|  |  |
| --- | --- |
| УТВЕРЖДАЮ | УТВЕРЖДАЮ |
| Заместитель руководителя  Федеральной службы государственной регистрации, кадастра и картографии | Заместитель генерального директора  Закрытого акционерного общества  «АМТ-ГРУП» |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Б. Приданкин  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 г. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.П. Лохин  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 г. |

ГОСУДАРСТВЕННЫЙ КОНТРАКТ

ОТ 16.10.2017 Г. № 0068-10-17 НА ВЫПОЛНЕНИЕ РАБОТ ПО ТЕМЕ:

«СОЗДАНИЕ ЕДИНОГО ГОСУДАРСТВЕННОГО РЕЕСТРА НЕДВИЖИМОСТИ: РАЗРАБОТКА ПОДСИСТЕМЫ СОПРОВОЖДЕНИЯ КАДАСТРОВОЙ ОЦЕНКИ»

ЭТАП №4

2018

Содержание

[1 Общие сведения о Подсистеме 4](#_Toc522573722)

[1.1 Наименование Подсистемы 4](#_Toc522573723)

[1.2 Наименование Заказчика 4](#_Toc522573724)

[1.3 Наименование Исполнителя 4](#_Toc522573725)

[1.4 Цели, назначение и области использования Подсистемы 4](#_Toc522573726)

[1.4.1 Цели создания Подсистемы 4](#_Toc522573727)

[1.4.2 Назначение и области использования Подсистемы 4](#_Toc522573728)

[1.5 Комплектность Подсистемы 5](#_Toc522573729)

[2 Описание программных средств 6](#_Toc522573730)

[2.1 Общие сведения 6](#_Toc522573731)

[2.2 Функциональное назначение 6](#_Toc522573732)

[2.3 Используемые технические средства 8](#_Toc522573733)

[3 Структура интеграционных компонентов Подсистемы и способы реализации интеграции 11](#_Toc522573734)

[3.1 Структура интеграционных компонентов 11](#_Toc522573735)

[3.1.1 Программные клиенты и сервисы 11](#_Toc522573736)

[3.1.2 Специализированные обработчики событий 11](#_Toc522573737)

[3.2 API ПСКО 11](#_Toc522573738)

[3.3 Сторонние функции 12](#_Toc522573739)

[3.4 Точки расширения 12](#_Toc522573740)

[3.5 Сторонние пакетные задачи 12](#_Toc522573741)

[4 Настройка интеграции 13](#_Toc522573742)

[4.1 Программные интерфейсы (SOAP API) 13](#_Toc522573743)

[4.1.1 Структура запроса 13](#_Toc522573744)

[4.1.2 Структура ответа 15](#_Toc522573745)

[4.1.3 Сервисы чтения и поиска данных 16](#_Toc522573746)

[4.1.4 Сервисы обеспечения качества данных 21](#_Toc522573747)

[4.1.5 Получение статистики 23](#_Toc522573748)

[4.1.6 Сервисы работы с операциями 24](#_Toc522573749)

[4.1.7 Зависимости SOAP 27](#_Toc522573750)

[4.2 Реализация сторонних функций 28](#_Toc522573751)

[4.2.1 Описание Java интерфейсов 28](#_Toc522573752)

[4.2.2 Регистрация функций в ПСКО 28](#_Toc522573753)

[4.3 Подключение сторонних пакетных задач 29](#_Toc522573754)

[4.3.1 Разработка сторонней задачи 29](#_Toc522573755)

[4.3.2 Упаковка и установка сторонней задачи 29](#_Toc522573756)

[4.3.3 Удаление или изменение сторонних задач 30](#_Toc522573757)

[4.3.4 Контроль ошибок 31](#_Toc522573758)

[5 Проверка корректности реализации интеграции 32](#_Toc522573759)

[6 Сообщения системному программисту 32](#_Toc522573760)

[Перечень сокращений и терминов 33](#_Toc522573761)

# Общие сведения о Подсистеме

## Наименование Подсистемы

Полное наименование подсистемы: Подсистема сопровождения кадастровой оценки ФГИС ЕГРН.

Условное обозначение подсистемы: ПСКО, Подсистема.

Условное обозначение Приложения №1 «Сопровождение ГКО» совместно с Приложением №2 «Мониторинг рынка недвижимости и надзор»: Приложения «ПСКО».

## Наименование Заказчика

Федеральная служба государственной регистрации кадастра и картографии (Росреестр).

## Наименование Исполнителя

Полное наименование: Закрытое акционерное общество «АМТ-ГРУП».

Сокращенное наименование: ЗАО «АМТ Груп».

## Цели, назначение и области использования Подсистемы

### Цели создания Подсистемы

ПСКО создается для достижения следующих целей:

* создание функционально полного механизма реализации роли Росреестра в системе проведения государственной кадастровой оценки (далее — ГКО);
* ведение достоверного ЕГРН в части сведений о кадастровой стоимости;
* обеспечение проведения мониторинга рынка недвижимости;
* повышение открытости и независимости процедур ГКО на каждом этапе ее осуществления;
* развитие контроля и проверяемости результатов ГКО.

### Назначение и области использования Подсистемы

Основным назначением ПСКО является обеспечение исполнения государственных функций, возложенных на Росреестр, в системе проведения ГКО объектов недвижимости на территории Российской Федерации, согласно Федеральному закону от 03.07.2016 № 237-ФЗ «О государственной кадастровой оценке».

ПСКО разрабатывается для автоматизации следующих основных групп деловых процессов, возникающих у Росреестра при проведении ГКО на территории Российской Федерации:

* сопровождение процедуры проведения ГКО по этапам и срокам;
* взаимодействие в электронном виде с Органами власти субъектов РФ в рамках проведения ГКО;
* взаимодействие с ГБУ субъектов РФ, наделенными полномочиями, связанными с определением кадастровой стоимости.

## Комплектность Подсистемы

Приложения «ПСКО» состоят из следующих модулей:

1. Модуль выполнения процедур с кадастровой стоимостью.
2. Модуль мониторинга рынка недвижимости.
3. Модуль управления деловыми процессами.
4. Модуль хранения данных.
5. Модуль управления и разграничения прав доступа.
6. Модуль логирования и аудита.
7. Модуль доставки уведомлений.
8. Модуль НСИ.
9. Модуль интеграции.

# Описание программных средств

## Общие сведения

Серверная часть функционирует под управлением операционной системы: CentOS Linux 7.4 64-bit. В качестве серверов приложений для развертывания приложений бизнес-логики и Web-интерфейса используются следующие приложения:

* Apache Tomcat;
* Camunda BPM Platform;
* Docker;
* Kubernetes;
* NodeJS.

Для административной части Подсистемы обеспечивается возможность работы в браузерах:

* Google Chrome версии 58 и выше;
* Mozilla Firefox версии 53 и выше.

При разработке использован язык программирования Java (версия 8).

## Функциональное назначение

Функциональное назначение Приложений «ПСКО» в зависимости от модуля представлено в таблице (см. Таблица 1):

Таблица 1 – Перечень функций

| **№** | **Название модуля** | **Назначение модуля** |
| --- | --- | --- |
|  | Модуль выполнения процедур с кадастровой стоимостью | 1. Выполнение пользователями действий по настроенным схемам деловых процессов, связанных с процедурами ГКО, государственного надзора, обработкой результатов оспаривания, обработкой результатов исправлений ошибок, расчетом кадастровой стоимости между предыдущей и последующей ГКО. 2. Отображение пользователям форм с обозначенными полями и элементами управления по каждому деловому процессу. 3. Отображение пользователям всей доступной информации ПСКО, связанной с деловыми процессами, связанными с кадастровой стоимостью. 4. Предоставление возможности ввода/скачивания данных. 5. Предоставление возможности редактирования текстов уведомлений до отправки. 6. Обеспечение сопровождающих проверок данных и полей обязательных к заполнению. |
|  | Модуль мониторинга рынка недвижимости | 1. Передача сведений о рынке недвижимости. 2. Формирование перечня ОН с пониженным индексом недвижимости. 3. Определение динамики индекса недвижимости. 4. Получение сведений о сделках с недвижимостью. 5. Расчет индекса недвижимости. 6. Обработка полученных сведений. |
|  | Модуль управления деловыми процессами | 1. Конфигурирование деловых процессов в визуальной форме. 2. Управление сроками (длительности) выполнения операций в отдельности и процессов в целом. 3. Назначение и конфигурирование процессов, в зависимости от роли пользователя. 4. Контроль ввода необходимых данных для осуществления перехода между шагами задачи. 5. Осуществление проверок корректности данных, вводимых на шагах процесса. 6. Отслеживание состояния процесса и прохождения по нему пользователя. |
|  | Модуль хранения данных | 1. Обеспечение хранения, архивирования всех используемых ПСКО документов. 2. Обеспечение хранения реестров и журналов, используемых в ПСКО. 3. Обеспечение хранения версий, используемых ПСКО документов. |
|  | Модуль управления и разграничения прав доступа | 1. Разделение полномочий пользователей. 2. Разграничение прав доступа к объектам ПСКО. |
|  | Модуль логирования и аудита | 1. Логирование действий пользователей всех ролей ПСКО. 2. Логирование всех попыток входа, удачных и неудачных. 3. Сохранение ошибок как ответа удаленных сервисов, так и произошедших в подсистеме. 4. Сохранение попыток обращения к удаленным сервисам и вызова процедур, статусы ответов, объем полученных данных. 5. Профилирование скорости выполнения действий в подсистеме для ее компонентов и процедур. |
|  | Модуль доставки уведомлений | Формирование уведомлений по произошедшим изменениям в ПСКО с помощью сформированных в конструкторе уведомлений шаблонов о:   * выполненных шагах процесса; * возникших проблемах; * назначенных задачах; * изменениях в документах, требующих внимания; * передаче соответствующих уведомлений Органам власти субъектов РФ и ГБУ субъектов РФ; * осуществлении уведомления Диспетчеров и Специалистов по надзору, о поступивших в ПСКО данных. |
|  | Модуль НСИ | 1. Моделирование структуры справочников, реестров и их взаимосвязей. 2. Указание ссылок и зависимостей. 3. Управление периодами актуальности изменений в данных. 4. Операции поиска дубликатов и возможности по их объединению. 5. Конфигурирование модели качества данных. 6. Конфигурирование правил проверки корректности данных, задание правил качества на сохранении и обновлении. 7. Обогащение данных при вводе. 8. Задачи по исправлению неточностей. 9. Интеграция с ЕС УНСИ с целью получения справочников единых для подсистем ФГИС ЕГРН. 10. Реализация периодических задач. 11. Поиск дубликатов и проблем в данных. 12. Задача обновления данных из ЕС УНСИ, выполняемой по расписанию. |
|  | Модуль интеграции | 1. Выполнение пользователями действий по настроенным схемам деловых процессов, связанных с обработкой сведений о рынке недвижимости от ГБУ; 2. Выполнение пользователями действий расчета индекса недвижимости по настроенным формулам; 3. Выполнение анализа полученного результата с ранее рассчитанным индексом рынка недвижимости. |

## Используемые технические средства

Техническая архитектура Подсистемы состоит из клиентской и серверной составляющих.

Серверная составляющая представляет собой совокупность функциональных серверов (сервер баз данных, сервер приложений, веб-сервер), объединенных решением, обеспечивающим обмен данными между функциональными серверами и взаимодействие Подсистемы с внешними информационными системами.

Клиентская составляющая, предоставляющая графический пользовательский интерфейс, обеспечивает возможность доступа к ресурсам и функциям Подсистемы с автоматизированными рабочими местами (АРМ) пользователей и администраторов без установки дополнительного программного обеспечения, вне зависимости от используемой операционной системы, при условии наличия возможности запуска одного из следующих Интернет-обозревателей (веб-браузеров):

* Google Chrome версии 58 и выше;
* Mozilla Firefox версии 53 и выше.

Программные средства, используемые в ПСКО:

* базовые компоненты:
* базовые хосты (виртуальные машины);
* операционная система: CentOS 7.4;
* инструмент контейнеризации и управления развертыванием: Docker версии 1.13.1 и выше;
* платформа оркестрации Docker контейнеров Kubernates;
* система сборки Gradle;
* многофункциональный плагин gradle для запуска веб-приложений на встроенных контейнерах сервлетов Gretty-gradle-plugin;
* интерфейс коммуникации frontend и backend-частей приложения – GraphQL.
* общие компоненты и технологии Backend разработки:
* резервное копирование: библиотека логгирования logback;
* отправка сообщений в logback, без сохранения их в файлы или другие сетевые промежуточные ресурсы logstash-logback-encoder;
* библиотека MyBatis ORM;
* инструмент миграции базы данных FlyWay;
* универсальный фреймворк для Java-платформы Spring;
* инструмент Spring boot для настройки фреймворка Spring;
* spock framework тестирования для приложений Java и Groovy;
* платформа для управления рабочими процессами и бизнес-процессами Camunda BPM.
* общие компоненты и технологии Frontend разработки:
* библиотека JavaScript для создания пользовательских интерфейсов React;
* контейнер состояний Redux;
* система сборки и зависимостей Webpack;
* менеджер пакетов для JavaScript npm;
* система сборки npm.
* управление конфигурацией, обнаружение сервисов:
* обнаружение и настройка служб Consul.io;
* инструмент для развертывания Ansible.
* шина взаимодействия Pub/Sub:
* брокер сообщений Apache Kafka;
* протоколирование событий:
* нереляционное хранилище (БД) Elasticsearch;
* сервис, обеспечивающий прием сообщений по сети от приложений их разбор, фильтрацию и сохранение Logstash;
* анализ поступивших логов и журналов Kibana.
* БД:
* хранилище пользовательских данных, процессов, решений, статусов регистрационных действий Postgres;
* хранения входящих документов и ответов внешних систем Couchbase server;
* сервер хранения объектов с API-интерфейсом Amazon S3-minio.
* отчеты и статистика:
* инструмент отображения статистики, отчетов Kibana embedded.

# Структура интеграционных компонентов Подсистемы и способы реализации интеграции

## Структура интеграционных компонентов

### Программные клиенты и сервисы

Программные клиенты и сервисы представляют собой специализированный набор логической архитектуры ПСКО, создаваемый с учетом специфики решаемой задачи. Основным его назначением является реализация нестандартной логики обработки данных.

Конечный состав программных клиентов и сервисов определяется конечной целью их применения.

Программные клиенты представляют собой приложения, разрабатываемые под конкретную задачу и поддерживающие возможность получения данных по протоколу SOAP. Интеграция с подобными приложениями производится через публичный API, реализованный на стороне ПСКО.

### Специализированные обработчики событий

Специализированные обработчики событий (или user-exits) представляют собой набор классов на языке программирования Java, с помощью которых реализуются публичные интерфейсы ПСКО. Вызов обработчиков может осуществляться при наступлении определенного события, либо в определенное время.

Например, в случаях, когда для выполнения операции создания записи необходима отправка особого уведомления, содержащего аутентификационные данные учетной записи для последующей авторизации в сторонней информационной системе.

## API ПСКО

Взаимодействие с публичным API ПСКО осуществляется по протоколу SOAP. Доступные версии API расположены по адресу: https://{host}:{port}/unidata-backend/api/, где host – это доменное имя или IP-адрес сервера, а port – порт для доступа к серверу, на котором исполняется Tomcat с unidata-backend приложением по протоколу HTTP. В случае использования кластерной конфигурации, значения host и port должны содержать IP-адрес и порт для доступа к серверу балансировки нагрузки.

## Сторонние функции

Сторонние функции предоставляют возможность настройки правил качества данных (например, для обогащения и валидации). Применение сторонних функций позволяет расширить существующий набор доступных операций без внесения изменений в кодовую базу ПСКО.

Использование сторонних функций осуществляется через публичный Java-интерфейс, предоставляемый ПСКО.

## Точки расширения

Точки расширения применяются при необходимости выполнения каких-либо нестандартных действий, выходящих за рамки функционала системы. Например, если для выполнения операции вставки необходима отправка особого уведомления, содержащего аутентификационные данные для последующей авторизации в сторонней системе.

Точки расширения представляют собой программный код на языке Java, выполняемый при наступлении определенного события. Внедрение кода осуществляется посредством расширения одного или более интерфейсов из пакетов:

* com.unidata.mdm.integration.exits (пакет интерфейсов для настройки расширенной обработки событий удаления, слияния и вставки);
* com.unidata.mdm.integration.auth (пакет интерфейсов для настройки интеграции с внешними системами разграничения прав доступа).

Точки расширения имеют отдельную конфигурацию, которая считывается при запуске сервера приложений и расположена в файле PSKO-conf.xml.

Подробное описание структуры конфигурационной ПСКО приведено в документе АМСЯ.РРПСКО.И5.2-01 «Руководство администратора».

## Сторонние пакетные задачи

ПСКО предоставляет возможность реализации сторонних пакетных задач, их подключение и централизованное управление запуском пользователями с ролью Администратор с помощью графического интерфейса.

# Настройка интеграции

## Программные интерфейсы (SOAP API)

### Структура запроса

Общая структура запроса приведена в таблице ниже (Таблица 2).

Запрос по протоколу SOAP состоит из:

* обязательной секции 'common', содержащей стандартный набор общих параметров, таких как идентификатор хранилища, безопасность, и т.д.;
* одного из конкретных запросов.
* Общая часть любого API запроса определяется двумя основными параметрами:
* параметры безопасности;
* параметры асинхронного вызова.

Помимо этого, каждый содержит обязательный параметр 'storageId', идентифицирующий логическое хранилище.

Предусмотрена возможность использования идентификатора логической операции в качестве дополнительного параметра. Идентификатор логических операций используется для аудита, а также для контроля композитных операций.

Идентификатор логической операции не является обязательным, если он не указан, то ПСКО автоматически сгенерирует значение.

Таблица 2 – Общая структура запроса по протоколу SOAP

| Наименование поля | Тип значения поля |
| --- | --- |
| Security | credentials (username, password) |
| sessionToken | SessionTokenDef |
| credentials | CredentialsDef |
| asyncOptions | userJMS |
| jmsReplyTo |
| jmsCorrelationId |
| storageId | строковый |
| operationId | строковый |

Структура общей части запроса описана в таблице (см. Таблица 3).

Таблица 3 – Структура общей части запроса (CommonSectionDef)

| Поле | Тип | Описание |
| --- | --- | --- |
| security | SecuritySectionDef, обязательный | Данные для аутентификации пользователя |
| asyncOptions | AsyncSectionDef | Параметры асинхронного вызова |
| storageId | строковый, обязательный | Идентификатор логического хранилища |
| operationId | строковый | Идентификатор логической операции |
| version | строковый | Версия API клиента |

Общие части запроса описаны в Таблицах (Таблица 4 – Таблица 7).

Секция 'security'является общей частью любого API запроса. Секция содержит данные для аутентификации пользователя. Имеется возможность работать в двух режимах. В первом режиме необходимо всегда передавать имя пользователя и пароль (элемент 'credentials'). При этом серверная сторона будет всегда создавать новую сессию со всеми вытекающими издержками.

Второй способ состоит в передаче сессионного ключа, который может быть получен после успешного исполнения сервиса RequestAuthenticate в режиме doLogin='true'. Данный подход рекомендован, если вызывающая сторона заведомо знает о необходимости выполнить последовательность API запросов.

Таблица 4 – Общая часть запроса. Аутентификация (SecuritySectionDef)

| Поле | Тип | Описание |
| --- | --- | --- |
| сredentials | CredentialsDef | Логин и пароль пользователя |
| sessionToken | SessionTokenDef | Сессионный ключ |

Сессионный ключ представляет собой произвольный набор символов. Используется для передачи ключа сессии между запросами. В типовом сценарии использования ключ возвращается в ответ на запрос 'RequestAuthenticate', затем передается во все последующие запросы.

Таблица 5 – Общая часть запроса. Токен сессии (SessionTokenDef)

| Поле | Тип | Описание |
| --- | --- | --- |
| token | строковый, обязательный | Сессионный ключ |

Таблица 6 – Общая часть запроса. Учетная запись (CredentialsDef)

| Поле | Тип | Описание |
| --- | --- | --- |
| Обязательные данные  Должен присутствовать логин/пароль: | | |
| username | строковый | Логин пользователя |
| password | строковый | Пароль пользователя |
| Или токен: | | |
| externalToken | строковый | Токен для аутентификации из сторонней системы |

Таблица 7 – Общая часть запроса. Параметры асинхронного вызова (AsyncSectionDef)

| Поле | Тип | Описание |
| --- | --- | --- |
| useJMS | логический | Признак доставки ответа через JMS |
| jmsReplyTo | строковый | JNDI имя целевой очереди JMS для публикации результата исполнения |
| jmsCorrelationId | строковый | Идентификатор запроса, для идентификации данного запроса в очереди JMS |

### Структура ответа

Общая часть любого API ответа (Таблица 8):

* содержит идентификатор логической операции, либо изначально переданный в запросе, либо авто-сгенерированный ПСКО;
* содержит 'exitCode', используемый вызывающей стороной для анализа ошибок;
* может содержать несколько элементов с сообщениями. В случае 'exitCode' отличного от 'Success', будет содержать как минимум один элемент.
* Структура любого сообщение возвращаемого вместе с ответом на запрос:
* содержит текст сообщения в поле 'messageText';
* может содержать структуры, содержащие описание ошибки, включая детальные сообщения от сервера.

Таблица 8 – Общая структура ответа

| Наименование поля | Тип значения поля |
| --- | --- |
| message | stackTrace |
| error (errorCode, userMessage, InternalMessage) |
| messageText |
| operationId | строковый |
| exitCode | ExitCodeType |
| processingTime | целочисленный |

Структура ответов описана в Таблицах ниже (Таблица 9 – Таблица 10).

Таблица 9 – Структура CommonResponseDef

| Поле | Тип | Описание |
| --- | --- | --- |
| message | SecuritySectionDef, обязательный | Опциональные сообщения, переданные вместе с ответом. В случае ошибок или предупреждений, может содержать текст для отображения пользователю, а также технические детали |
| operationId | строковый, обязательный | Идентификатор логической операции, либо изначально переданный в запросе, либо сгенерированный ПСКО |
| exitCode | ExitCodeType | Результирующий код исполнения запроса |
| processingTime | целочисленный | Время исполнения запроса на стороне сервера в миллисекундах |

Таблица 10 – Структура ExecutionMessageDef

| Поле | Тип | Описание |
| --- | --- | --- |
| stackTrace | строковый | Необязательный элемент, содержащий Java stack-trace. Заполняется для службы поддержки |
| error | секция | Необязательный элемент, содержащий детальное описание ошибки |
| messageText | строковый, обязательный | Текст сообщения |

### Сервисы чтения и поиска данных

#### Аутентификация пользователя

Общая часть запроса RequestAuthenticate должна содержать данные для аутентификации пользователя (элемент 'common' из структуры '*U*nidataRequestBody').

Доступны следующие способы аутентификации пользователей:

* с использованием логина и пароля пользователя (элемент 'credentials' общей секции);
* с использованием сессионного ключа, при условии активности текущей сессии.

Во втором случае сервер вернет ответ, соответствующий вводу корректной комбинации логина и пароля, с последующим сбросом таймера активности сессии.

По умолчанию сервер всегда создает новую сессию. Для проверки логина и пароля пользователя, а также для получения списка ролей, присвоенных пользователю, используется режим doLogin='false'. При этом новая сессия не создается.

В случае успешного прохождения процедуры аутентификации ответ сервера будет содержать список ролей, присвоенных пользователю. При создании новой сессии ответ будет содержать сессионный ключ, который будет использоваться для ответа на последующие запросы в рамках одной сессии.

Структура запроса описана в таблице ниже (Таблица 11).

Таблица 11 – Структура запроса

| Поле | Тип | Описание |
| --- | --- | --- |
| doLogin | логический | Определяет необходимость создания новой сессии. Значение по умолчанию: 'true' |

Общая часть ответа ResponseAuthentificate содержит код возврата, идентификатор логической операции и сообщения об ошибках (элемент 'common' из структуры 'UnidataResponseBody'). Структура ответа описана в таблице ниже (Таблица 12).

Таблица 12 – Структура ответа

| Поле | Тип | Описание |
| --- | --- | --- |
| sessionToken | SessionTokenDef | Опциональное значение сессионного ключа |
| role | RoleRefDef | Роль пользователя |
| isAdmin | логический,  обязательный | Признак наличия прав администратора для данной учетной записи |

Структура SessionTokenDef описывает сессионный ключ и представляет собой произвольный набор символов (Таблица 13). Структура используется для передачи ключа сессии между запросами.

В типовом сценарии использования, данный ключ возвращается в качестве ответа на запрос 'RequestAuthenticate' и используется для ответа на все дальнейшие запросы в рамках одной сессии.

Таблица 13 – Структура SessionTokenDef

| Поле | Тип | Описание |
| --- | --- | --- |
| token | строковый, обязательный | Опциональное значение сессионного ключа |

Структура RoleRefDef описывает роль, присвоенную пользователю   
(Таблица 14).

Таблица 14 – Структура RoleRefDef

| Поле | Тип | Описание |
| --- | --- | --- |
| name | строковый,обязательный | имя роли |

#### Поиск основных записей сущности

Общая часть запроса RequestSearch должна содержать данные для аутентификации пользователя (элемент 'common' из структуры 'UnidataRequestBody').

Запрос должен содержать условия поиска, параметры постраничного вывода результатов и сортировки результатов.

Далее указана структура запросов поиска и сортировки (Таблица 15 – Таблица 17).

Таблица 15 – Структура запроса вызова поиска и сортировки

| Поле | Тип | Описание |
| --- | --- | --- |
| searchCondition | SearchConditionDef | Критерий, на основе которого осуществляется поиск |
| sortCondition | SortConditionDef | Секция, описывающая порядок сортировки по полям |
| entityName | строковый, обязательный | Имя сущности, для которой осуществляется поиск |
| returnCount | логический | Параметр, который указывает, возвращать ли общее количество записей, соответствующих критерию поиска |
| doCountOnly | логический | При установке параметра сформированный ответ будет содержать число, отображающее общее количество записей, соответствующих заданному критерию поиска (без предоставления самих записей) |
| pageSize | целочисленный | Размер страницы. Значение данного параметра определяет количество сущностей, составляющих одну страницу поисковой выдачи |
| pageNumber | целочисленный | Значение параметра определяет номер страницы поисковой выдачи. Сформированный ответ будет содержать релевантные записи с текущей страницы и со всех последующих страниц поисковой выдачи, с одновременным отсутствием записей из предыдущих страниц |

Таблица 16 – Структура поискового запроса SearchConditionDef

| Поле | Тип | Описание |
| --- | --- | --- |
| SearchConditionDef | | |
| expression | SearchAtomDef  ИЛИ  SearchOrDef  ИЛИ  SearchAndDef | Выбор одного из алгоритмов поиска:   * поиск на конкретный атрибут сущности * поиск с логическим «ИЛИ» * поиск с логическим «И» |
| SearchAtomDef | | |
| attributeName | строковый | Обязательное имя атрибута.  Введенный поисковой запрос |
| operator | перечисление | Обязательный оператор сравнения.  Принимает значения:  EQUALS – равен значению атрибута поиска (attributeName)  NOT\_EQUALS – не равен значению поиска  GREATER – выше значения поиска  GREATER\_OR\_EQUALS – выше или равно значению поиска  LESS – меньше значения поиска  LESS\_OR\_EQUALS – меньше или равно  LIKE – схожее значение (частичное совпадение)  FUZZY\_EQUALS – нечеткое равенство (по сходству) |
| constant | строковый | Обязательное значение константы.  Значения в БД, среди которых производится поиск |
| SearchOrDef | | |
| and | SearchAndDef | Поиск с использованием логического оператора «И» |
| Поле | Тип | Описание |
| or | SearchOrDef | Поиск с использованием логического оператора «ИЛИ» |
| atom | SearchAtomDef | Поиск на конкретный атрибут сущности |
| SearchAndDef | | |
| and | SearchAndDef | Поиск с использованием логического оператора «И» |
| or | SearchOrDef | Поиск с использованием логического оператора «ИЛИ» |
| atom | SearchAtomDef | Поиск на конкретный атрибут сущности |

Таблица 17 – Структура запроса сортировки SortConditionDef

| Поле | Тип | Описание |
| --- | --- | --- |
| field | строковый | Поле, по которому будет осуществляться сортировка |
| type | перечисление | Направление сортировки. Принимает одно из значений:  ASC – сортировка по возрастанию  DESC – сортировка по убыванию |

Общая часть ответа ResponseSearch (Таблица 18) содержит код возврата, идентификатор логической операции и сообщения об ошибках (элемент 'common' из структуры 'UnidataResponseBody').

Таблица 18 – Структура ответа

| Поле | Тип | Описание |
| --- | --- | --- |
| etalonRecord | data:EtalonRecord | Основные записи, соответствующие критериям поискового запроса |
| pageSize | целочисленный обязательный | Размер страницы (количество сущностей, составляющих одну страницу поисковой выдачи) |
| pageNumber | целочисленный, обязательный | Номер страницы поисковой выдачи с релевантными записями. Сформированный ответ содержит релевантные записи с данной страницы, а также всех последующих страниц поисковой выдачи, с одновременным отсутствием записей из предыдущих страниц |
| count | целочисленный | Число, обозначающее общее количество записей, подпадающих по критерий поиска |

#### Получение списка ошибок основной сущности

Общая часть запроса RequestGetDataQualityErrors (Таблица 19) должна содержать данные для аутентификации пользователя (элемент 'common' из структуры 'UnidataRequestBody').

Запрос должен содержать имя сущности, ключ идентифицирующий основную, либо исходную запись, а также, необязательный параметр 'forDate', при указании которого сформированный ответ будет содержать список ошибок сущности по состоянию на соответствующую дату. В противном случае ответ будет содержать список ошибок сущности по состоянию на текущую дату.

Таблица 19 – Структура ответа

| Поле | Тип | Описание |
| --- | --- | --- |
| etalonKey | EtalonKey | Значение ключа, идентифицирующего основную запись |
| originKey | OriginKey | Значение ключа, идентифицирующего исходную запись |
| entityName | строковый, обязательный | Имя сущности |
| forDate | dateTime | Значение параметра определяет дату для получения списка ошибок сущности |

Ответ на запрос на получение списка ошибок основной сущности, возникших после применения правил контроля качества данных ('RequestGetDataQualityErrors').

Общая часть ответа ResponseGetDataQualityErrors содержит код возврата, идентификатор логической операции и сообщения об ошибках (элемент 'common' из структуры 'UnidataResponseBody').

Ответ содержит список ошибок качества данных (Таблица 20).

Таблица 20 – Структура ответа

| Поле | Тип | Описание |
| --- | --- | --- |
| dqError | DataQualityError | Ошибка, возникшая в результате применения правил качества данных |

### Сервисы обеспечения качества данных

#### Применение правил качества данных

ПСКО поддерживает возможность применения правил качества к записям, полученным из сторонней информационной системы, без сохранения исправленных записей в собственную базу данных.

Данное действие выполняется в двух режимах: ONLINE и BATCH. При выборе BATCH производится сохранение информации о найденных ошибках в БД ПСКО с возможностью последующего получения информации по запросу (п. 4.1.4.2).

Доступ к сервисам обеспечения качества данных осуществляется по адресу /unidata-backend/api/public/dq/v4/DataQualityService.

Общая часть запроса должна содержать данные для аутентификации пользователя (элемент 'securityHeader'), а также идентификатор запроса и имя реестра/справочника, для которого применяются правила (элемент 'infoHeader').

Запрос applyDQRequest (Таблица 21) должен содержать информацию о выбранном режиме запуска (info.mode), имя одной или нескольких информационных систем-источников, для которых должны применяться правила, список применяемых правил, а также перечень записей выбранного реестра/справочника. Также должен присутствовать параметр, определяющий применение правил к исходным или эталонным записям (info.applicable).

Таблица 21 – Структура запроса

| Поле | Тип | Описание |
| --- | --- | --- |
| info.mode | строковый, обязательный | Режим запуска.  Возможные значения:  ONLINE и BATCH |
| info.sourceSystem | строковый, обязательный | Список систем-источников, для которых должны применяться правила качества  Значение ALL соответствует выбору всех систем-источников |
| info.applicable | строковый, обязательный | Параметр, определяющий применение правил к исходным, либо к эталонным записям.  Возможные значения: ETALON (эталонные записи) и ORIGIN (исходные записи) |
| info.ruleName | строковый | Список применяемых правил. Если значение не указано, то применяются существующие правила |
| payload | элемент | Список записей выбранного реестра/справочника. Может содержать от 1 до 500 записей. Поле ID является обязательным |

Общая часть ответа содержит данные для аутентификации пользователя (элемент 'securityHeader'), а также идентификатор запроса и имя реестра/справочника, для которого применяются правила (элемент 'infoHeader').

Ответ applyDQResponse содержит имя сущности и идентификатор запроса (элемент 'info'), а также идентификатор записи, статус (содержит/не содержит ошибки) и перечень ошибок с указанием имени правила, а также категории и уровня критичности ошибки (элемент 'payload').

Таблица 22 – Структура ответа

| Поле | Тип | Описание |
| --- | --- | --- |
| id | строковый | Идентификатор записи |
| status | строковый | Статус по результатам проверки (содержит/не содержит ошибки). |
| errors | строковый | Информация о найденных ошибках, где:  ruleName – имя правила, по которому выявлена ошибка;  message – сообщение об ошибке;  severity – уровень критичности ошибки;  category – категория ошибки |

#### Получение информации о найденных ошибках

Фиксация информации о найденных ошибках производится при применении правил качества данных в режиме BATCH (п. 4.1.4.1). Если данное действие ранее никогда не производилось, получить указанную информацию невозможно ввиду ее отсутствия в БД ПСКО.

Доступ к сервисам обеспечения качества данных осуществляется по адресу https://{host}:{port}/unidata-backend/api/public/dq/v4/DataQualityService.

Общая часть запроса должна содержать данные для аутентификации пользователя (элемент 'securityHeader'), а также идентификатор запроса (элемент 'infoHeader').

Запрос getResultsRequest (Таблица 23) должен содержать идентификатор записи.

Таблица 23 – Структура запроса

| Поле | Тип | Описание |
| --- | --- | --- |
| id | строковый, обязательный | Идентификатор записи |

Общая часть ответа содержит данные для аутентификации пользователя (элемент 'securityHeader'), а также идентификатор запроса (элемент 'infoHeader').

Ответ getResultsResponse содержит идентификатор ошибки, дату создания, статус, имя правила, по которому была выявлена данная ошибка, а также информацию о е категории и уровне критичности (Таблица 24).

Таблица 24 – Структура ответа

| Поле | Тип | Описание |
| --- | --- | --- |
| errors | строковый | Информация о найденных ошибках, где:  errorId – идентификатор ошибки;  createDate – дата фиксации ошибки;  ruleName – имя правила, по которому выявлена ошибка;  severity – уровень критичности ошибки;  category – категория ошибки |

### Получение статистики

#### Структура запроса

Общая часть запроса RequestInfoGet на получении статистики данных должна содержать данные для аутентификации пользователя (элемент 'common' из структуры 'UnidataRequestBody'). Описание структуры запроса приведено в таблице ниже (Таблица 25).

Все элементы являются обязательными.

Таблица 25 – Структура запроса RequestInfoGet

| Поле | Тип | Описание |
| --- | --- | --- |
| from[1] | dateTime, обязательный | Левая граница периода |
| to[1] | dateTime, обязательный | Правая граница периода |
| entityName[1] | строковый, обязательный | Имя сущности |

#### Структура ответа

Общая часть ответа ResponseInfoGet содержит код возврата, идентификатор логической операции и сообщения об ошибках (элемент 'common' из структуры 'UnidataResponseBody'). В случае успешного исполнения запроса, в ответе будет указан массив объектов, содержащих статистические данные. Описание структуры ответа приведено в Таблицах ниже (Таблица 26 – Таблица 28).

Таблица 26 – Структура ResponseInfoGet

| Поле | Тип | Описание |
| --- | --- | --- |
| statistic [0..n] | tdapi:Statistic | Массив объектов. Содержит статистические данные |

Таблица 27 – Структура tdapi:Statistic

| Поле | Тип | Описание |
| --- | --- | --- |
| type [1] | tdapi:StatisticEnum | Возможные значения: NEW, ERRORS, UPDATED, TOTAL, MERGED, DUPLICATES |
| series [1..n] | tdapi:TimeSerie | Массив объектов |

Таблица 28 – Структура tdapi: TimeSerie

| Поле | Тип | Описание |
| --- | --- | --- |
| time[1] | dateTime | Дата снятия среза статистики |
| value[1] | int | Количество записей |

### Сервисы работы с операциями

Валидация запросов сервера отключена. Необходимо проверять правильность сформированных запросов при работе с SOAP согласно XSD. Ошибка запроса определяется ответом формата <soap:Fault></soap:Fault>.

#### Поиск всех доступных операций

Запрос requestFindAllJobs должен содержать пустое значение (элемент *'common'* из структуры *'UnidataRequestBody'*).

Ответ requestFindAllJobs должен содержать список найденных операций с параметрами. Описание параметров для каждой из операций приведено в таблице ниже (Таблица 29).

Таблица 29 – Параметры операции

| Поле | Тип | Описание |
| --- | --- | --- |
| id | long | Идентификатор операции |
| enabled | boolean | Значение, активна ли операция или нет |
| error | boolean | Значение, имеет ли операция ошибки.  Со стороны пользователя поле доступно только для чтения |
| cronExpression | string | Выражения CRON для задания времени запуска операции |
| description | string | Описание операции |
| skipCronWarnings | boolean | Игнорировать ли предупреждения CRON |
| name | string | Имя операции |
| jobNameReference | string | Внутреннее имя batch задачи |
| parameters | JobParameter  (Таблица 30) | Список параметров |

Таблица 30 – Описание параметров, входящих в список параметров операций (parameters таблицы «Параметры операции»)

| Поле | Тип | Описание |
| --- | --- | --- |
| id | long | Идентификатор параметра |
| type | string, long, double, date, boolean | Тип параметра. Задается тип значения параметра из вариантов: string, long, double, date или boolean |
| value | string, long, double, date, boolean | Значение параметра (соответствует заданному типу параметра) |

#### Сохранение операции

Запрос requestSaveJob должен содержать параметры операции для сохранения (элемент *'common'* из структуры *'UnidataRequestBody'*). Параметры сохраняемой операции указаны в таблице выше (Таблица 29).

Ответ requestSaveJob должен содержать параметры сохраненной операции и добавленный id операции. Параметры сохраненной операции указаны в таблице выше (Таблица 29).

#### Удаление операции

Запрос requestRemoveJob должен содержать идентификатор удаляемой операции (элемент *'common'* из структуры *'UnidataRequestBody'*). Описание параметра указано в таблице ниже (Таблица 31).

Таблица 31 – Идентификатор операции (параметр операции)

| Поле | Тип | Описание |
| --- | --- | --- |
| id | long | Идентификатор операции |

Ответ requestRemoveJob должен содержать пустое значение.

#### Поиск операции по ID

Запрос requestFindJob должен содержать идентификатор искомой операции (элемент *'common'* из структуры *'UnidataRequestBody'*). Описание параметра указано в таблице (Таблица 31).

Ответ requestFindJob должен содержать найденную по id операцию с параметрами. Описание параметров операции указано в таблице (Таблица 29).

#### Запуск операции

Запрос requestRunJob должен содержать идентификатор запускаемой операции и список параметров запуска (элемент *'common'* из структуры *'UnidataRequestBody'*).

Описание параметра id указано в таблице (Таблица 31).

Описание параметров запуска операции указано в таблице (Таблица 30).

Ответ requestRunJob должен содержать результаты исполнения операции. Описание параметров указано в таблице ниже (Таблица 32).

Таблица 32 – Параметры информации о выполнении операции

| Поле | Тип | Описание |
| --- | --- | --- |
| job | job  (Таблица 29) | Информация, по которой получена информация |
| parameters | JobParameter  (Таблица 30) | Список параметров |
| exitStatus | JobExecutionExitStatus  (Таблица 33) | Статус завершения выполнения операции |
| status | enum of strings | Статус выполнения. Возможные значения: COMPLETED, STARTING, STARTED, STOPPING, STOPPED, FAILED, ABANDONED, UNKNOWN |
| startTime | dataTime | Дата, время старта операции |
| createTime | dataTime | Дата, время создания операции |
| endTime | dataTime | Дата, время завершения операции |
| lastUpdated | dataTime | Дата, время последнего обновления операции |

Таблица 33 – Параметры JobExecutionExitStatus

| Поле | Тип | Описание |
| --- | --- | --- |
| code | string | Код завершения операции |
| description | string | Описание статуса операции |

#### Получение статуса последнего выполнения операции

Запрос requestJobStatus должен содержать идентификатор операции (элемент *'common'* из структуры *'UnidataRequestBody'*). Описание параметра указано в таблице (Таблица 31).

Ответ requestJobStatus должен содержать информацию о статусе операции. Описание параметров указано в таблице (Таблица 33).

### Зависимости SOAP

Для корректной работы soap-api-r4.7.jar необходимы зависимости:

* com.unidata.mdm:common:r4.7;
* org.apache.cxf.xjc-utils:cxf-xjc-runtime:3.0.5;
* org.apache.cxf: cxf-rt-bindings-soap: 3.1.0;
* commons-lang:commons-lang:2.6;
* javax.xml.ws:jaxws-api:2.1.

## Реализация сторонних функций

### Описание Java интерфейсов

Для использования сторонней функции необходимо реализовать два метода интерфейса com.*u*nidata.mdm.cleanse.CleanseFunction:

getDefinition. Возвращает объект типа com.*u*nidata.mdm.meta.CleanseFunctionExtendedDef*,* содержащий информацию о функции:

* описание входящих портов;
* описание исходящих портов;
* название и другую информацию о функции.

execute. содержит реализацию самой логики функции.

метод принимает на вход ассоциативный массив Map<String, Object>, где первый аргумент – это имя входного параметра, а второй это его значение,

метод возвращает на выход ассоциативный массив Map<String, Object>, где первый аргумент – это имя выходного параметра, а второй – его значение.

### Регистрация функций в ПСКО

Для регистрации функции в ПСКО необходимо загрузить собранный jar-файл при помощи интерфейса администратора. Для этого:

* перейдите в раздел «Функции»;
* нажмите «Добавить новую функцию»;
* выберите «Сторонняя»;
* в появившемся окне, укажите место расположения jar-файла, содержащего требуемую стороннюю функцию, и загрузите его в ПСКО.

После загрузки выбранная функция отобразится в списке доступных функций.

Для проверки работоспособности загруженной функции нажмите на ее имя в списке доступных функций, после чего на экране отобразится соответствующий графический интерфейс.

При отсутствии результата выполнения функции необходимо провести повторную проверку содержимого файла функции, исправить обнаруженные ошибки и произвести повторную загрузку. В случае отсутствия ошибок в файле функции обратитесь в службу технической поддержки ПСКО.

## Подключение сторонних пакетных задач

### Разработка сторонней задачи

Сторонние задачи разрабатываются с помощью фреймворка Spring Batch 3.6. Задача описывается XML конфигурацией и классами реализации (в Spring Batch уже содержится большинство готовых классов).

Информацию по описанию и настройке задач можно найти по ссылке: <http://docs.spring.io/spring-batch/reference/html/>.

### Упаковка и установка сторонней задачи

#### Конфигурирование XML

XML-конфигурация и классы сторонней задачи должны быть сохранены в jar-файле.

Загрузчик XML конфигураций сторонних задач в jar-файле выполняет поиск по шаблону: com/PSKO/mdm/integration/job/\*\*/\*-job.xml. То есть XML конфигурация сторонней задачи должна иметь суффикс «-job.xml» и находится внутри каталога: com/PSKO/mdm/integration/job/.

Особенности реализации сторонних задач:

* для сторонних задач в XML конфигурации необходимо указать ссылку на диспетчера транзакций (Transaction manager) и присвоить значение идентификатора: id="jobTransactionManager". Диспетчер транзакций используется для фиксации информации по работе сторонних задач в БД.
* каждая сторонняя задача при старте создает дочерний контекст с родительским контекстом. Таким образом, все классы родительского контекста будут доступны дочернему контексту сторонней задачи.
* для каждой задачи можно определить список параметров, доступных в форме создания задачи. Для этого необходимо указать класс следующего типа в XML конфигурации:

class="com.unidata.mdm.backend.service.job.registry.JobTemplateParameters"

* укажите название задачи, для которой будут использоваться заданные параметры, в свойствах требуемого класса.
* для каждой сторонней задачи, требующей распределенной работы с использованием Hazelcast, необходимо добавить класс в соответствующую XML-конфигурацию:

<bean id="myJobStepLocator" class="org.springframework.batch.integration.partition.BeanFactoryStepLocator"/>

Каждая сторонняя задача должна иметь уникальный идентификатор «id».

#### Установка

Скопируйте в каталог <TOMCAT\_HOME>/PSKO-integration/ JAR-файл, содержащий XML конфигурацию и классы реализации, затем перезапустите ПСКО.

При использовании кластерной конфигурации файл необходимо скопировать в указанный каталог для каждого узла кластера.

Загрузка сторонних задач производится автоматически во время запуска приложения. Все загруженные задачи доступны через интерфейс администратора. При создании новой задачи необходимо привязать ее к сторонней задаче, для этого необходимо перейти в форму создания задачи, затем выбрать стороннюю задачу в выпадающем списке поля «Тип». Запуск сторонней задачи осуществляется с помощью интерфейса администратора.

### Удаление или изменение сторонних задач

Существуют следующие сценарии удаления и изменения сторонних задач:

Если JAR-файл сторонней задачи был удален из каталога <TOMCAT\_HOME>/PSKO-integration/ на одном из узлов кластера, все задачи, конфигурации которых содержатся в указанном файле, будут иметь соответствующую маркировку в интерфейсе администратора. Данные задачи недоступны для запуска. При этом, если в параметрах задачи был установлен флаг активности, то он будет снят.

Если JAR-файл сторонней задачи был изменен в каталоге <TOMCAT\_HOME>/PSKO-integration/ на любом из узлов кластера, при этом список параметров задач, созданных через интерфейс, не совпадает со списком параметров, указанных в XML-конфигурации сторонней задачи, указанные задачи не будут доступны для запуска в интерфейсе администратора. При восстановлении соответствия набора параметров в указанных списках восстановится возможность запуска данных задач.

#### Диагностика неисправностей

Сторонние задачи не отображаются в выпадающем списке поля «Тип» в форме создания задачи

Если сторонние задачи не отображаются в выпадающем списке поля «Тип», проверьте наличие записи загрузки соответствующего файла конфигурации при старте приложения. Для этого выполните поиск в журнале для класса JobExternalLoader строки "Found resources:", результаты поиска будут содержать все XML-конфигурации загруженных сторонних задач. Список загруженных сторонних задач можно найти в журнале по строке ""Reloaded jobs".

### Контроль ошибок

Если в процессе выполнения пакетных задач возникли ошибки, то по завершении выполнения задачи на экране отобразится уведомление. В содержании уведомления будет отражено общее количество ошибок и ссылка для скачивания файла с подробным описанием каждой ошибки. Каждая запись в файле содержит наименование выполняемого действия, идентификатор записи, тип ошибки, ее описание.

# Проверка корректности реализации интеграции

Проверка корректности реализации интеграции осуществляется посредством выполнения действия, предусмотренного тем или иным способом (например, выполнение сторонней задачи, либо проверка факта наличия требуемых данных в сторонней информационной системе после выполнения какой-либо операции на стороне ПСКО в рамках информационного взаимодействия через API). В случае отсутствия требуемого результата необходимо проверить корректность настройки используемого интеграционного механизма в соответствии с п. 4 настоящего руководства.

# Сообщения системному программисту

Перечень сообщений системному программисту приведен в документе «Руководство администратора» АМСЯ.РРПСКО.И5.2-01.

Перечень сокращений и терминов

| **Сокращение/Термин** | **Наименование** |
| --- | --- |
| **Авторизация** | Предоставление пользователю прав на выполнение определенных действий. |
| **АРМ** | Автоматизированное рабочее место |
| **Аутентификация** | Проверка подлинности идентификатора, предъявляемого пользователем. |
| **БД** | База Данных |
| **Валидация** | Подтверждение факта выполнения требований, предназначенных для конкретного использования или применения, на основе представления объективных свидетельств. |
| **ГБУ** | Государственное бюджетное учреждение, созданное субъектом РФ и наделенное полномочиями, связанными с определением кадастровой стоимости |
| **ГКО** | Государственная кадастровая оценка — совокупность процедур, направленных на определение кадастровой стоимости |
| **Государственный надзор** | Федеральный государственный надзор за проведением государственной кадастровой оценки, в том числе за установленной 237-ФЗ периодичностью проведения государственной кадастровой оценки — деятельность, направленная на предупреждение, выявление, пресечение нарушений порядка проведения государственной кадастровой оценки, установленного 237-ФЗ, в том числе методических указаний о государственной кадастровой оценке |
| **ЕГРН** | Единый государственный реестр недвижимости |
| **ЕС УНСИ** | Единая система управления нормативно-справочной информацией |
| **Кадастровая стоимость** | Установленная в процессе государственной кадастровой оценки рыночная стоимость объекта недвижимости, определенная методами массовой оценки, или, при невозможности определения рыночной стоимости методами массовой оценки, рыночная стоимость, определенная индивидуально для конкретного объекта недвижимости в соответствии с законодательством |
| **Кластер** | Группа компьютеров, объединенных высокоскоростными каналами связи, представляющая с точки зрения пользователя единый аппаратный ресурс. |
| **Консолидация (записей)** | Объединение нескольких исходных записей в единую мастер-запись по определенным правилам |
| **Массовая оценка** | Процесс определения стоимости при группировании объектов оценки, имеющих схожие характеристики, в рамках которого используются математические и иные методы моделирования стоимости на основе подходов к оценке |
| **Модель данных** | Самодостаточное, логическое определение объектов модели данных (реестров, справочников и классификаторов), отражающее взаимосвязь между указанными объектами с учетом их атрибутивного состава и набора правил обработки информации в части поиска дубликатов и обеспечения качества данных. |
| **НСИ** | Нормативно-справочная информация |
| **Орган власти субъекта РФ** | Уполномоченный исполнительный орган власти субъекта Российской Федерации, принимающий решение о проведении государственной кадастровой оценке на территории субъекта |
| **Очистка данных** | Операция с данными, в ходе которой происходит приведение этих данных их к единому стандартному виду, который будет соответствовать определенным критериям. |
| **ПО** | Программное Обеспечение |
| **Программное обеспечение** | Совокупность компьютерных программ, процедур, правил и соответствующей документации системы обработки информации. |
| **СУБД** | Система Управления Базы Данных |
| **УРД** | Учетно-регистрационные действия |
| **ФГИС ЕГРН** | Федеральная государственная информационная система ведения Единого государственного реестра недвижимости |
| **ФЗ** | Федеральный закон Российской Федерации |
| **ФСТЭК** | Федеральная служба по техническому и экспортному контролю |
| **API** | Application Programming Interface |
| **API ПСКО** | Набор готовых процедур, функций и структур, предоставляемых ПСКО для организации доступа внешних систем |
| **CSV** | Coma Separated Value |
| **DB** | Data Base |
| **HTTP** | The Hypertext Transfer Protocol |
| **IP** | Internet Protocol |
| **JAR** | Java Archive |
| **JMS** | Java Message Service |
| **JNDI** | Java Naming and Directory Interface |
| **JSON** | JavaScript Object Notation |
| **PNG** | Растровый формат хранения графической информации |
| **REST** | Representational State Transfer |
| **SOAP** | Simple Object Access Protocol |
| **SVG** | Язык разметки масштабируемой векторной графики, входящий в подмножество расширяемого языка разметки XML |
| **WSDL** | Web Service Definition Language |
| **XML** | Extensible Markup Language |
| **XML** | Расширяемый язык разметки, удобный для создания и обработки документов программами и одновременно удобный для чтения и создания документов человеком |
| **XSD** | XML Schema Definition. Файл с расширением .XSD определяет схему XML-документа — его структуру, порядок элементов, правила, которым должен соответствовать документ |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Лист регистрации изменений*** | | | | | | | | | |
| *Изм.* | *Номера листов (страниц)* | | | | *Всего листов (страниц) в доку-менте* | *Номер доку-мента* | *Входящий номер сопроводи-тельного документа и дата* | *Подпись* | *Дата* |
| *изменен-ных* | *заменен-ных* | *новых* | *аннулиро-ванных* |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |