Настройка правила автоназначения

Правило автоназначения роли позволяет настроить автоматическое назначение роли пользователю или группе пользователей на основе заданного правила соответствия.

В случае если в системе присутствуют пользователи, которые удовлетворяют правилу, выполняется автоматическое назначение роли. В случае если атрибуты пользователей, которым была автоматически назначена роль, перестали соответствовать правилу, производится автоматический отзыв роли.

Назначение и отзыв ролей по правилу автоназначения выполняется стандартным периодическим заданием Комплекса «approlesbyrule».

Для создания правила автоназначения роли администратору необходимо выполнить следующие действия:

1) Перейти в подменю «Список ролей» раздела «Роли».
2) Нажать кнопку «Создать». Откроется окно «Создание роли».
3) Заполнить необходимые поля.

4) Указать в поле «Правила автоназначения ролей» правило. Ввод правила доступен в двух режимах: в режиме конструктора или в режиме кода.
5) Нажать кнопку «Создать» для завершения действия. В случае ввода недопустимых данных отображается подсказка о некорректно введенных данных. Для отмены действий следует нажать кнопку «Отменить».

Для одной роли допустимо создавать только одно правило. Правило автоназначения представляет собой фильтр атрибутов для следующих объектов:

– «Пользователи»;
– «Роли»;
– «Учетные записи ресурса».

Перечень возможных атрибутов для объектов приведен в таблице 2.46.

Таблица 2.46 – Перечень возможных атрибутов для объектов

Объект	**Атрибут**
Пользователи	– «Электронная почта»; – «Фамилия»; – «Имя»; – «Отчество»; – «Вход в систему отключен»; – «Имя пользователя»; – «Телефон»; – «Табельный номер»; – «Линейный руководитель»; – «Подразделение»; – «Должность»; – «Идентификатор орг. присвоения»
Роли	– «Название»; – «Описание»; – «Информационная система»; – «Правило автоназначения»
Учетные записи ресурса	– «Название»; – «Описание»; – «Бенефициар»; – «Ресурс»; – «Форма ресурса»; – «Тип объекта коннектора»; – «Тип правила связывания»

В одном правиле допускается объединение нескольких фильтров. В этом случае будет осуществляться проверка на соответствие правилу по всем фильтрам.

Операции поддерживаемые фильтром правил автоназначения приведены в таблице 2.47.

Таблица 2.47 – Операции поддерживаемые фильтром правил автоназначения

Операция	**Отображение в конструкторе правила**	**Описание**	**Пример для REST API**
eq	Равно	Равенство с учётом регистра	lastName eq 'Глазырин'
eqic	Равно без уч. рег.	Равенство без учёта регистра	userName eqic 'glazyrin-s'
sw	Начинается с	Начинается с (start with)	userName sw 'GLAZYRIN'
co	Содержит	Содержит (contains)	userName co 'GLAZ'
tree	В иерархии	В иерархии (на данный момент операция поддерживается только для атрибута organization_id объекта user). Проверяет, что переданное значение содержится в иерархии ссылочного объекта	organization_id tree 16
in	Массив	Среди перечисленных значений. Массив данных поддерживается для строковых полей (string и textField) и поля типа «Поиск объекта» (lookup). Для строковых полей: – поиск ведется по точному значению, включая любые символы, в том числе и символ пробела; – значение массива может состоять из нескольких значений с использованием символов. При использовании символов «,» и «\» следует учитывать следующие особенности: – в случае использования запятой в режиме «Код» символ запятой отделяется знаком экранирования. В качестве знака экранирования используется двойной символ обратной косой черты (\\). Знак экранирования, предшествующий символу запятой, включает этот символ в результат. В режиме «Конструктор» символ запятой экранировать не требуется, необходимо ввести	1) lastName in "Глазырин,Барбашев" – выполнится для пользователей с фамилией «Глазырин» и «Барбашев». 2) lastName in "Месье,Глазырин,Барбашев"– выполнится для пользователей с фамилией «Месье» и «Глазырин» и «Барбашев». 3) lastName in "Месье\\, Глазырин,Барбашев" – выполнится для пользователей с фамилией «Месье, Глазырин» и «Барбашев». 4) position in 'Бухгалтер-аудитор,Главный бухгалтер,Аналитик\\, экономист' – выполнится для пользователей с должностью «Бухгалтер-аудитор» и «Главный бухгалтер» и «Аналитик,экономист»

Операция	Отображение в конструкторе правила	Описание	Пример для REST API
		подряд значения через запятую и нажать «Enter». Например, в конструкторе необходимо указать "111-222", "222-333\", "555-123", а в коде значение пропишется, как (/position in '"111-222"\\, "222-333\\\\"\\, "555-123"'); – в случае использования символа «\» в режиме «Код» в качестве знака экранирования используется тройной символ обратной косой черты (\\\). В режиме «Конструктор» экранировать символ «\» не требуется. Например, в конструкторе необходимо указывать "Инженер\Аналитик", а в коде значение пропишется, как (/position in 'Инженер\\\\Аналитик'). Для поля типа «Поиск объекта» (lookup) поиск ведется по точному или частичному значению	
pr	Присутствует	Присутствует (present)	title pr
lt	Меньше	Меньше (less than)	organization_id lt 2
le	Меньше или равно	Меньше или равно (less than or equal)	organization_id le 2
gt	Больше	Больше (greater than)	organization_id gt 3
ge	Больше или равно	Больше или равно (greater than or equal)	organization_id ge 3
and	И	Одновременное выполнение всех перечисленных условий	userName eqic 'GLAZYRIN-S' and lastName eq 'Глазырин'
or	ИЛИ	Выполнение хотя бы одного из перечисленных условий	userName eqic 'GLAZYRIN-S' or lastName eq 'Глазырин'
not	!	Операция отрицания	!(userName eqic 'GLAZYRIN-S')

Фильтр может иметь символьный тип или числовой. Символьный необходимо заключать в одиночные кавычки, например, firstName eq “Анатолий”. Для числового типа кавычки опускают.

Переоценка соответствия существующим правилам происходит в одном из следующих случаев:

1) Изменение атрибутов пользователя (выполняется переоценка конкретного пользователя).
2) Изменение атрибутов организации (выполняется переоценка всех пользователей).
3) Изменение правила автоматического назначения.

Если в условие правила добавлено неактуальное значение, отображается знак «!» и выводится предупреждение «Объект был удален!». Предупреждение отображается только в режиме конструктора. Варианты отображения значения удаленного объекта в условии:

1) Значение удаленного объекта отображается в поле. При этом возможно сохранение такого правила, но правило для данного условия не будет выполняться Комплексом. Если правило содержит другие условия с существующими объектами, то правило будет работать только для этих объектов.
2) Значение удаленного объекта отсутствует для отображения в поле. Поле считается не заполненным, при этом невозможно перейти в режим «Код» или сохранить правило. Необходимо удалить условие или указать новое значение.

2.16.2. Настройка правила связывания

Правило связывания (далее – запрос beneficiarQuery) настраивается в объекте «Учетная запись ресурса» (подраздел 2.3.1). Запрос beneficiarQuery выполняет поиск владельца для УЗ по УЗ из коннектора (не по форме ресурса). Правило связывания применяется только для целевой синхронизации.

Запрос beneficiarQuery может быть следующих типов:

1) Тип «correlationQuery». Правило корреляции, представляющее собой фильтр «queryFilter» для поиска подходящих владельцев объекта-источника (source-объекта). В правиле можно использовать атрибуты

source-объекта (коннектора) в качестве переменных (*username*). Пример кода с переменной ({username}):

```
"beneficiarQuery": "userName eqic '${username}'"
```

2) Тип «correlationScript». Скрипт, возвращающий подходящих владельцев по заданному source-объекту. В скрипте можно использовать source-объект (коннектора) в качестве переменной (source.username). Пример кода с переменной (source.username):

```
"beneficiarQuery": {
"type": "text/javascript",
"source": "var prefix = \\"LTGCORP\\\\\\\\\\"; var login = source.username; login =
login.indexOf(prefix) === 0 ? login.substring(8) : login; var users =
openidm.query(\\"managed/user\\", {\\"\_queryFilter\\": \\"/userName eqic '\\" + login +
\\"'\\"});var ret = (users && users.result) ? users.result : null; ret;"
  }
```

Пример скрипта javascript:

```
var prefix = "LTGCORP\\\\";
var login = source.username;
login = login.indexOf(prefix) === 0 ? login.substring(8) : login;
var users = openidm.query("managed/user", {"_queryFilter": "/userName eqic '" + login +
"'"});
var ret = (users && users.result) ? users.result : null;
ret;
```

Пример с указанием пути к файлу JS скрипта:

```
"beneficiarQuery": {
    "type": "text/javascript",
    "file": "script/shpCorrelate.js"
 }
```

2.16.3. Настройка правила разграничения доступа

Правило разграничения доступа определяет при каких условиях пользователи могут иметь роли или полномочия роли, в случае если они были назначены в обход Комплекса.

Настройка правил разграничения доступа выполняется администратором в подменю «Правила» меню «Админ. системы» и включает в себя следующие действия администратора:

– создание правила;

– редактирование правила;
– удаление правила.

2.16.3.1. Создание правила разграничения доступа

Для создания правила разграничения доступа администратору необходимо выполнить следующие действия:

1) Перейти в подменю «Правила» раздела «Админ. сист.», затем нажать кнопку «Создать». Появится страница создания правила.
2) Заполнить обязательные поля: «Имя», «Тип».
3) При необходимости заполнить необязательное поле «Описание».
4) Для активации действия правила необходимо перевести выключатель «Правило неактивно» в состояние «Правило активно».
5) В подразделе «Правило» для задания условий необходимо открыть конструктор правил.
6) Указать роль «Оператор SoD», на которого будут назначаться новые инциденты выявленные по данному правилу. Либо выбрать «Объект» и указать ИС, для которой задана роль оператора SoD.

Конструктор правила разграничения доступа позволяет задавать условия в графической форме или в виде кода. Для задания условий правила разграничения доступа в графической форме администратору необходимо выполнить следующие действия:

1) В разделе фильтра «Пользователь» выбрать атрибут объекта «Пользователь», далее выбрать «Условие», далее ввести значение для атрибута. Значение атрибута задается с тем же типом, что сам атрибут. Если не задано ни одно условие для фильтра «Пользователи», то правило применяется для всех пользователей Комплекса.
2) При необходимости в разделе фильтра «Пользователи» добавить условия, нажав кнопку «Добавить условия». Условия можно объединять логическими операторами «И», «ИЛИ».
3) В разделе фильтра «Роли» выбрать атрибут объекта «Роли», далее выбрать «Условие», далее ввести значение для атрибута. Значение

атрибута задается с тем же типом, что сам атрибут. Если не задано ни одно условие для фильтра «Роли», то правило не будет создано.

4) При необходимости в разделе фильтра «Роли» добавить условия, нажав кнопку «Добавить условия». Условия можно объединять логическими операторами «И», «ИЛИ».
5) Для обоих фильтров можно объединять или разделять условия, выполнять их группировку. Для необходимо использовать соответствующие кнопки «Объединить» и «Разделить».
6) Для удаления неактуального условия необходимо нажать кнопку «Удалить».
7) Для просмотра кода правила необходимо перейти в раздел «Код». В разделе «Код» также доступно редактирование условий правила. При возврате в раздел «Конструктор» код правила преобразуется в графической форме.

Поддерживаемые условные операторы приведены в таблице 2.48.

Таблица 2.48 – Поддерживаемые условные операторы

Поддерживаемые условные операторы	**Код**	**Назначение**
Равно	РАВНО	Находит полное соответствие значению. Поддерживается для всех типов атрибутов
Не равно	НЕ РАВНО	Находит все объекты, которые не соответствуют значению. Поддерживается для всех типов атрибутов
Присутствует	ПРИСУТСТВУЕТ	Возвращает значение «ИСТИНА» в случае, если значение атрибута найдено у объекта. Поддерживается для всех типов атрибутов
Отсутствует	ОТСУТСТВУЕТ	Возвращает значение «ЛОЖЬ» в случае, если значение атрибута не найдено у объекта. Поддерживается для всех типов атрибутов
В иерархии	В ИЕРАРХИИ	Выполняет поиск значения по всем дочерним связанным объектам. Поддерживается только для атрибута «Подразделение»
Не в иерархии	НЕ В ИЕРАРХИИ	Выполняет поиск значения только по текущему уровню объекта. Поддерживается только для атрибута «Подразделение»
Массив	in	Выполняет поиск среди перечисленных значений. Поддерживается для текстовых атрибутов в

Поддерживаемые условные операторы	Код	Назначение
		разделе фильтра «Пользователи»: – «Электронная почта»; – «Фамилия»; – «Имя»; – «Отчество»; – «Имя пользователя»; – «Телефон»; – «Табельный номер»; – «Должность»

Если в фильтр добавлено неактуальное значение, отображается знак и выводится предупреждение «Объект был удален!». Предупреждение отображается только в графическом режиме конструктора. Варианты отображения значения удаленного объекта в условии:

1) Значение удаленного объекта отображается в поле. При этом возможно сохранение такого правила, но правило для данного условия не будет выполняться Комплексом. Если правило содержит другие условия с существующими объектами, то правило будет работать только для этих объектов.
2) Значение удаленного объекта отсутствует для отображения в поле. Поле считается не заполненным, при этом невозможно перейти в режим «Код» или сохранить правило. Необходимо удалить условие или указать новое значение.

2.16.3.2. Редактирование правила разграничения доступа

Для редактирования правила разграничения доступа доступны все атрибуты, кроме типа правила. Администратору доступны все действий при редактировании, как при создании правила.

2.16.3.3. Удаление правила разграничения доступа

Для удаления правила разграничения доступа администратору необходимо выполнить следующие действия:

1) Перейти в подменю «Правила» раздела «Админ. сист.».
2) Отметить одно или несколько правил и нажать кнопку «Удалить».

3) В открывшемся окне подтверждения действия при необходимости исключить правила из списка удаляемых, после чего нажать кнопку «Удалить».
4) Для просмотра удаленных правил необходимо выбрать в фильтре списка статус «Удалено».

2.17. Настройка поиска

Комплекс обеспечивает поиск объектов в интерфейсе средствами внешней системы Opensearch. Настройка поиска включает следующие действия администратора:

1) Настройка подключения к поисковой системе.
2) Настройка поискового индекса.
3) Настройка фильтра поиска заместителя.
4) Настройка фильтра поиска бенефициара при оформлении заявки.

2.17.1. Настройка подключения к поисковой системе

Настройка подключения к поисковой системе выполняется в файле conf/search.connection.json. Конфигурационный файл conf/search.connection.json содержит следующие параметры настройки:

```
{
   "host": <адрес сервера службы поиска Opensearch, по умолчанию localhost>,
   "port": <порт службы поиска Opensearch, по умолчанию 9200>
}
```

Для настройки подключения к поисковой системе с включенной аутентификацией по логину и паролю необходимы данные авторизации. В конфигурационный файл conf/search.connection.json следует добавить обязательный параметр «auth» и следующие настройки:

- «type» – тип аутентификации. Значение по умолчанию «basic»;
- «principal» – имя пользователя;
- «password» – пароль пользователя.

Пример настройки подключения к поисковой системе с включенной аутентификацией по логину и паролю:

```
{
   "host": "localhost",
   "port": 9200,
   "protocol": "https",
   "auth": {
   "type": "basic",
   "principal": "searchLogin",
   "password": "searchPassword"
   }
}
```

2.17.2. Настройка поискового индекса

Настройка поискового индекса Opensearch выполняется в файле conf/search.settings.json.

Конфигурационный файл conf/search.settings.json содержит следующие параметры настройки:

1) enabled – признак того активен ли механизм поиска (true/false).
2) indexName – имя индекса, которое будет использоваться для хранения данных.
3) allowedValues – разрешенные символы в поисковых запросах.
4) minLength – минимальное количество символов в строке поиска для его запуска (иначе будет возвращаться пустой результат).
5) settings – настройки индекса (в примере, настройки по умолчанию).

Пример настройки конфигурационного файла conf/search.settings.json:

```
{
 "enabled": true,
 "indexName": "ankey",
 "allowedValues": "A-Za-zA-ЯЁa-яё0-9-@,.=+~'\" ",
 "minLength": 2,
 "settings": {
  "number_of_shards": 1,
  "number_of_replicas": 1,
  "analysis": {
   "analyzer": {
    "default": {
     "type": "custom",
     "tokenizer": "keyword",
     "filter": ["lowercase"],
     "char_filter": ["e_char_filter"]
    },
    "default_search": {
     "type": "custom",
     "tokenizer": "keyword",
     "filter": ["lowercase"],
     "char_filter": ["e_char_filter"]
```

```
      }
     },
     "char_filter": {
      "e_char_filter": {
       "type": "mapping",
       "mappings": [ "Ё => E", "ё => e" ]
      }
     }
    }
  }
}
```

2.17.3. Настройка фильтра поиска заместителя

Настройка фильтра поиска заместителя выполняется в конфигурационном файле conf/endpoint.lookup.substitution.json. Фильтр работает только для заявок на назначение ролей. Фильтр поиска заместителя позволяет пользователям выбирать заместителя из определенных подразделений, не обязательно своего.

Конфигурационный файл conf/endpoint.lookup.substitution.json содержит следующие параметры настройки:

```
{
   "accessibility": "<идентификатор области видимости>"
}
```

Области видимости заместителей приведены в таблице 2.49.

Таблица 2.49 – Области видимости заместителей приведены в таблице

Идентификатор области видимости	**Описание**
all	Любой сотрудник может установить заместителем любого другого сотрудника
limited	Администратор и оператор могут устанавливать заместителем любого сотрудника. Остальные пользователи могут выбирать заместителя среди следующих вариантов: – сотрудников из той же организации, что и сам пользователь; – своего непосредственного руководителя; – своих подчиненных

Примеры настройки фильтра в зависимости от типа видимости:

1) Полная видимость заместителей.

```
{
   "accessibility": "all"
}
```

2) Ограниченная видимость заместителей.

```
{
   "accessibility": "limited"
}
```

Если решения, предоставляемые системой, не подходят под бизнес-логику заказчика, возможна разработка своего решения. Описание приведено в документе «Руководство разработчика».

2.17.4. Настройка фильтра поиска бенефициара при оформлении заявки

Настройка фильтра поиска бенефициара при оформлении заявки выполняется в конфигурационном файле conf/beneficiary.picker.provider.json.

Фильтр работает только для заявок на назначение ролей.

Фильтр поиска бенефициара позволяет пользователям самостоятельно запрашивать для других пользователей, которые возвращаются поиском, роли в ИС.

Конфигурационный файл beneficiary.picker.provider.json содержит следующие параметры настройки:

```
{
   "name": "<идентификатор области видимости>",

      "enabled": <включение возможности смены бенефициара в заявках при создании, может принимать значения true или false>
}
```

Комплекс поддерживает следующую область видимости бенефициаров: «limited». При настройке идентификатора области видимости «limited» используются следующие варианты:

1) Администратор может менять бенефициара на любого сотрудника.
2) Оператор может менять бенефициара на любого управляемого им сотрудника.
3) Остальные пользователи могут выбирать заместителя среди следующих вариантов:
   - сотрудников из той же организации, что и сам пользователь;
   - своего непосредственного руководителя;

– своих подчиненных.

Включение возможности менять бенефициара, например:

```
{
   "name": "limited",
   "enabled": true
}
```

Если решение по ограничению области видимости, предоставляемое системой, не подходит, возможна разработка собственного решения, описание приведено в руководстве разработчика Комплекса.

В случае некорректной загрузки нестандартного решения, или, когда название сервиса в конфигурационном файле не совпадает с требуемым, будет включена область видимости по умолчанию «limited».

2.18. Настройка политики валидации

Политика валидации используется для определения условий, которым должно соответствовать значение атрибута объекта.

Система проверяет вводимое значение на соответствие политике. Если вводимое значение не соответствует политике, система выдает ошибку валидации.

Система поддерживает два типа политик:

– «PolicyFunction» вызывается при создании и любом изменении объекта (даже если изменился не тот атрибут, на который была настроена политика);
– «OptionalPolicyFunction» вызывается при заданном условии. По умолчанию это условие создание объекта или изменение значения валидируемого атрибута. Условие можно менять, переопределив соответствующий метод (подробнее в javadoc).

Примечание: Если при изменении состава политик типа «PolicyFunction» (настроенных на атрибут), существующие объекты, которые удовлетворяли старым политикам, перестали удовлетворять новым политикам, изменение состава политик следует выполнять с устранением всех ошибок валидации.

Все политики, кроме required, разрешают отсутствие значения у проверяемого атрибута (null).

Список и описание политик валидации приведен в таблице 2.50.

Таблица 2.50 – Список и описание политик валидации

Имя политики	Описание	Параметры конфигурации
required	Проверка, что значение не null и не пустая строка после удаления всех начальных и конечных пробелов	–
unique	Проверка, что нет другого объекта с таким же значением валидируемого атрибута. В случае, если значение строкового типа, проверка осуществляется без учёта регистра	–

cannotContainCharacters	Проверка, что в значении атрибута нет запрещенных символов	forbiddenChars - массив запрещенных строк.Например,
		
		"params": {
		"forbiddenChars": [
		"/"
		]
		}

maximumLength	Проверка максимальной длины	numChars - максимальная длина атрибута. Например, "params": { "numChars": 255}

regexpMatches	Проверка соответствия значения регулярному выражению	1) regexp - регулярное выражение.
		2) flags - флаг (необязательный).
		Допустимые значения:
		– \i – регистровая независимость;
		– \m – многострочный режим.
		Например,
		
		"params": {
		"regexp":
		"^([^\\s\\@])+\\@([^\\s\\@])+$"
		}

enum	Проверка, что значение среди множества допустимых	values — массив разрешённых значений. Например, "params": { "values": [

Имя политики	Описание	Параметры конфигурации
		"active", "disabled"] }

resourceExists	Проверка существования ресурса с заданными числовым идентификатором (ouid)	resource - имя ресурса для проверки (обязательный параметр).Например,
		
		"params": {
		"resource":"managed/approle"
		}

isInteger	Проверка целочисленности значения	–
multivaluedFieldsCombinationUnique	Проверка, что комбинация ключевых атрибутов у полномочий уникальна. Если вы попытаетесь выделить УЗ с двумя и более полномочиями, состав ключевых полей которых совпадает (отличий либо нет вообще, либо есть только в неключевых полях), выделение завершится ошибкой валидации. Если данная ошибка у вас произошла во время синхронизации УЗ, скорее всего у вас некорректно настроен состав ключевых полей (не определяет уникальность привилегии)	Переиспользовать политику нельзя

2.19. Управление фотографиями пользователей

Управление фотографиями пользователей включает в себя следующие события:

- загрузка;
- обновление;
- удаление.

Управление фотографиями пользователей выполняется администратором двумя способами:

– с помощью задач синхронизации. Применяется для управления фотографиями из внешней системы;
– в меню «Пользователи». Применяется для загрузки/редактирования фотографии при создании или редактировании пользователя. Подробнее приведено в документе «Руководство пользователя» в подразделе 2.4.

Управление фотографиями для всех событий выполняется в следующей последовательности:

1) Настройка параметров отображения загружаемых фотографий в файле conf/photo.json.
2) Синхронизация фотографий из внешней системы.
3) Синхронизация пользователей. Загружаются данные пользователей и осуществляется создание/обновление/удаление связи каждого пользователя с фотографией.

Параметры конфигурационного файла photo.json приведены в таблице 2.51.

Таблица 2.51 – Параметры конфигурационного файла photo.json

Атрибут	Тип	Описание	Пример
ttl	Число	Время в секундах, через которое фотография, которая не привязана ни к одному из пользователей, удалится из БД Комплекса	1800
maxFileSize	Число	Максимальный размер фотографии	4194304
mimeTypes	Объект	Поддерживаемые типы загружаемой фотографии	["image/jpeg","image/png"]
fileTypes	Объект	Поддерживаемые расширения файлов	["jpeg","jpg","png"]
resolution	Объект	Минимально допустимое разрешение загружаемой фотографии	{"width": 80, "height": 80}
minSidePreviewSize	Число	Размер в пикселях, до которого уменьшается оригинал фотографии, чтобы создать форму предпросмотра	200
previewSuffix	Строка	Суффикс, который добавится к имени файла фотографии при сохранении формы предпросмотра	-preview

Настроенные параметры в конфигурационном файле «photo.json» не должны превышать значения параметров из конфигурационного файла «document.json».

Фотографии загружаются/обновляются/удаляются в объекте «managed/document». При загрузки фотографии в объекте «managed/document» дополнительно генерируется уменьшенная копия изображения, которая используется для предпросмотра.

При удалении в процессе запуска синхронизации фотографий оба документа: фотография и уменьшенная копия изображения удаляются. При удалении в процессе запуска синхронизации пользователей удаляется только уменьшенная копия изображения.

Объект «managed/document» включает следующие поля:

- «content» – в параметр записывается визуальное содержимое в формате UrlBase64;
- «name» – в параметр записывается название изображения (пример, «image.jpeg»);
- «mimeType» – в параметр записывается mime-тип изображения (пример, «image/jpeg»);
- «size» – в параметр записывается размер изображения в байтах.

Каждая фотография в объекте «managed/document» уникальна. Идентификация фотографии происходит по совокупности полей объекта «managed/document».

При синхронизации фотографий следует учитывать следующие ограничения, настроенные по умолчанию в конфигурационном файле photo.json:

1) Доступные форматы: .jpg, .jpeg и .png, в противном случае возникает ошибка «400: Bad Request».
2) Доступные mime-типы: image/jpeg и image/png, в противном случае возникает ошибка «400: Bad Request».
3) Размер файла не более 4Мб, в противном случае возникает ошибка «400: Bad Request».

4) Разрешение фотографии не менее 80 пикселей с каждой стороны фотографии, в противном случае возникает ошибка «Размер кадра меньше 80x80!». При загрузке прямоугольного изображения учитывается размер по меньшей стороне.
5) В названии файла нельзя использовать пробелы и спецсимволы. Если их использование необходимо, то название должно быть перед отправкой закодировано стандартной URL кодировкой строк.

Для применения функциональности всех событий синхронизации администратору необходимо изменить настройки маппингов:

1) Синхронизации фотографий.
2) Синхронизации пользователей.

Пример маппинга синхронизации фотографий для всех событий (загрузки, обновления, удаления):

```
{
   "mapName": "newmappingphoto",
   "mapDesc": "new mapping photo",
   "mapType": "recon",
   "mapSource": "connector/newresource/userphoto",
   "mapTarget": "managed/document",
   "correlationQuery": "/name eqic '${filename}' and /mimeType eqic '${mimetype}' and /content eqic
'${content}'",
   "mapBody": {
      "properties": [{
            "source": "content",
            "target": "content"
         },
         {
            "source": "filename",
            "target": "name"
         },
         {
            "source": "mimetype",
            "target": "mimeType"
         },
         {
            "source": "",
            "target": "size",
            "transform": {
               "type": "text/javascript",
               "source": "source.content.length"
            }
         }
      ],
      "policies": [{
            "situation": "ABSENT",
```

```
            "action": "CREATE"
          },
          {
            "situation": "UNASSIGNED",
            "action": {
              "type": "text/javascript",
              "globals": {},
              "source": "var usage=(openidm.query('repo/documentusage', {'_queryFilter': '/document_id eq ' +
target._ouid }, ['usageType']).result[0].usageType);if (usage==='avatar') {action='DELETE'} else
{action='IGNORE'};action;"
            }
          }, {
            "situation": "FOUND",
            "action": "UPDATE"
          }
        ]
    }
}
```

Пример маппинга синхронизации пользователей для создания, обновления, удаления связи с фотографиями:

```
{   "mapName":"newmappingname",
  "mapType":"recon",
  "mapDesc":"new mapping name",
  "mapSource":"connector/newresource/account",
  "mapTarget":"managed/user",   "correlationQuery":"userName eqic '${username}'",
  "mapBody":{
    "policies":[
      {
        "action":"IGNORE",
        "situation":"CONFIRMED"
      },
      {
        "action":"UPDATE",
        "situation":"FOUND"
      },
      {
        "action":"CREATE",
        "situation":"ABSENT"
      },
      {
        "action":"EXCEPTION",
        "situation":"AMBIGUOUS"
      },
      {
        "action":"UNLINK",
        "situation":"MISSING"
      },
      {
        "action":"DELETE",
        "situation":"SOURCE_MISSING"
      },
      {
        "action":"IGNORE",
```

```
        "situation":"UNQUALIFIED"
      },
      {
        "action":"DELETE",
        "situation":"UNASSIGNED"
      },
      {
        "action":"IGNORE",
        "situation":"TARGET_IGNORED"
      }
    ],
    "properties":[
      {
        "source":"",
        "target":"userName",
        "transform":{
          "type":"text/javascript",
          "source":"source.username.toUpperCase();"
        }
      },
      {
        "source":"lastname",
        "target":"lastName"
      },
      {
        "source":"firstname",
        "target":"firstName"
      },
      {
        "source":"middlename",
        "target":"middleName"
      },
      {
        "source":"password",
        "target":"password"
      },
      {
        "source":"manager",
        "target":"managerId"
      },
      {
        "source":"",
        "target":"organization_id",
        "transform":{
          "type":"text/javascript",
          "source":"Number(\"${orgId}\");"
        }
      },
      {
        "target":"avatarId",
        "source":"userPhotoFileName",
        "transform":{
          "type":"text/javascript",
          "source":"if (source != null) {openidm.query('managed/document', { '_queryFilter': '/name eq \"' + source
+ '\"'}, ['_ouid']).result[0]._ouid;}"
        }
      }
```

```
    ]
  }
}
```

Объект «mapBody» содержит блок «policies», в котором описываются возможные события синхронизации.

За связь пользователей с фотографиями отвечает следующая часть в маппинге:

```
{
        "target":"avatarId",
        "source":"userPhotoFileName",
        "transform":{
          "type":"text/javascript",
          "source":"if (source != null) {openidm.query('managed/document', { '_queryFilter': '/name eq \"' + source
+ '\"'}, ['_ouid']).result[0]._ouid;}"

}
```

Когда поле «userPhotoFileName» из внешней системы указано в настройках маппинга пользователя, скрипт выполняет запрос документа из объекта «managed/document».

Подставляет его ouid, как «avatarId» у пользователя и осуществляет и создание/обновление/удаление связи каждого пользователя с фотографией.

За обновление фотографий отвечает следующая часть в настройке синхронизации фотографий:

```
{
  "situation": "FOUND",
  "action": "UPDATE"
}
```

2.20. Управление лицензиями

Управление лицензиями Ankey IDM включает следующие действия администратора:

1) Настройка параметров лицензирования в файле настроек conf/license.activator. Описание настроек приведено в приложении А.
2) Активация пользовательской лицензии.
3) Активация лицензии на коннектор.

Для активации пользовательской лицензии администратору необходимо выполнить следующие действия в меню «О продукте»:

1) Нажать кнопку «Активировать». Откроется форма для ввода ключа лицензии и E-mail адреса.
2) Ввести в обязательные поля ключ лицензии и E-mail адрес. В случае неверных данных отображаются сообщения об ошибках.
3) Нажать кнопку «Далее». После чего выполняется генерация запроса для формирования запроса на активацию.
4) Нажать кнопку «Скопировать», чтобы скопировать запроса на активацию.
5) Нажать ссылку «Система лицензирования». В новой вкладке откроется страницы активации системы лицензирования. Для подключения требуется наличие доступа в Интернет.
6) В окне системы лицензирования вставить скопированный запрос на активацию и нажать кнопку «Активировать».
7) Вставить ключ активации, который был отправлен на указанный email-адрес и нажать кнопку «Активировать». Сформируется сертификат.
8) Нажать кнопку «Скопировать в буфер обмена» и перейти в Ankey IDM для продолжения активации лицензии.
9) Нажать кнопку «Далее» и вставить сертификат в поле «Сертификат лицензии».
10) Нажать кнопку «Активировать». Отобразится окно «Продукт успешно активирован!».
11) Нажать кнопку «Закрыть» для закрытия окна. После чего в меню «О продукте» отобразится окно с актуальными данными о лицензии.

Обновление активированной лицензии Ankey IDM при приобретении дополнительных пользовательских лицензий выполняется аналогично действиям выше.

Активацию лицензии в кластере достаточно выполнить на одном узле кластера. При этом требуется наличие одинаковых ключей, хранимых в каталоге «security» во всех узлах кластера. В случае изменения ключей, хранимых в каталоге «security», активированная лицензия не может быть прочитана на другом узле.

2.20.1. Активация лицензии на коннектор

В меню «Админ. системы»/«Коннекторы» выполняется активация лицензии на один класс коннектора. Список коннекторов отображает следующую информацию:

- «Название» – название коннектора;
- «Описание» – описание коннектора;
- «Класс» – класс коннектора;
- «Статус активации» – статус активации лицензии;
- «Версия» – версия коннектора;
- «Техническая поддержка» – тип технической поддержки («базовая», «гарантийная» или «отсутствует») и сроки гарантийного обслуживания. Лицензии с истекшим сроком гарантийного обслуживания отображаются красным цветом и при наведении выводится подсказка «Срок действия истек», лицензии с действительным сроком гарантийного обслуживания отображаются зеленым цветом и при наведении выводится подсказка «Действительна».

Возможные статусы активации:

- «Активирован»;
- «Не активирован»;
- «Активация не требуется». Статус отображается для коннекторов, разработанных сторонними организациями.

Для активации лицензии на коннектор администратору необходимо выполнить следующие действия:

1) В списке коннекторов выбрать коннектор со статусом «Не активирован».
2) Нажать кнопку «Активировать».
3) Пройти активацию, выполнив аналогичные действия, приведенные в подразделе 2.20.

После активации лицензии статус активации принимает значение «Активирован». В случае, когда список содержит несколько коннекторов одного класса, достаточно активировать лицензию только одного коннектора. При активации

коннектора с одним классом у остальных коннекторов автоматически статус активации принимает значение «Активирован».

Активация коннектора, у которого не задан атрибут «productName», невозможна.

Активация лицензии на коннектор в кластере аналогична активации пользовательской лицензии для кластера. Описание приведено в подразделе 2.20.

2.21. Управление незавершенными задачами

Раздел «Незавершенные задачи» доступен администратору Комплекса и отображает список незавершенных задач.

Незавершенные задачи – это задачи создание/обновление/удаление УЗ в ЦС, завершенные с ошибкой. Незавершенные задачи автоматически повторяются периодическим заданием «failedsync». На вкладке «Информация» в блоке «Ошибка» выводится сообщение об ошибке и причина ошибки.

Список незавершенных задач может меняться в фоновом режиме при частом срабатывании периодического задания «failedsync». В подразделе «Незавершенные задачи» есть счетчик, на котором для контроля выполнения задач отображается количество незавершенных задач на текущий момент. Счетчик обновляется после совершения контекстного действия (удаление задач, запуск задач), либо по нажатию кнопки «Обновить». Если незавершенных задач нет, то счетчик отсутствует.

Рекомендуется отключить выполнение задания «failedsync» при работе с незавершенными задачами.

Примеры ситуаций, при которых могут возникать незавершенные задачи приведены в таблице 2.52.

Таблица 2.52 – Примеры ситуаций, при которых могут возникать незавершенные задачи

Ситуация	Операция	Причины	Решение
Такой пользователь уже существует в ЦС	Создание УЗ (CREATE)	Не соблюдена последовательность: 1) Синхронизация существующих УЗ. 2) Выделение новых УЗ	1) Провести синхронизацию с ЦС, после чего привяжется действующая УЗ. 2) Удалить

Ситуация	Операция	Причины	Решение
			неактуальную УЗ в статусе «Создаётся» у пользователя
ЦС недоступна (отключена, истек срок действия пароля служебной УЗ, заблокирована служебная УЗ)	Все операции	Неверная настройка подключения Комплекса к ЦС	Исправить настройки ресурса в части параметров подключения к ЦС. Дождаться автоматического повтора упавших заданий
ЦС отвергла атрибуты создаваемой/изменяемой УЗ (не заполнены обязательные поля, пароль не прошел парольную политику на стороне ЦС)	1) Создание УЗ (CREATE). 2) Изменение УЗ (UPDATE)	Неправильная настройка на стороне Комплекса. Форма ресурсов и маппинги не учитывают логику для атрибутов УЗ	1) Исправить логику заполнения атрибутов УЗ. 2) Вручную изменить атрибуты для проблемной УЗ на стороне Комплекса. Дождаться автоматического повтора
Валидация Комплекса отвергла атрибуты создаваемой/изменяемой УЗ (не заполнены обязательные поля)	1) Создание УЗ (CREATE). 2) Изменение УЗ (UPDATE)	Неправильная настройка на стороне Комплекса. Предзаполнение формы не учитывает логику для ее атрибутов	1) Исправить логику заполнения атрибутов УЗ. 2) Вручную изменить атрибуты для проблемной УЗ на стороне Комплекса. Дождаться автоматического повтора
Пользователь на стороне ЦС отсутствует (например, был удален напрямую)	1) Изменение УЗ (UPDATE). 2) Удаление УЗ (DELETE)	Возможно неправильная настройка синхронизации на стороне Комплекса или отключенная синхронизация с ЦС	Настроить мапинг синхронизации УЗ, указав для ситуации отсутствующего пользователя действие «Удаление» (DELETE). Дождаться выполнения синхронизации с ЦС, после чего на отсутствующую запись будет применено действия «Удаления» на стороне Комплекса

Список незавершенных задач содержит следующую информацию:

– «Тип» – тип операции (создание, обновление, удаление);

- «Приложение» – значение атрибута «resAccName» задачи синхронизации. В случае, когда пользователь еще не связан с какой-либо УЗР и значение атрибута «resAccName» не определить, отображается значение «Отсутствует»;
- «Учетная запись» – УЗ пользователя, для которой выполнялась задача;
- «Владелец» – ФИО пользователя, которому принадлежит УЗ;
- «Дата» – дата текущего события.

Незавершенные задачи в списке сортируются по дате события от наиболее ранних к наиболее поздним. Для настройки фильтрации незавершенных задач следует нажать кнопку «Фильтр» в правом углу строки поиска. Все настройки фильтра комбинируются между собой. Доступны следующие настройки фильтра:

1) Дата события.
2) Тип операции (создание, обновление, удаление).

Возможные действия администратора:

1) Просмотр списка событий незавершенных задач.
2) Редактирование события незавершенной задачи
3) Поиск событий в списке по ФИО владельца.
4) Удаление события из списка незавершенных задач.
5) Выборочный запуск незавершенных задач.

Для просмотра события незавершенной задачи администратору следует выбрать событие из списка. После чего на вкладке «Информация» в зависимости от типа события отображаются следующие данные:

- наименования атрибутов при операции «создание»;
- наименования атрибутов и разница их значений при операции «обновление».

Для редактирования события незавершенной задачи администратору следует:

1) Выбрать событие из списка и нажать кнопку «Редактировать». Кнопка «Редактировать» доступна для задач типа «Создание» (CREATE).
2) Внести изменения в форме «Редактирование учетной записи».

3) Нажать «Сохранить». Для отмены действия следует нажать «Отменить».

Для удаления событий из списка незавершенных задач администратору следует выбрать одно или несколько событий и нажать кнопку «Удалить». Откроется окно «Удаление незавершенных задач».

Для подтверждения действия следует нажать кнопку «Удалить».

Для отмены действия следует нажать кнопку «Отменить». Удаление события необходимо выполнять после устранения причин, вызвавших ошибку, в противном случае событие будет повторно создано.

Для выборочного запуска незавершенных задач следует выбрать задачу из списка. После выбора задачи кнопка «Запустить» станет активна и на ней появится счетчик. После запуска выбранной задачи в списке незавершенных задач меняется ее тайминг (запись времени). Можно запустить несколько (более одной) задач, выделив их чекбоксом. Когда задача, выбранная из списка незавершенных задач, запускается и выполняется, то список обновляется: из него удаляется выполненная задача.

2.22. Настройка автоматической генерации атрибутов

2.22.1. Генерация логина

Генерация логина выполняется автоматически, если при создании пользователя не заполнено поле «Имя пользователя». Настройки генерации логина содержатся в конфигурационном файле logingeneration.json. Конфигурационный файл logingeneration.json является обязательным и включает в себя следующие настройки:

1) «maxIterations» – отвечает за максимальное количество итераций, которые будут повторяться, если сгенерированный логин не пройдет политику уникальности логина (uniqueLogin). Если максимальное количество итераций будет превышено, то вернется исключение и пользователь не создастся.
2) «LoginGenerationScript» – конфигурация выполняемого скрипта со следующими параметрами:
   - type – тип скрипта (в данном примере JavaScript);
   - source – исходный код.

Пример настройки конфигурационного файла logingeneration.json:

```
{
  "maxIterations": 20,
  "LoginGenerationScript":{
    "type" : "text/javascript",
    "source":"user.userName = iteration == 0 ? user.lastName : user.lastName + iteration; user;"
  }
}
```

При написании скрипта двумя основными объектами являются:

- iteration – номер текущей итерации (нумерация начинается с нуля);
- user – json модель пользователя.

Скрипт должен возвращать json модель пользователя с заполненным полем userName.

Если логика генерации логина специфична и необходимо получать другие объекты из системы, то внутри скрипта возможно использовать CRUD операции с объектами. Пример генерации логина:

```
...
if(user.managerId){
    var params = {
        "_queryFilter": ("_ouid eq " + user.managerId)
    };
    var results = openidm.query("managed/user", params);
    if(results.result.length > 0){
        var manager = results.result[0];
        ...
    }
}
...
```

2.22.2. Генерация идентификатора организационного присвоения

При создании организационного присвоения если не заполнен атрибут «Идентификатор орг. присвоения», выполняется его автоматическая генерация. Генерация выполняется только при создании объекта организационного присвоения.

Конфигурационный файл conf/usrorgtabnumgeneration.json включает в себя настройку параметра «maxIterations». Параметр «maxIterations» отвечает за максимальное количество итераций, которые будут повторяться, если сгенерированный атрибут «Идентификатор орг. присвоения» не пройдет проверку на уникальность.

Значение атрибута будет генерироваться до тех пор, пока не достигнет максимального числа повторов. Если максимальное количество итераций будет превышено, то вернется исключение и сохранение данных не произойдет.

Пример конфигурационного файла conf/usrorgtabnumgeneration.json:

```
{
   "maxIterations": 20,
   "UsrOrgTabNumGenerationScript": {
      "type": "text/javascript",
      "source": "var tabNum = usrorg.organizationId + '_' + usrorg.userId; usrorg.tabNum = iteration == 0 ? tabNum
: tabNum + '_' + iteration; usrorg;"
   }
}
```

2.23. Управление конфигурацией

Управление конфигурациями выполняется в меню «Админ. системы»/ «Конфигурации» и включает в себя следующие действия администратора:

- создание конфигурации;
- экспорт конфигурации. Описание приведено в пункте 2.23.1;
- восстановление конфигурации. Описание приведено в пункте 2.23.2;
- импорт конфигурации. Описание приведено в пункте 2.23.3;
- удаление конфигурации. Описание приведено в пункте 2.23.4.

Создание конфигурации позволяет сохранить настройки следующих объектов Комплекса:

- config;
- connector;
- mapping;
- resform;
- restype;
- workflow.

Комплекс поддерживает следующие типы файлов: * json - для объектов config, connector, mapping, resform, restype. * xml - для объекта workflow (BPMN-схемы).

Для создания конфигурации администратору необходимо выполнить следующие действия:

1) Нажать кнопку «Создать».

2) Ввести название конфигурации, и при необходимости задать описание.
3) Нажать кнопку «Далее».
4) Выбрать объекты для создания конфигурации.
5) Нажать кнопку «Сохранить».

Список конфигураций отображает следующие данные:

1) «Название».
2) «Дата создания».
3) «Статус». Статус может принимать следующие значения:
   – «Изменена» – конфигурация, часть объектов которой не соответствуют настройкам объектов системы;
   – «Соответствует» – конфигурация, все объекты которой соответствуют настройкам объектов системы.
4) «Кем создано».

Для просмотра деталей конфигурации необходимо выбрать конфигурацию в списке, после чего детали отобразятся в правой части экрана на вкладке «Информация». Детали конфигурации содержат следующую информацию:

1) Основные данные.
2) Список объектов конфигурации. Рядом с удаленным объектом отображается знак  и выводится предупреждение «Объект удален». Рядом с измененным объектом отображается знак  и выводится предупреждение «Объект изменен».

2.23.1. Экспорт конфигурации

Для экспорта конфигурации администратору необходимо выполнить следующие действия:

1) Выбрать нужную конфигурацию из списка. Экспорт конфигурации не поддерживает множественный выбор.
2) Нажать кнопку «Экспорт». Откроется окно «Экспорт конфигурации».
3) Дождаться окончания формирования конфигурации. Файл формируется в виде архива .zip.

4) Дождаться окончания стандартного сохранения zip-архива. Отобразится сообщение «Конфигурация сохранена». В случае ошибки отображается сообщение «Ошибка при экспорте». Для повтора экспорта следует нажать кнопку «Попробовать еще раз».

2.23.2. Восстановление конфигурации

Для восстановления конфигурации необходимо выполнить следующие действия:

1) Выбрать конфигурацию в статусе «Изменена».
2) Нажать на кнопку «Применить конфигурацию». Откроется окно подтверждения.
3) Нажать на кнопку «Применить» в окне подтверждения. В случае успешной активации отобразится сообщение «Конфигурация успешно применена». Статус конфигурации изменится на «Соответствует». В противном случае отобразится сообщение «Ошибка применения конфигурации».

При восстановлении конфигурации объектов «resform» и «mapping» следует учесть:

1) Нельзя восстановить конфигурацию, которая содержит удаленные объекты «resform». При попытке восстановить отобразится сообщение «Ошибка применения конфигурации».
2) Нельзя восстановить конфигурацию с объектом «mapping», который имеет те же характеристики (по совокупности параметров «mapType», «mapSource» и «mapTarget»), что и объект «mapping» текущей конфигурации. Для восстановления такого объекта необходимо сначала удалить «mapping», а затем восстановить конфигурацию с этим объектом.

2.23.3. Импорт конфигурации

Для импорта конфигурации администратору необходимо выполнить следующие действия:

1) Нажать кнопку «Импорт». Откроется окно «Импорт конфигурации».

2) Заполнить уникальное значение обязательного поля «Название». Если введено не уникальное значение, отображается «Объект уже существует».
3) Нажать кнопку «Добавить файл». На загрузку файла действуют следующие ограничения:
   – можно импортировать только один файл;
   – доступный тип файла: zip;
   – размер файла не более 10 МБ.
4) После добавления файла отобразится сообщение «Файл загружен без ошибок». Ниже отображаются импортируемые объекты конфигурации.
5) Нажать кнопку «Импортировать». Отобразится сообщение «Конфигурация успешно импортирована». В случае ошибки отображается сообщение «Ошибка при импорте». Для повтора импорта следует нажать кнопку «Попробовать еще раз». Конфигурация импортируется без применения. Восстановление конфигурации приведено в пункте 2.23.2.

Перечень возможных предупреждений при импорте конфигурации:

1) Объект не прошел валидацию по политике.
2) «Не поддерживаемый тип файла в объекте» отображается, если файл имеет неверный тип или содержит ошибки открытия архива (например, на архив установлен пароль, нарушена целостность архива).
3) "«Ошибка при загрузке»" отображается, если объекты конфигурации внутри файла не соответствуют структуре конфигурации.

После импорта объектов «workflow» рабочие потоки отображаются в меню «Админ. системы»/«Бизнес-процессы». Описание рабочих потоков приведено в подразделе 2.6.

2.23.4. Удаление конфигурации

Для удаления конфигурации администратору необходимо выполнить следующие действия:

1) Выбрать нужную конфигурацию из списка. Поддерживается множественный выбор.
2) Нажать на кнопку «Удалить». Откроется окно подтверждения «Удаление конфигураций».
3) Нажать на кнопку «Удалить» в окне подтверждения. В случае успешного удаления отобразится сообщение «Конфигурации успешно удалены». В случае ошибки отображается сообщение «Ошибка при удалении конфигураций». Для повтора удаления следует выполнить действия сначала.

2.24. Настройка импорта ролей через коннектор «Excel Matrix Upload»

Для настройки импорта ролей через коннектор «Excel Matrix Upload» необходимо внести в конфигурационный файл excelexport.json следующие изменения:

```
{
"_id": "excelexport",
"characterEncoding": "DEFAULT",
"rbacConfig": {
"directoryPath": "/home/vagrant/ankey/samples",
"importRbacModelTaskId": "schedule-excel-import-task",
"reconRbacModelTaskId": "schedule-excel-target-recon"
}
```

Настройка содержит следующие обязательные параметры:

- «directoryPath» - путь к директории с xls/xlsx файлами, прописанный в настройках коннектора;
- «importRbacModelTaskId» - id периодического задания для считывания данных из excel файла и записи их в rf_таблицу;
- «reconRbacModelTaskId» - id периодического задания для создания ролей на основе данных в rf_таблице.

2.25. Настройка для отображения столбцов в подразделе «Учетные записи»

Комплекс позволяет настроить отображение столбцов в подразделе «Учетные записи» в меню «Пользователи». Можно изменять отображение столбцов: добавлять новые, скрывать стандартные, менять порядок.

Настройки выполняются в конфигурационном файле ui.custom.table.json.

Пример конфигурационного файла ui.custom.table.json для нового числового столбца «resaccount_id»:

```
{
  "userAccounts":{
    "columns":[
      {
        "headerName":"number",
        "field":"resaccount_id"
      }
    ],
    "order":[
      "createDate",
      "accountType",
      "resaccount_id"
    ]
  }
}
```

Добавление столбцов выполняется в параметре «columns», который должен содержать следующие обязательные поля:

1) Поле «headerName» – заголовок столбца, ключ транслитерации.
2) Поле «field» – поле для отображения заголовка в столбце. Указывается значение, полученное на стороне backend-сервиса.

Основные варианты значений для поля «field»:

- beneficiar_objectid – идентификатор бенефициара;
- updateDate – дата изменения;
- resaccount_id – идентификатор УЗР;
- accountType – тип УЗ, например основная или дополнительная;
- beneficiarPath – управляемый объект, который является бенефициаром. Например, пользователь;
- resformtable_id – идентификатор таблицы, в которой хранятся УЗ;
- resFormName – имя формы ресурса;

- passwordPolicyName – название парольной политики, если она задана для УЗР;
- resAccDesc – описание УЗР;
- resAccName – название УЗР или «Приложение»;
- resource_id – идентификатор ресурса;
- displayField – отображаемое имя УЗ;
- resFormId – идентификатор формы ресурса;
- status – статус УЗ пользователя в УЗР;
- createDate – дата создания УЗ.

Перевод для заголовка столбца задается в файлах для русской локализации translation_ru.properties и англоязычной локализации translation_en.properties. Файлы перевода расположены в директории localization/i18n. Для русскоязычной локализации требуется выполнить конвертацию текста в кодировку JS/JAVA. Для изменения заголовка существующего столбца следует изменить его перевод по умолчанию.

В параметре «order» перечисляются столбцы Комплекса по умолчанию и новые для отображения их в определенном порядке. В интерфейсе столбцы будут отображаться слева направо. Интерфейс будет отображать только те столбцы, которые перечислены в параметре «order». Параметр «order» необязательный и если он отсутствует, то новые столбцы будут отображаться в интерфейсе справа от существующих.

Возможные типы данных в ячейке столбца:

- текст (число);
- переведенный текст;
- дата;
- дата и время;
- статус.

Пример настройки с разным типом данных:

1) Текст (число):

```
{
headerName: 'templates.managed.form.user.accounts.application',
field: 'resAccName'
}
```

2) Переведенный текст:

```
{
 headerName: 'templates.managed.form.user.accounts.type',
 field: 'accountType',
 type: 'translatedText',
 valuePrefix: 'templates.managed.form.user.accounts.type.'
}
```

3) Дата:

```
  {
  headerName: 'templates.managed.form.user.accounts.date',
  field: 'createDate',
  type: 'date',
  maxWidth: 150
  }
```

4) Дата и время:

```
{
 headerName: 'dateTime',
 field: 'updateDate',
 type: 'dateTime'
}
```

5) Статус:

```
{
 headerName: 'templates.users.columns.status',
 field: 'status',
 type: 'status',
 valuePrefix: 'templates.managed.form.user.accounts.status.',
 maxWidth: 150,
 params: {
 active: 'green',
 deleted: 'grey',
 deleting: 'orange',
 disabled: 'red',
 provisioning: 'blue'
 }
}
```

Перевод нового значения в ячейке необходимо прописать в файлах локализации аналогично, как для заголовка. Ключ транслитерации значения в ячейке складывается из «valuePrefix» и «item.field», где:

- «valuePrefix» – собственное название, которое нужно задать для нового столбца;
- «item.field» – название на стороне backend-сервиса.

Если «valuePrefix» отсутствует в конфигурационном файле, то значение отобразится без перевода.

Пример перевода заголовка для столбца «status»:

templates.managed.form.user.accounts.status=\u0421\u0442\u0430\u0442\u0443\u0441

Пример перевода значения «active» в столбце «status»:

templates.managed.form.user.accounts.status.active=\u0410\u043A\u0442\u0438\u0432\u043D\u0430

Пример конфигурационного файла ui.custom.table.json с разным типом данных:

```
{
  "userAccounts": {
    "columns": [
      {
        "headerName": "templates.managed.form.user.accounts.date",
        "field": "createDate",
        "type": "date",
        "maxWidth": 150
      },
      {
        "headerName": "templates.managed.form.user.accounts.application",
        "field": "resAccName"
      },
      {
        "headerName": "templates.managed.form.user.accounts.type",
        "field": "accountType",
        "type": "translatedText",
        "valuePrefix": "templates.managed.form.user.accounts.type."
      },
      {
        "headerName": "dateTime",
        "field": "updateDate",
        "type": "dateTime"
      },
      {
        "headerName": "templates.users.columns.status",
        "field": "status",
        "type": "status",
        "valuePrefix": "templates.managed.form.user.accounts.status.",
        "maxWidth": 150,
        "params": {
          "active": "green",
          "deleted": "grey",
          "deleting": "orange",
          "disabled": "red",
          "provisioning": "blue"
        }
      }
    ],
    "order": [
      "resAccName",
```

```
      "createDate",
      "updateDate",
      "accountType",
      "status"
    ]
  }
}
```

3. НАСТРОЙКА МАССОВЫХ ОПЕРАЦИЙ

Конфигурационный файл bulk.operation включает в себя настройку следующих параметров:

- «enabled» – включение/выключение возможности отслеживания массовых операций.

Кроме этого, имя ресурса включается в настройки как параметр, для которого выполняется настройка массовой операции. Указанный параметр может принимать следующие значения:

- «managed/user»;
- «managed/organization»;
- «managed/usrapprole»;
- «managed/is»;
- «resform/*", где "*» означает любое имя resform. Например, для подсчета массовых операций для какой-нибудь определенной resform необходимо указать: «resform/ad».

В свою очередь, для каждого имени ресурса есть собственный массив конфигураций массовых операций одного типа, в котором каждая конфигурация содержит следующие параметры (в скобках приведены значения по умолчанию):

- «enabled» – включение/выключение возможности отслеживания массовых операций данного типа (true);
- «requestType» – тип отслеживаемого запроса массовых операций (нет);
- «countingObject» – объект, который подсчитывают массовые операции («ENTITY»);
- «operationSource» – источник данных, из которых отслеживаются массовые операции («RECON»);
- «maxOperationsAllowed» – максимально допустимое количество операций (0);
- «duration» – период времени, за который отслеживается превышение порога (0);

– «timeUnit» – единица измерения времени («MINUTES»).

Возможные значения для параметра " requestType ":

– «create» – создание;

– «update» – обновление;

– «delete» – удаление.

Возможные значения для параметра " countingObject ":

– «ENTITY» – сам объект для которого настраивается конфиг (managed/user, resform/* и т.д.);

– «ENTITLEMENTS» – вместо объекта подсчитываются полномочия. Применимо только для “resform/*” и «requestType» – «update».

Возможные значения для параметра «operationsSource»:

– «RECON» – данные поступают из коннектора;

– «MANUAL» – источником данных является сам Комплекс (не целевая система и не доверенный источник). Например, данные поступили в Комплекс через REST API.

Возможные значения для параметра «duration» (может принимать только целочисленные значения):

– 0 (ноль, подсчет не работает);

– больше нуля (>0), ведется подсчет массовых операций.

Возможные значения для параметра «timeUnit»:

– «HOURS»;

– «MINUTES»;

– «SECONDS».

Пример конфигурационного файла:

```
{
 "enabled": true,
 "resform/ad": [
  {
   "requestType": "update",
   "maxOperationsAllowed": 100,
   "duration": 10
  }
 ],
 "resform/*": [
```

```
   {
    "requestType": "update",
    "maxOperationsAllowed": 50,
    "duration": 50,
    "timeUnit": "HOURS",
    "countingObject": "ENTITY"
   },
   {
    "requestType": "update",
    "maxOperationsAllowed": 100,
    "duration": 50,
    "timeUnit": "HOURS",
    "enabled": true,
    "operationsSource": [
     "RECON"
    ],
    "countingObject": "ENTITLEMENTS"
   }
  ],
  "managed/user": [
   {
    "requestType": "create",
    "maxOperationsAllowed": 1,
    "duration": 10,
    "timeUnit": "MINUTES",
    "enabled": true
   },
   {
    "requestType": "update",
    "maxOperationsAllowed": 1,
    "duration": 10,
    "timeUnit": "MINUTES",
    "enabled": false,
    "operationsSource": [
     "MANUAL"
    ]
   }
  ]
}
```


Настройка порога является общей для всех типов объектов и типов операций по действию (CREATE/UPDATE/DELETE), но регистрация массовой операции выполняется для каждого объекта отдельно.

3.1. Настройка согласований

Просмотр настройки согласований выполняется в меню «Админ. системы»/ «Настройка согласований».

Меню «Настройка согласований» отображает администратору настроенные бизнес-процессы на действия в системе.

Список «Настройка согласований» содержит следующие данные:

- тип заявки;
- бизнес-процесс.

В списке не отображаются типы заявок, которые формируются системой на периодической основе (периодическим заданием).

3.1.1. Создание настройки согласования

Для создания настройки согласования администратору следует выполнить следующие действия:

1) Перейти в меню «Админ. системы»/ «Настройка согласований».
2) Нажать на кнопку «Создать». Откроется окно «Создание настройки согласования». Кнопка «Создать» заблокирована, если все возможные типы заявок уже заданы.
3) Выбрать из выпадающего списка «Тип заявки». В выпадающем списке с типом заявок отображаются только те типы, которые еще не настроены в системе.
4) Выбрать из выпадающего списка «Бизнес-процесс».
5) Нажать на кнопку «Сохранить». Для отмены действий следует нажать на кнопку «Отменить». В случае ошибки отображается сообщение «Ошибка при создании настройки согласования».

3.1.2. Редактирование настройки согласования

Для редактирования настройки согласования администратору следует выполнить следующие действия:

1) Перейти в меню «Админ. системы»/ «Настройка согласований».
2) Нажать на кнопку «Редактировать». Откроется окно «Редактирование настройки согласования», в котором представлена информация о типе заявки и бизнес-процессе.
3) Выбрать из выпадающего списка «Бизнес-процесс».
4) Нажать на кнопку «Сохранить». Для отмены действий следует нажать на кнопку «Отменить». В случае ошибки отображается сообщение «Ошибка при редактировании настройки согласования».

3.1.3. Удаление настройки согласования

Для удаления настройки согласования администратору следует выполнить следующие действия:

1) Перейти в меню «Админ. системы»/ «Настройка согласований».
2) Нажать на кнопку «Удалить». Откроется окно «Удаление настройки согласования», в котором представлена информация о типе заявки и бизнес-процессе.
3) Подтвердить удаление, нажав на кнопку «Удалить». Для отмены действий следует нажать на кнопку «Отменить». В случае ошибки отображается сообщение «Ошибка при удалении настройки согласования».

ПРИЛОЖЕНИЕ А

В данном приложении приводятся примеры выполнения запросов с использованием REST API.

Пример 1. Самостоятельная смена пароля пользователем.

Обязательные параметры запроса:

– «oldPassword» – значение старого пароля;

– «newPassword» – значение нового пароля.

Пример Rest-запроса для самостоятельной смены пароля пользователем, например, petrov-p:

```
C:\> curl--cacert self-signed.crt ^
--header "Content-Type: application/json" ^\
--header "X-Ankey-Username: ankey" ^\
--header "X-Ankey-Password: ankey" ^\
--request POST ^
--data "{
        "oldPassword": "qwe",
        "newPassword": "newPasswordValue"}"^
'http://localhost:port/ankey/endpoint/user?_action=changePassword'\
```

Пример 2. Добавление фотографии пользователя в профиль.

Основные параметры запроса при добавлении фотографии в профиль:

1) «df8123b3-4206-4caf-bd6d-6f2c17083982» – ID пользователя.
2) 'file=@/home/vagrant/photo.jpg' – путь к файлу с фотографией. Путь до файла указывается либо полный, либо относительно директории откуда curl запускается.

Пример Rest-запроса добавления фотографии пользователя в профиль:

```
curl -X POST
--header 'X-Ankey-Username: ankey'
--header 'X-Ankey-Password: ankey'
--form 'json={
   "name": {"$ref":"cid:file#filename"},
   "mimeType": {"$ref":"cid:file#mimetype"},
   "content": {"$ref":"cid:file#content"},
   "size": {"$ref":"cid:file#size"}
};type=application/json'
--form 'file=@/home/ankey/photo.jpg;type=image/jpeg' 'http://localhost:port/ankey/endpoint/user/df8123b3-4206-
4caf-bd6d-6f2c17083982/photo'
```

При добавлении фотографии недопустимо менять или устанавливать фотографию у удаленного сотрудника, в противном случае возникает ошибка «404: Not Found».

Другие ограничения приведены в подразделе 2.19.

Пример 3. Процесс лицензирования Ankey IDM.

Перед выполнением действий по лицензированию Ankey IDM следует убедиться, что сервер приложений Ankey IDM развернут на ОС Linux x64. Все REST-запросы выполняются строго на той машине, где развернут активируемый Ankey IDM. В противном случае полученный сертификат может оказаться недействительным.

Администратору следует выполнить следующие шаги:

1) Проверить настройки конфигурации активатора.
2) Сгенерировать offline-запрос для проведения активации.
3) Получить сертификат на сервере лицензирования.
4) Применить полученный сертификат в Ankey IDM.
5) При необходимости осуществить проверку лицензии.

Шаг 1. Для проверки настройки конфигурации активатора следует выполнить Rest -запрос проверки актуальности настроек активатора:

GET: http://host:port/ankey/config/license.activator

Ответ содержит следующие настройки активатора:

- activationResource – ресурс для проведения завершающих этапов по активации ключа и принимает значение «https://license.gaz-is.ru/offlineActivate»;
- productName – имя продукта, совпадает с именем продукта, на который выписан ключ на сервере лицензирования и принимает значение «Ankey IDM»;
- maxActiveUsersPropertyName – имя свойства лицензии, хранящее максимально допустимое количество пользователей, совпадает со свойством продукта Ankey IDM на сервере лицензирования и принимает значение «user_count».

Шаг 2. Для генерации offline-запроса для проведения активации следует выполнить следующий Rest-запрос:

```
POST: http://host:port/ankey/endpoint/license/offline?_action=activate
Body:
{
    "email":"admin-name@gaz-is.ru",
    "licenseKey":"Y7AL1-6SEHE-COL0E-GE9"
}
```

Основные параметры запроса на генерацию offline-запроса:

- email – электронная почта администратора;
- licenseKey – ключ активации.

Ответ на запрос по активации ключа:

```
{
   "certificationRequest": {
         "email": "admin-name@gaz-is.ru",
         "license_request": "--CERTIFICATE REQUEST--"},
   "activationResource": "https://license.gaz-is.ru/offlineActivate"
}
```

Запрос по активации ключа содержит следующие параметры:

- certificationRequest – содержит сгенерированный offline-запрос, который используется для проведения активации на сервере лицензирования;
- activationResource – содержит ссылку для перехода на ресурс для проведения завершающих этапов по активации ключа.

Шаг 3. Получить сертификат на сервере лицензирования, используя данные полученные на шаге 2.

Шаг 4. Выполнить Rest-запрос применения сертификата в Ankey IDM. Перед выполнением запроса следует выполнить форматирование полученного сертификата, заменив все переносы строк на пробелы. После чего выполнить запрос применения сертификата в Ankey IDM:

```
PUT: http://host:port/ankey/config/license
Body:
{
   "certificate": "---BEGIN CERTIFICATE--- SAMPLE ---END CERTIFICATE--- "
}
```

Шаг 5. Для проверки лицензии необходимо выполнить следующий Rest-запрос: GET: http://host:port/ankey/endpoint/license

Ответ на запрос проверки лицензии содержит следующие параметры:

– productName – имя продукта;
– activeUsers – текущее количество активных пользователей в Ankey IDM;
– maxActiveUsers – максимально допустимое количество активных пользователей Ankey IDM (количество пользовательских лицензий), заданное при создании/редактировании лицензии на сервере лицензирования.

Пример ответа на запрос проверки лицензии:

```
{
    "_id": "",
    "_rev": "",
    "productName": "Ankey IDM",
    "activeUsers": 0,
    "maxActiveUsers": 10
}
```

Пример 4. Смена разделителя в файле экспорта ролей.

Для смены разделителя необходимо внести изменения в конфигурационный файл csvexport.json под УЗ «Ankey». Запроса для смены разделителя содержит следующие параметры:

– «fieldDelimiter» – разделитель полей. Доступные символы («\t», «;», «,», « », «|»);
– «newLineString» – знак конца строки, перевода на новую строку. Доступные символы («\n» и «\r\n»).

Пример запроса для смены разделителя в файле экспорта ролей:

```
PUT /ankey/config/csvexport

Body:

{
"fieldDelimiter": ";",
"newLineString": "\n"
}
```

Пример 5. Смена кодировки для экспортирумого файла excel.

```
PUT /ankey/config/excelexport

Body:
```

```
{
"characterEncoding": "ANSI"
}
```

Ниже приведены примеры для создания маппинга:

1) Пример использования Rest-запроса для создания маппинга распространения данных в ЦС:

```
{
   "mapName": "xml_connectorAuto_create",
   "mapType": "provision",
   "mapDesc": "rf_xml to xml",
   "mapSource": "resform/XML1",
   "mapTarget": "connector/auto",
   "mapBody": {
      "policies": [],
      "properties": [{
            "source": "username",
            "target": "username"
         }]
   }
}
```

2) Пример использования Rest-запроса для создания маппинга синхронизации УЗ из ЦС:

```
{
   "mapName": "xml_recon",
   "mapType": "recon",
   "mapDesc": "xml to rf_xml",
   "mapSource": "connector/auto",
   "mapTarget": "resform/XML1",
   "correlationQuery": {
      "type": "text/javascript",
      "source": "var qry = { '_queryFilter': 'username eq \"' + source.username + '\"' }; qry;"
   },
   "mapBody": {
      "policies": [{
            "situation": "CONFIRMED",
            "action": "UPDATE"
         },
         {
            "situation": "FOUND",
            "action": "UPDATE"
         },
         {
            "situation": "ABSENT",
            "action": "CREATE"
         },
         {
            "situation": "AMBIGUOUS",
            "action": "EXCEPTION"
         },
```

```
            {
               "situation": "MISSING",
               "action": "UNLINK"
            },
            {
               "situation": "SOURCE_MISSING",
               "action": "EXCEPTION"
            },
            {
               "situation": "UNQUALIFIED",
               "action": "IGNORE"
            },
            {
               "situation": "UNASSIGNED",
               "action": "IGNORE"
            },
            {
               "situation": "TARGET_IGNORED",
               "action": "IGNORE"
            }
         ],
         "properties": [{
            "source": "username",
            "target": "username"
         }]
      },
      "scripts": {
         "result": {
            "type": "text/javascript",
            "file": "script/reconresult.js"
         },
         "onCreate": {
            "type": "text/javascript",
            "file": "script/onCreate.js"
         },
         "onUpdate": {
            "type": "groovy",
            "source": "println \"onUpdateScript\";"
         },
         "onDelete": {
            "type": "text/javascript",
            "source": "console.log(\"onDeleteScript\");"
         }
      }
}
```

3) Пример использования Rest-запроса для создания маппинга распространения атрибутов из объекта «Пользователь» в форму ресурса с заданными условиями (condition):

```
{
   "mapName": "ad_provision_user_to_resform",
   "mapType": "provision",
   "mapDesc": "Mapping for provision from managed/user to aresform/ADForm on Update",
```

```
  "mapSource": "managed/user",
  "mapTarget": "resform/ADForm",
  "mapBody": {
    "policies": [
      ..............
    ],
    "properties": [{
          "source": "lastName",
          "target": "sn",
          "condition": {
            "type": "text/javascript",
            "source": "(object.lastName!=null && !object.lastName==")"
          }
      },
      {
          "source": "firstName",
          "target": "givenname",
          "condition": {
            "type": "text/javascript",
            "source": "(object.firstName!=null && !object.firstName==")"
          }
      }
      ...................
    ]
  }
}
```

4) Пример маппинга с указанием действия (action) в правиле маппинга (properties):

```
{
  "mapName": "xml_connectorAuto_create",
  "mapType": "provision",
  "mapDesc": "rf_xml to xml",
  "mapSource": "resform/XML1",
  "mapTarget": "connector/auto",
  "mapBody": {
    "policies": [],
    "properties": [{
      "source": "username",
      "target": "username"
    }, {
      "action": "create",
      "source": "password",
      "target": "password"
    }, {
      "action": "update",
      "source": "password",
      "target": "password",
      "transform": {
        "type": "text/javascript",
        "source": "openidm.decrypt(source)"
      }
    }]
```

```
    }
}
```

5) Пример маппинга с указанием скриптов (scripts):

```
{
   "mapName": "xml_recon",
   "mapType": "recon",
   "mapDesc": "xml to rf_xml",
   "mapSource": "connector/auto",
   "mapTarget": "resform/XML1",
   "mapBody": {
      "policies": [],
      "properties": [......]
   },
   "scripts": {
      "result": {
         "type": "text/javascript",
         "file": "script/reconresult.js"
      },
      "onCreate": {
         "type": "text/javascript",
         "file": "script/onCreate.js"
      },
      "onUpdate": {
         "type": "groovy",
         "source": "println \"onUpdateScript\";"
      },
      "onDelete": {
         "type": "text/javascript",
         "source": "console.log(\"onDeleteScript\");"
      }
   }
}
```

Пример 6. Убрать видимость кнопки «Сменить пароль» в меню «Мой профиль».

Пример REST запроса на изменение конфигурационного файла ui-configuration.json:

```
PATCH: /ankey/config/ui/configuration

[
{
"operation": "replace",
"field": "configuration/hideChangePassword",
"value": true
}
]
```

Возможные значения параметра «value»:

– false - включает видимость кнопки «Сменить пароль» (установлено по умолчанию).
– true - выключает видимость кнопки «Сменить пароль».

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

IDM	—	Identity Manager
JDK	—	Java Development Kit
SSL	—	Secure Sockets Layer
TLS	—	Transport Layer Security
БД	—	база данных
БП	—	бизнес пакет
ИС	—	информационная система
ОС	—	операционная система
СУБД	—	система управления базами данных
УЗ	—	учетная запись
УЗР	—	учетная запись ресурса
ЦС	—	целевая система

Лист регистрации изменений

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводи-тельного докум. и дата	Подп.	Дата
	измененных	замененных	новых	аннулированных					

Изм.	Подп.	Дата