Платформа разработки Sezal

Руководство по эксплуатации программного продукта

www.galaktika.ru

Table of Contents

Введение ---- 3
Архитектура Sezal-приложения ---- 4
Структура Sezal-приложения ---- 7
Развертывание Sezal-приложения ---- 10
Настройка Sezal-приложения ---- 13
Взаимодействие с Sezal-приложением ---- 17
Основы работы с Sezal-приложением ---- 20

Введение

Платформа разработки Sezal (сокращенно - **платформа Sezal**, **Sezal**) может применяться для разработки приложений на рабочих станциях как на ОС Linux, так и на ОС Windows. Все входящие в состав платформы компоненты и инструменты являются кроссплатформенными.

В данной инструкции описывается функциональность **платформы Sezal**, доступная на рабочей станции под управлением ОС Windows.

Архитектура Sezal-приложения

Построенное на Entity Framework и **Платформе разработки Sezal** приложение в общем случае состоит из функциональных блоков.

На диаграмме приведены основные блоки, указано, когда и как эти блоки создаются и отображены зоны, где приложение может быть расширено.

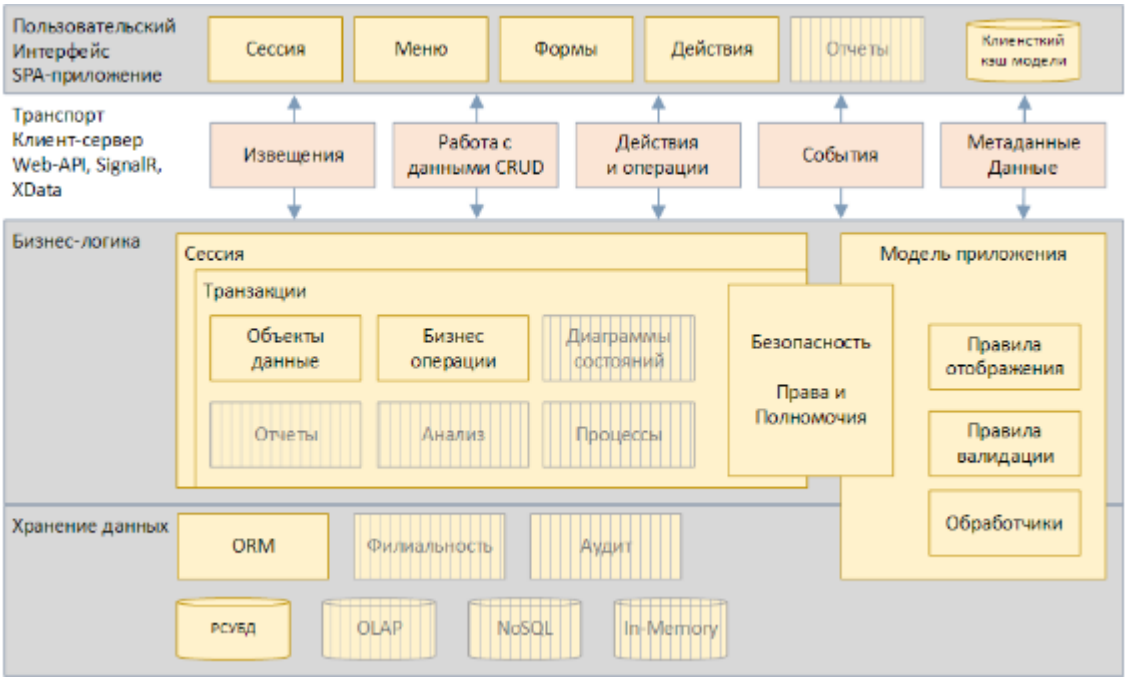


Рисунок 1. Архитектура приложения на платформе разработки Sezal

Persistent Objects

При разработке приложения с использованием **платформы Sezal** отсутствует необходимость создания базы данных в СУБД, настройки таблиц, полей, и проч. Для доступа к данным не требуется использовать низкоуровневые ODBC-конструкции. Вместо этого используется встроенная система ORM (Object-Relational Mapping) - библиотека классов Entity Framework (далее - EF).

EF позволяет описать данные для приложения, используя структуры языков программирования - классы, свойства и их атрибуты.

Так, для создания таблицы данных необходимо объявить класс. Его публичные свойства будут определены как поля данных в таблице. Можно создать столько таблиц, сколько нужно и установить отношения между ними с использованием специально предназначенных атрибутов.

Поскольку таблица данных описывается классом, фактические данные представляют коллекцию экземпляров класса. Таким образом, чтобы изменить поле в записи, необходимо получить требуемый объект из коллекции и изменить его свойства. Это более простой и естественный

способ обработки данных для разработчика. При этом скрыты все детали реализации, что позволяет сосредоточиться на прикладной логике приложения.

Бизнес классы и контроллеры

Платформа Sezal предоставляет набор базовых классов, которые можно использовать в приложении.

Контроллеры являются объектами, которые управляют потоком приложения. Они также отвечают за взаимодействие с пользователем. Любое приложение, построенное на **платформе Sezal**, использует ряд встроенных контроллеров, которые, прежде всего, служат для управления данными. С их помощью пользователь может добавлять новые записи, удалять существующие, выполнять полнотекстовый поиск и т. д.

Контроллеры также служат в качестве контейнеров для действий. Как EF-классы являются абстракциями таблиц данных, действия являются абстракциями элементов взаимодействия пользователей: кнопки, меню и т. п. Действие определяет визуальное представление элемента пользовательского интерфейса и связанный с ним код. Таким образом, нет необходимости заниматься низкоуровневой реализацией деталей конкретных редакторов, панелей инструментов, контекстных меню и другими подобными задачами.

Прикладные классы и контроллеры

Поскольку **платформа Sezal** позволяет разработчику значительную часть времени уделять прикладной логике приложения, в первую очередь от него требуется разработать модель прикладных данных и описать ее с помощью прикладных классов. Прикладные классы должны быть унаследованы от одного из базовых классов EF.

После того, как создана модель данных, необходимо реализовать взаимодействие как между классами, так и между классами и пользователем. Для этого разработчику необходимо определить, какие прикладные действия будут производиться с прикладными классами и реализовать эти действия в виде контроллеров.

Стандартное поведение встроенных контроллеров при необходимости может быть изменено разработчиком в прикладных контроллерах.

Модель приложения

Одной из основных особенностей использования **платформы Sezal** является автоматическое построение пользовательского интерфейса. Это означает, что визуальные элементы приложения создаются автоматически, основываясь на данных классов, объявленных в коде приложения.

Платформа Sezal анализирует объявления классов в коде приложения и генерирует метаданные - данные, которые определяют структуру базы данных и функции приложения. Эти метаданные

называются моделью приложения. Это мощный инструмент, который позволяет настраивать приложение без изменения исходного кода.

Файлы модели приложения хранятся в формате XML и могут легко редактироваться вручную.

Формы

Как уже отмечалось, одной из ключевых особенностей **платформы Sezal** является автоматическая генерация пользовательского интерфейса на основе данных приложения. Предположим, что есть EF-класс, который описывает данные о контакте с какой-либо организацией. Это все, что необходимо для того, чтобы получить приложение для хранения контактной информации. Можно просто запустить такое приложение, и в нем будет отображен список контактов, появятся возможности добавлять новые записи или изменять существующие. Эти операции выполняются через автоматически сгенерированный набор отдельных редакторов.

Используемый для отображения данных и управления этими данными автоматически созданный элемент пользовательского интерфейса называется формой.

Структура Sezal-приложения

Для разработки приложения с использованием **платформы Sezal** необходимо создать в Microsoft Visual Studio новое решение (Application Solution).

Следует создать:

- Persistent Module;
- Logic Module;
- Forms Module.

Модуль

Построение **Sezal**-приложений заключается в разработке необходимого набора модулей и подключении различных комбинаций этих модулей в различные приложения. И Module Project, и Forms Project являются модулями.

В отличие от стандартной библиотеки классов, модуль содержит потомка" класса ModuleBase, по которому выделяет в приложении модули. Это необходимо, потому что при **платформа Sezal** автоматической генерации пользовательского интерфейса используются только классы, объявленные в проектах модулей. Например, если объявить контроллер в модуле, приложение будет создавать этот контроллер в каждом окне. Если же объявление будет в обычной библиотеке классов, то **платформа Sezal** не найдет этот контроллер.

Можно преобразовать обычный проект в модуль, если определить в нем потомка класса ModuleBase.

Forms Application Project не является модулем.

Все классы, необходимые для автоматической генерации пользовательского интерфейса, должны быть объявлены в используемых модулях.

Платформа Sezal предоставляет большой набор модулей для использования в приложениях.

Существует несколько способов подключения модуля к приложению.

Модуль, зарегистрированный в панели инструментов, можно подключить в дизайнере модуля или приложения.

Модуль можно подключить напрямую в коде инициализации модуля - для этого необходимо добавить подключаемый модуль в коллекцию `ModuleBase.RequiredModuleTypes`:

```
public sealed class Sezal StartupModule: ModuleBase
{
```

```
  public Sezal StartupModule()
  {
    InitializeComponent();
  }
  private void InitializeComponent()
  {
    this.RequiredModuleTypes.Add(typeof(T2Plus.Sezal .SystemModule.SystemModule));
  }
}
```

Модуль можно подключить и в коде инициализации приложения - для этого следует добавить подключаемый модуль в коллекцию **платформы** `Application.Modules`:

```
public class Sezal StartupFormsApplication: Application
{
  public FormsApplication()
  {
    InitializeComponent();
  }
  private void InitializeComponent()
  {

    …

    this.viewVariantsModule1 = new T2Plus.Sezal .ViewVariantsModule.ViewVariantsModule();

    …

    this.Modules.Add(this.viewVariantsModule1);

    …
  }

  private T2Plus.Sezal .ViewVariantsModule.ViewVariantsModule viewVariantsModule1;

  …

}
```

Можно указать список подключаемых модулей в конфигурационном файле приложения:

```
<appSettings>
  <add key="Modules" value="T2Plus.Core.Module;" />
```

```
</appSettings>
```

И добавить подключаемые модули, которые перечислены в файле конфигурации, в коллекцию **платформы Sezal** `Application.Modules`.

Развертывание Sezal-приложения

Настройка подключения к базе данных

Подключение к базе данных настраивается в конфигурационном файле приложения:

```
<connectionStrings>
  <add name="ConnectionString" connectionString="ImMemory; Pooling=false; Data
Source=сервер; Initial Catalog=база данных;" />
</connectionStrings>
```

Развертывание приложения

Приложение хранится и выполняется на каждой рабочей станции. Данный подход применим в случае, если:

- нет доступа к терминальному серверу;
- не требуется удаленный доступ к приложению;
- рабочие станции обладают достаточными ресурсами и производительностью.

При таком подходе обновлять файлы приложения необходимо на каждой рабочей станции. Чтобы иметь только одну копию, можно хранить приложение на файловом сервере, а выполнять - на каждой рабочей станции. Для этого в конфигурационном файле приложения должен быть установлен параметр:

```
<runtime>
  <loadFromRemoteSources enabled="true"/>
</runtime>
```

Еще один способ иметь только одну копию приложения - хранить и выполнять приложение на терминальном сервере (например, через Remote Desktop Connection).

Данный подход применим в случае, если:

- есть доступ к надежному терминальному серверу, обладающему высокой производительностью;
- требуется удаленный доступ к приложению;
- на рабочих станциях используются различные операционные системы;
- рабочие станции не обладают достаточными ресурсами или производительностью.

При желании можно использовать все эти подходы одновременно.

Обновление приложения

Создание и обновление структуры базы данных

Приложение выполняется успешно в случае, когда версия приложения (версии модулей приложения) соответствует версии базы данных. При запуске выполняется проверка совместимости версий. Для этого приложение пытается получить доступ к базе данных.

Для хранения версии приложения в базе данных создается таблица ModulesInfo. При первом выполнении приложения в этой таблице сохраняются текущие версии модулей приложения и процесс проверки версии проходит успешно.

При запуске приложения после обновления версии процесс проверки версии обнаружит несоответствие и выдаст исключение.

Обновление приложения

Обновление приложения осуществляется заменой файлов приложения файлами новой версии приложения. При обновлении приложения нежелательно заменять пользовательский конфигурационный файл приложения и файл модели пользователя (model.xml).

По умолчанию файл модели пользователя хранится в каталоге приложения. Чтобы обезопасить пользовательскую модель от случайной потери, можно установить в конфигурационном файле приложения параметр:

```
<appSettings>
  <add key="UserModelDiffsLocation" value="CurrentUserApplicationDataFolder" />
</appSettings>
```

Автоматическое обновление приложения

Если приложение размещено на рабочих станциях пользователей, то для обновления приложения необходимо обновить файлы приложения на каждой рабочей станции. Для экономии времени можно автоматизировать процесс обновления на рабочих станциях, выполнив следующие действия:

1. Выделить общедоступный сетевой ресурс для размещения файлов новой версии приложения;

```
<appSettings>
  <add key="NewVersionServer" value="UNC путь к файлам новой версии" />
</appSettings>
```

2. Теперь для обновления приложения достаточно разместить файлы новой версии на выделенном сетевом ресурсе и обновить структуру базы данных. После этого при запуске приложения на рабочей станции произойдет автоматическое обновление копии приложения пользователя.

На сетевом ресурсе следует размещать только файлы новой версии, так как при обновлении на рабочую станцию будет скопировано все содержимое сетевого ресурса.

Настройка безопасности сервера базы данных

Аутентификация пользователей демоверсии **платформы Sezal** осуществляется в basic-режиме.

При такой настройке безопасности управление доступом к приложению осуществляется включением в группу (исключением из группы) учетных записей пользователей.

Настройка Sezal-приложения

Модель приложения

Модель приложения формируется из слоев. Первый слой формируется из кода. Следующие слои представляют изменения, хранящиеся в файлах model.xml, которые расположены в используемых модулях. Окончательный слой представляет изменения, хранящиеся в файле model.xml в проекте приложения. При загрузке модели приложения все эти изменения накладываются одно на другое послойно.

Кроме того, существует слой, который содержит изменения, сделанные пользователем - это самый верхний слой модели приложения, который загружается во время выполнения приложения. Пользовательский слой хранится в файле model.xml.

Во время разработки можно настроить модели уровней модуля и приложения.

По умолчанию файл model.xml хранится в каталоге приложения. Чтобы обезопасить пользовательскую модель от случайной потери, можно установить в конфигурационном файле приложения параметр:

```
<appSettings>
  <add key="UserModelDiffsLocation" value="CurrentUserApplicationDataFolder" />
</appSettings>
```

Для сброса всех настроек пользователя достаточно удалить этот файл.

Настройка внешнего вида формы.

Модель приложения предоставляет широкие возможности по настройке различных параметров внешнего вида списковых и карточных форм.

Настройка поведения формы, приложения.

Модель приложения предоставляет возможности по настройке поведения форм. При подключении дополнительных модулей эти возможности можно расширить. Так, например, после подключения модуля `ViewVariantsModule` становится доступной возможность настроить несколько различных представлений одной формы. При подключении модуля `ConditionalAppearanceModule` становится доступной возможность настраивать внешний вид прикладных объектов в зависимости от заданных условий.

Возможные настройки приведены в следующей таблице.

Есть возможность настроить	Списковая форма		Карточная форма				Дей
	ячейка	редактируемая ячейка	редактор поля	статический текст	элемент разметки	группа элементов разметки	
Цвет шрифта	да		Да	да	да	да	
Стиль шрифта	да		Да	да	да	да	
Цвет фона	да		Да	да	да		
Доступность		да	Да				да
Видимость			Да		да	да	да

Не следует использовать `ConditionalAppearanceModule` для решения задач разграничения прав доступа, для этого в **Sezal**-приложении есть `Security System`.

Динамические свойства классов

При необходимости можно расширить прикладной класс новыми свойствами. Такие свойства определяются на уровне кода модуля.

Для расширения класса следует добавить определение нового свойства в методе настройки метаданных модуля:

```
public override void CustomizeTypesInfo(ITypesInfo typesInfo)
{
  base.CustomizeTypesInfo(typesInfo);
  var typeInfo = typesInfo.FindTypeInfo(typeof(T2Plus.Aps.ApsResource));

  if (typeInfo != null)
    typeInfo.CreateMember("Amount ", typeof(decimal?));
}
```

Во время генерации модели приложения в класс `T2Plus.Aps.ApsResource` будет добавлено новое свойство `Amount` типа `decimal`. Значение этого свойства будет храниться в базе данных.

Для реализации расширения класса вычисляемым свойством необходимо создать класс:

```
public sealed class CalculatedMember: Sezal .EF.Metadata.XPCustomMemberInfo
{
  …
  public override object GetValue(object theObject)
  {
    var theObjectType = theObject.GetType();
    if (theObjectType == typeof(T2Plus.Aps.ApsResource))
    {
      var pecypc = (T2Plus.Aps.ApsResource)theObject;
      if (Name == "Amount")
        return GetResourceQuantity(pecypc);
    }

    return base.GetValue(theObject);
  }

  public override void SetValue(object theObject, object theValue)
  {
    var theObjectType = theObject.GetType();
    if (theObjectType == typeof(T2Plus.Aps.ApsResource))
    {
      var pecypc = (T2Plus.Aps.ApsResource)theObject;
      if (Name == "Amount")
        SetResourceQuantity (pecypc);
    }
  }
}
```

Методы `GetResourceQuantity` и `SetResourceQuantity` должны реализовывать логику получения и установки значения.

Добавить определение нового свойства в метод настройки метаданных модуля:

```
public override void CustomizeTypesInfo(ITypesInfo typesInfo)
{
  base.CustomizeTypesInfo(typesInfo);
  var classInfo = Sezal
TypesInfo.EFTypeInfoSource.GetClassInfo(typeof(T2Plus.Aps.ApsResource));
  if (classInfo != null)
  {
    CalculatedMember.CreateMember(classInfo, "Amount", тип данных);
    Sezal TypesInfo.Instance.RefreshInfo(typeof(T2Plus.Aps.ApsResource));
  }
}
```

В данном случае в класс `T2Plus.Aps.ApsResource` будет добавлено новое свойство `Amount`, тип которого должен совпадать с типом, возвращаемым методом `GetResourceQuantity`. Значение такого свойства вычисляется и не хранится в базе данных.

В обоих случаях новое свойство становится доступным в модели приложения. Его можно использовать на формах. В коде использовать значение этого свойства можно следующим образом:

```
var ресурс = new T2Plus.Aps.ApsResource(Session);

ресурс.SetMemberValue("Amount", newvalue);

var value = ресурс.GetMemberValue("Amount");
```

Использовать динамическое расширение классов следует очень осторожно, так как это существенно влияет на производительность приложения.

Взаимодействие с Sezal-приложением

Вход в систему

Чтобы начать работу с системой, необходимо перейти по ссылке на веб-страницу приложения **Sezal** либо открыть браузер и вставить путь в адресную строку.

Во всплывающем окне необходимо ввести свое имя пользователя, а также первоначальный пароль, выданный администратором, после нажать кнопку **Вход в систему**.

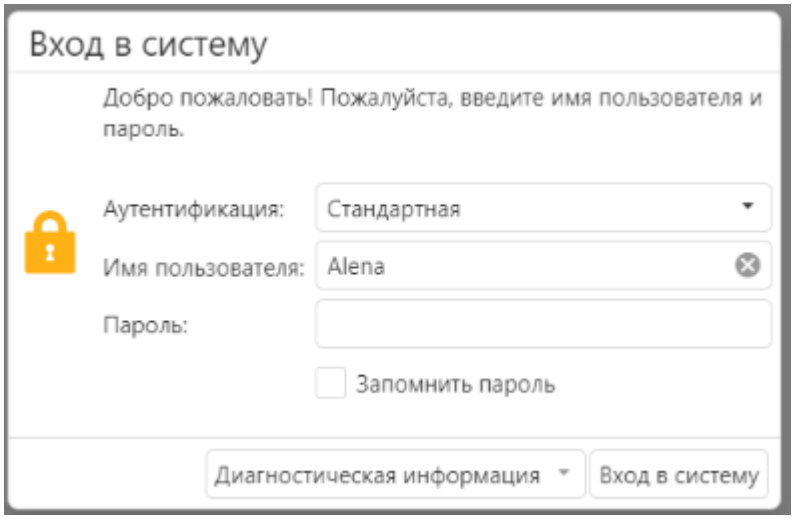


Рисунок 2. Страница входа в систему

После этого система предложит сменить пароль на новый.

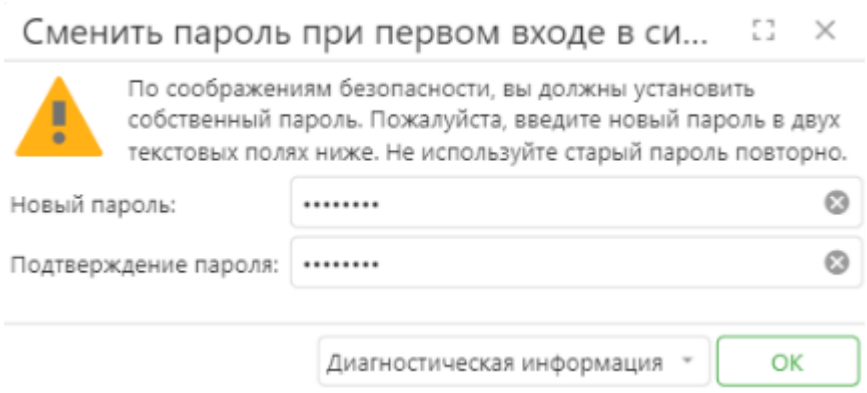


Рисунок 3. Смена пароля при первом входе в систему

Откроется главное окно системы.

Завершение работы

Для завершения работы приложения можно воспользоваться как средствами операционной системы, так и встроенным в **Sezal**-приложение средствами.

При использовании любого ОС-средства система немедленно прекратит работу, не требуя подтверждения (так называемый быстрый выход из системы).

Для завершения работы с помощью собственных средств **Sezal**-приложения следует воспользоваться общесистемной функцией "Выйти" или пиктограммой  на панели заголовка.

Основные понятия пользовательского интерфейса

В основе лежат два способа организации диалога пользователя с системой: меню и экранные формы.

Вся поступающая от пользователя к системе информация по своему содержанию может быть отнесена к понятиям "команды" или "данные".

Команды - это информация о намерениях пользователя, выраженная в виде прямых указаний, которые должны быть выполнены системой. Отдавая команды, пользователь направляет работу системы, заставляя ее сделать то, что ему необходимо в тот или иной момент. Само управление работой системы в большинстве случаев производится с помощью меню и заключается в процедуре выбора одного из предложенных вариантов действий (режимов работы).

Данные - это информация числового и (или) текстового характера, которая сообщается системе с целью ее последующей обработки. Операция занесения данных производится с использованием различных экранных форм путем ручного набора на клавиатуре или выбором информации из предложенных системой списков.

Команды выполняются системой сразу после их ввода.

Данные могут быть использованы не сразу после ввода, а спустя какое-то время, многократно, в течение многих сеансов работы. Поэтому в любых системах, связанных с обработкой больших объемов информации, имеется база данных (БД) - все множество необходимых для работы данных, организованных (упорядоченных) особым образом и хранящихся на носителе в информационно-вычислительной инфраструктуре (ИВИ) компании.

Вся хранящаяся в БД информация представлена в виде совокупности связанных таблиц.

Таблица - представляет собой совокупность данных одинаковой структуры.

Строки таблицы называются записями. Каждая таблица может состоять из практически неограниченного числа записей (на самом деле, объем определяется при проектировании ИВИ).

Запись состоит из отдельных элементов - полей, различающихся смысловым содержанием заносимых в них сведений. Аналогом поля в таблице БД является клетка таблицы на бумаге. В пределах одной таблицы число полей во всех имеющихся записях всегда одинаково. Структура записи, то есть состав образующих ее полей, зафиксирована в момент создания таблицы.

В документации и интерфейсах системы используются такие понятия как: каталоги, справочники и классификаторы.

Каталог/справочник - это таблица или группа связанных таблиц, содержащих систематизированную информацию, имеющую долгосрочный характер и предназначенную для ввода данных в экранные формы методом выбора из каталога.

Основы работы с Sezal-приложением

Далее приведено описание интерфейса пользователя на примере приложения **Галактика EAM на платформе Sezal** разработанного с использованием **платформы Sezal**.

Главное окно Sezal-приложения

Главное окно является интерфейсом для управления и настройки работы модулей приложения. Содержит следующие основные элементы:

- **Главное меню**: содержит общие функции системы и функции для текущего представления, а также позволяет настроить внешний вид системы.
- **Панель инструментов**: содержит функции, актуальные для текущей формы. Рабочие окна приложения могут содержать главную панель инструментов и локальную панель инструментов рабочей области либо отдельных вкладок окна.
- **Рабочая область**: область для работы с данными.
- **Строка состояния**: отражает справочную информацию подключения к приложению: имя сервера базы данных; название базы данных; данные пользователя, от имени которого запущено приложение.

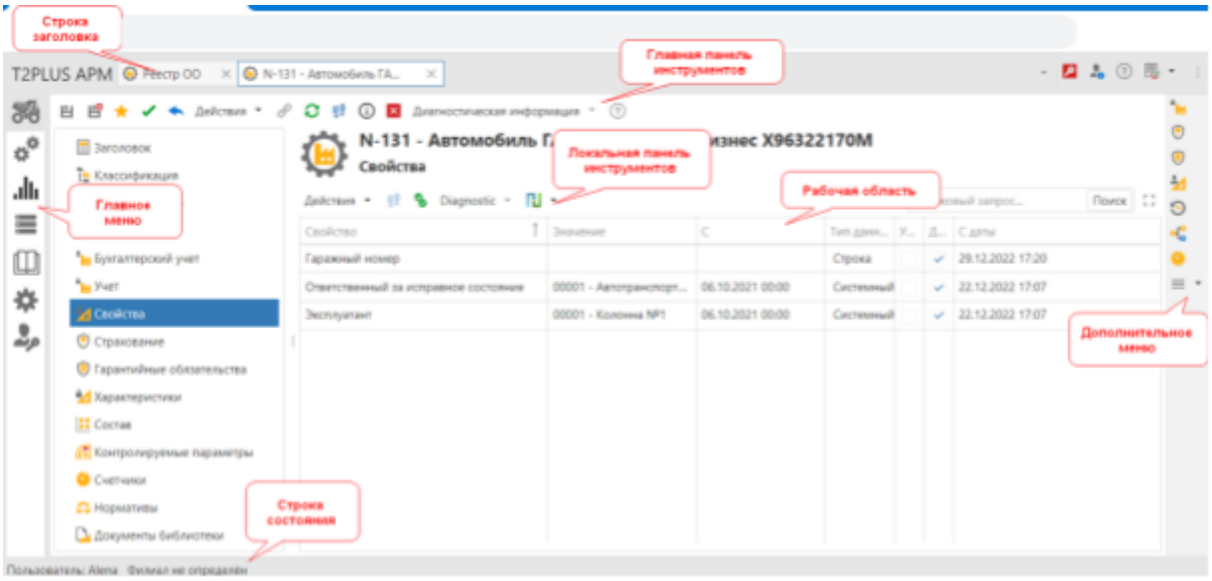


Рисунок 4. Главное окно приложения

Главное меню

Главное меню системы представлено в виде ряда пиктограмм, расположенных вертикально справа вверху окна интерфейса.

Рисунок 5. Вид главного меню

Панель инструментов

В интерфейсах существуют главная панель инструментов (1) и локальная панель (2) инструментов.

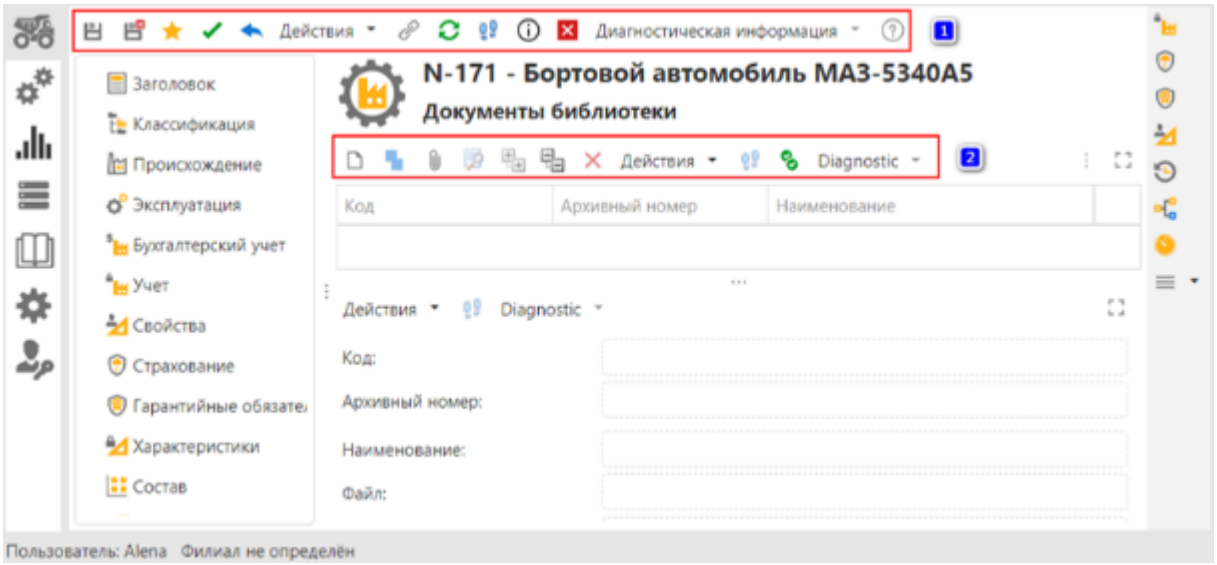


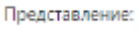
Рисунок 6. Панели инструментов системы

Главная панель инструментов

Ниже приведены функции главной панели инструментов.

Пиктограмма, кнопка	**Действие**	**Примечание**
Создание объектов		
	Создать	В зависимости от контекста выполняется создание новой записи и автоматический переход к карточной форме записи.

Пиктограмма, кнопка	Действие	Примечание
		Наличие стрелки на пиктограмме обозначает возможность выбора из нескольких вариантов при создании новой записи.
	Добавить/удалить избранное	Отметить выбранные записи как "избранные". Доступна в реестре ОО.
Сохранение		
	Сохранить	Сохранить изменения. Кнопка активна только при наличии внесенных изменений.
	Сохранить и закрыть	Сохранить изменения и закрыть текущую карточную форму. Кнопка активна только при наличии внесенных изменений.
	Сохранить и создать новый	Сохранить внесенные изменения и создать новую запись. Текущая запись при этом будет закрыта.
Правка		
	Удалить	Удалить запись.
	Контроль	Проверить выбранные объекты во всех имеющихся контекстах.
	Копировать	Копировать текущую запись в буфер обмена.
	Вставить	Вставить запись, находящуюся в буфере обмена.
Редактирование записи		
Действия	Дополнительные действия	В зависимости от контекста возможно наличие дополнительных действий применительно к текущим записям.

Пиктограмма, кнопка	Действие	Примечание
	Раскрыть текущий узел	Развертывание текущего узла и его дочерних узлов.
	Свернуть все	Свернуть все узлы.
	Гиперссылка	Формирование гиперссылки на интерфейс.
Откат/возврат изменения		
	Отменить	Отменить последнее выполненное действие.
Вид		
	Сменить текущий вариант представления: список/ иерархия	Переключение режима отображения данных для иерархических каталогов.
	Обновить	Обновить данные рабочей области.
	Панели	Список панелей для отображения дополнительных данных по текущей записи.
	Данные аудита	Показать данные аудита для текущей записи.
	Правила отображения	Настройка пользовательских правил отображения. Доступно по кнопке **Действия** на главной панели инструментов.
	Параметры записи	Открытие окна с параметрами выбранной записи.
Фильтры		
	Отображать в статусе *Черновик*	Показать записи со статусом *Черновик*.
	Отображать в статусе *Архив*	Показать записи со статусом *Архив*.
	Отображать в статусе *Удержание*	Показать записи со статусом *Удержание*.

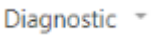
Пиктограмма, кнопка	Действие	Примечание
Отчеты		
	Правка	
	Экспорт/импорт	Экспортировать/импортировать шаблон отчета в файл.
Экспорт/импорт		
	Экспорт/импорт	Импорт каталога из системы.
	Импорт из файла Excel	Импорт каталога из Excel-файла.
	Экспорт в файл Excel	Экспорт каталога в Excel-файл.
	Множественный импорт из Excel	Множественный импорт из Excel.
	Множественный экспорт в файлы Excel	Множественный экспорт в Excel.
	Импорт в XML	Импорт каталога в XML.
	Экспорт в XML	Экспорт каталога в XML.
Открытие объектов		
	Открыть объект	Открыть карточку текущего объекта, связанного с активным редактором.

Локальная панель инструментов

Ниже приведены функции локальной панели инструментов.

Пиктограмма, кнопка	Действие	Примечание
Создание объектов		

Пиктограмма, кнопка	Действие	Примечание
	Создать	В зависимости от контекста выполняется создание новой записи и автоматический переход к карточной форме записи. Наличие стрелки на пиктограмме обозначает возможность выбора из нескольких вариантов при создании новой записи.
	Клонировать	Создать новую запись (дерево) с копированием данных текущей. Наличие стрелки на пиктограмме обозначает возможность выбора из нескольких вариантов при клонировании.
	Создать множество объектов	Групповое создание записей.
	Прикрепить объект	
Сохранение		
	Сохранить	Сохранить изменения. Кнопка активна только при наличии внесенных изменений.
	Сохранить и закрыть	Сохранить изменения и закрыть текущую карточную форму. Кнопка активна только при наличии внесенных изменений.
	Сохранить и создать новый	Сохранить внесенные изменения и создать новую запись. Текущая запись при этому будет закрыта.
Правка		
	Копировать	Копировать текущую запись в буфер обмена.
	Вставить	Вставить запись, находящуюся в буфере обмена.
Правка записи		
Действия	Дополнительные действия	В зависимости от контекста возможно наличие дополнительных действий применительно к текущим записям.

Пиктограмма, кнопка	Действие	Примечание
Установить статус	Статус готовности	Изменение статуса текущей записи.
	Переместить вниз/ вверх	Уменьшает/увеличивает номер по порядку на единицу.
Открытие объекта		
	Открыть объект	Открыть объект, связанный с активным редактором.
	Открыть документ библиотеки	Доступно в карточке объекта обслуживания.
Вид		
	Данные аудита	Показать данные аудита для текущей записи.
	Параметры записи	Краткая информация о дате и авторе, создавшем/ изменившем текущую запись.
	Правила отображения	Настройка пользовательских правил отображения.
Диагностика		
	Диагностическая информация	Диагностическая информация о правилах.
Фильтры		
	Отображать в статусе *Черновик*	Показать записи со статусом *Черновик*.
	Отображать в статусе *Архив*	Показать записи со статусом *Архив*.
	Отображать в статусе *Удержание*	Показать записи со статусом *Удержание*.

Пиктограмма, кнопка	Действие	Примечание
	Отображать в статусе *Блокировка*	Показать записи со статусом *Блокировка*.
Экспорт		
	Экспорт/импорт	Импорт каталога из системы.
	Импорт из файла Excel	Импорт каталога из Excel-файла.
	Экспорт в файл Excel	Экспорт каталога в Excel-файл.
	Множественный импорт из Excel	Множественный импорт из Excel.
	Множественный экспорт в файлы Excel	Множественный экспорт в Excel.
	Импорт в XML	Импорт каталога в XML
	Экспорт в XML	Экспорт каталога в XML

Контекстное меню в списочной форме

Списочная форма (список) используется для представления данных в виде перечня объектов. Данные могут отображаться в виде линейного списка или в виде иерархической структуры. Для работы со списками может использоваться контекстное меню, которое открывается щелчком правой кнопки мыши.

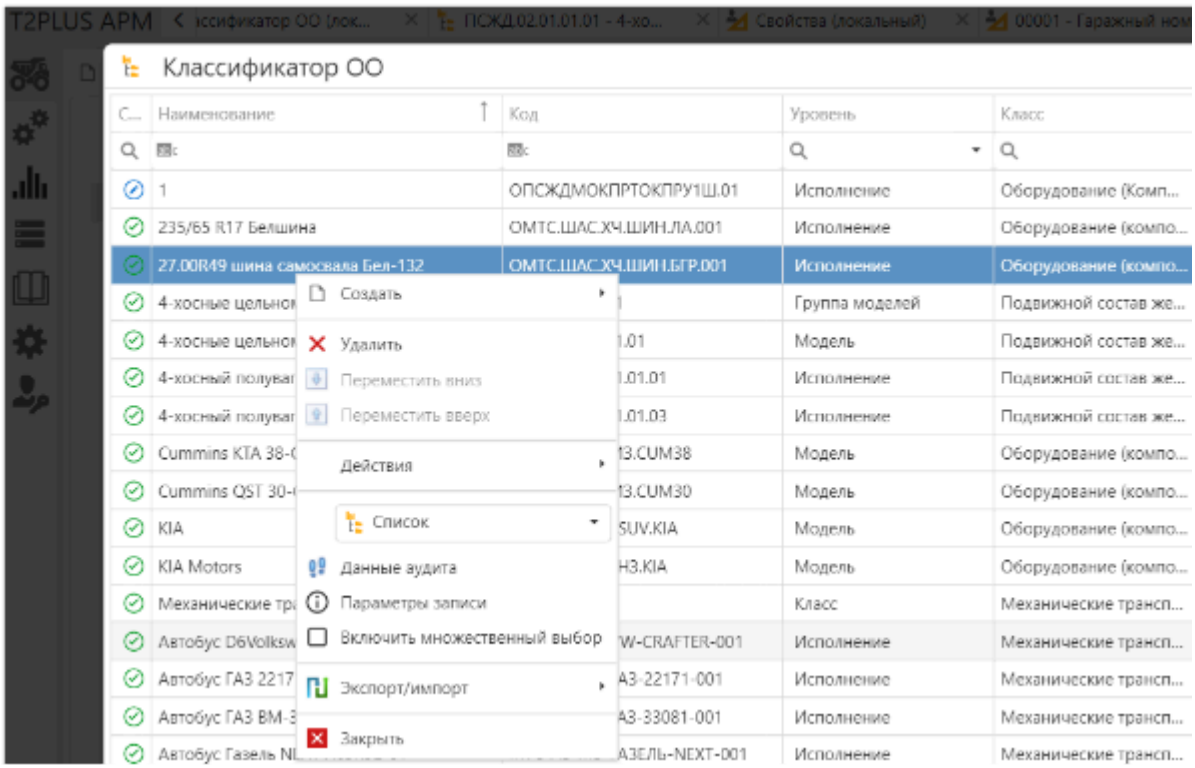


Рисунок 7. Пример контекстного меню в списочной форме

В данном меню в зависимости от контекста могут отображаться следующие функции.

Пиктограмма, кнопка	**Действие**	**Примечание**	
	Создать	Создание новой записи и автоматический переход к карточной форме записи. Наличие стрелки на пиктограмме обозначает возможность выбора из нескольких вариантов при создании новой записи.	
	Удалить	Удаление записи.	
	Переместить вниз/вверх	Уменьшает/увеличивает номер по порядку на единицу.	
Действия	Дополнительные действия	В зависимости от контекста возможно наличие дополнительных действий применительно к текущим записям.	
	Раскрыть текущий узел	Развертывание текущего узла и его дочерних узлов.	

Пиктограмма, кнопка	Действие	Примечание	
	Свернуть все	Свертывание всех узлов.	
Список	Сменить текущий вариант представления: список/иерархия	Переключение режима отображения данных для иерархических каталогов.	
	Данные аудита	Показать данные аудита для текущей записи.	
	Параметры записи	Открытие окна с параметрами выбранной записи.	
Включить	Включить множественный выбор	Возможность выбора несколько объектов из справочника.	
	Экспорт/импорт	Импорт каталога из системы.	
	Импорт из файла Excel	Импорт каталога из Excel-файла.	
	Экспорт в файл Excel	Экспорт каталога в Excel-файл.	
	Множественный импорт из Excel	Множественный импорт из Excel.	
	Множественный экспорт в файлы Excel	Множественный экспорт в Excel.	
	Импорт в XML	Импорт каталога в XML.	
	Экспорт в XML	Экспорт каталога в XML.	
Закрыть	Закрыть	Закрытие текущего окна.	

Сортировка, группировка, выбор столбцов

Сортировка по колонке

Функции Сортировка по возрастанию, Сортировка по убыванию позволяют сортировать данные по значению в выбранной колонке (колонкам) в прямой и обратной последовательности. Для быстрой сортировки данных следует кликнуть мышкой по заголовку нужной колонки.

Заголовок колонки содержит признак наличия сортировки:

- по возрастанию;
- по убыванию.

Для сброса сортировки по колонке используется функция Сбросить сортировку контекстного меню колонки.

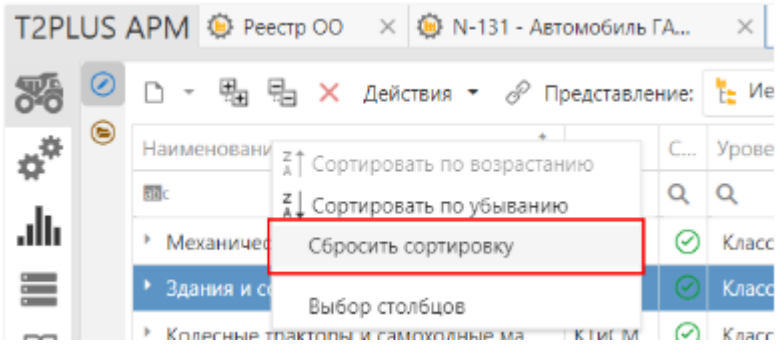


Рисунок 8. Контекстное меню заголовка колонки "Наименование"

Группировка данных

Группировка данных по значениям колонок позволяет отображать список в виде иерархии записей.

Для группировки по колонке в контекстном меню заголовка колонки следует выбрать функцию Сгруппировать данные по этому столбцу.

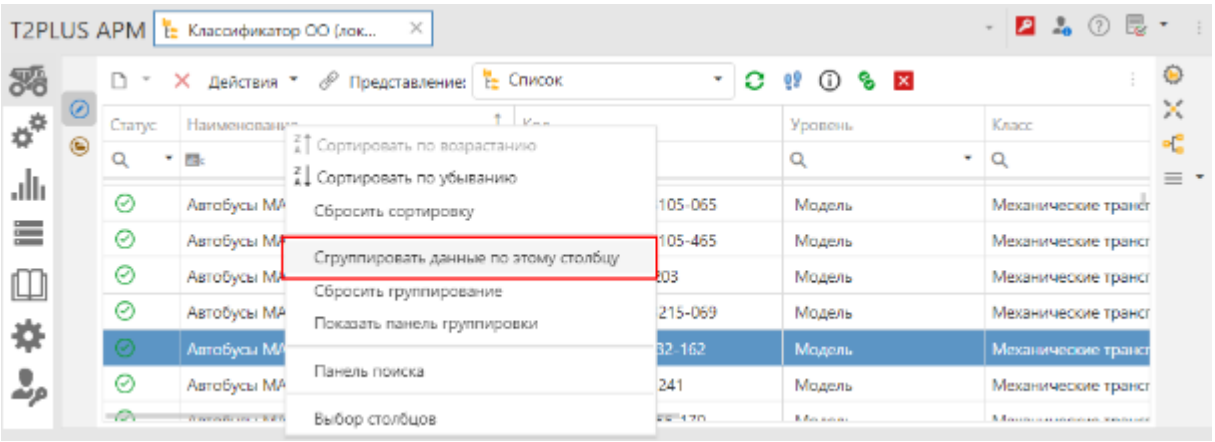


Рисунок 9. Группировка записей по колонке

Ниже приведен внешний вид с группировкой по наименованию объектов обслуживания.

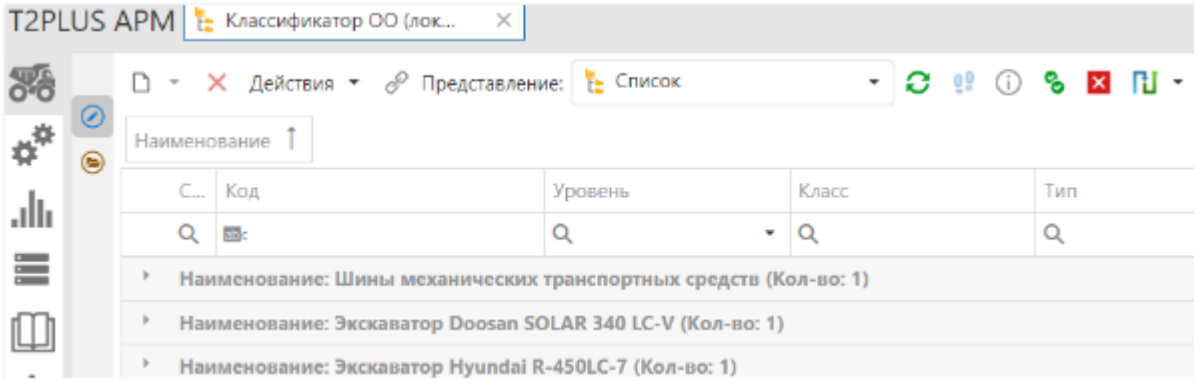


Рисунок 10. Результат группировки по колонке

Если активировать функцию Показать панель группировки, можно перетащить заголовок колонки в область группировки.

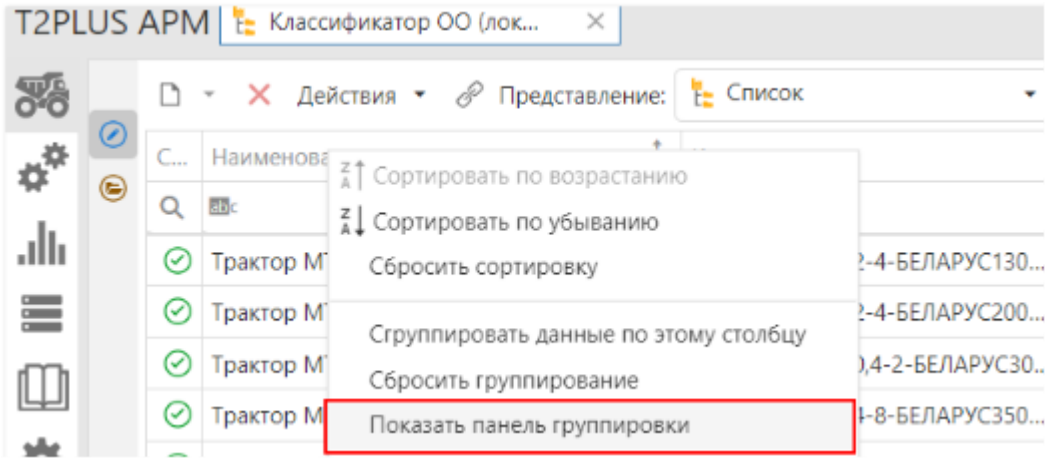


Рисунок 11. Функция "Показать панель группировки"

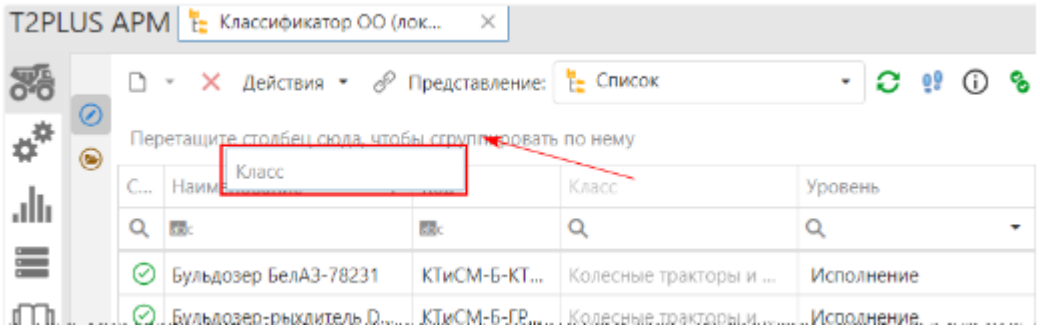


Рисунок 12. Группировка по столбцу "Класс"

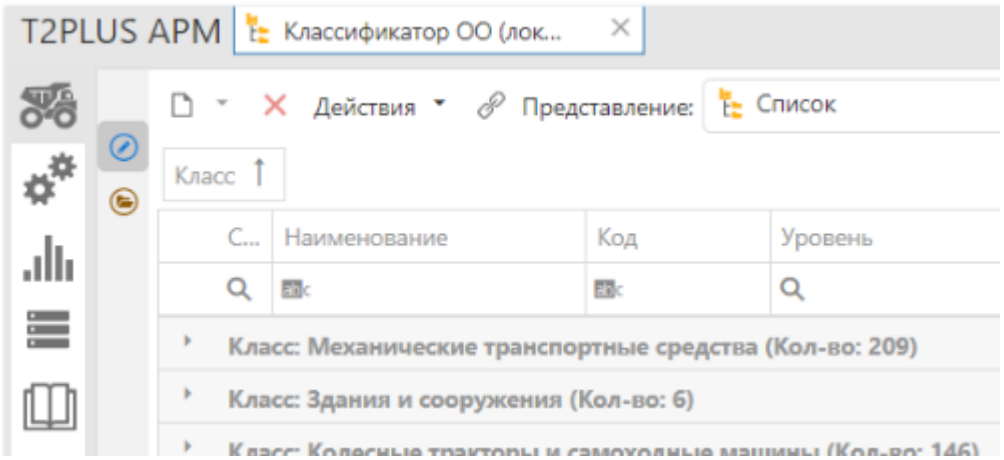


Рисунок 13. Результат группировки списка по столбцу "Класс"

Для создания многоуровневой структуры следует выполнить группировку по нескольким колонкам.

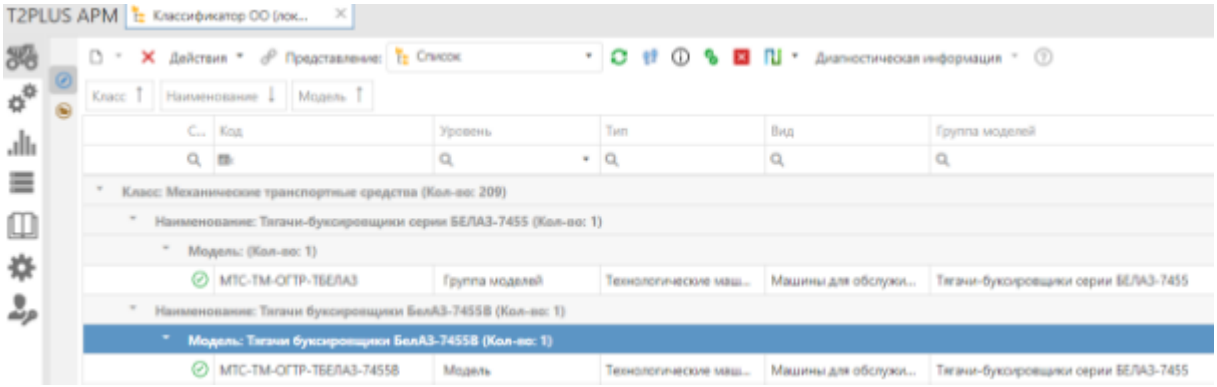


Рисунок 14. Группировка по нескольким колонкам

Изменение порядка группировки выполняется путем перетаскивания заголовка колонки на нужную позицию в области группировки. Для удаления уровня группировки следует перетащить нужный заголовок из области группировки в строку заголовков списка на нужную позицию.

Для настройки отображения многоуровневого списка используются функции локального меню области группировки.

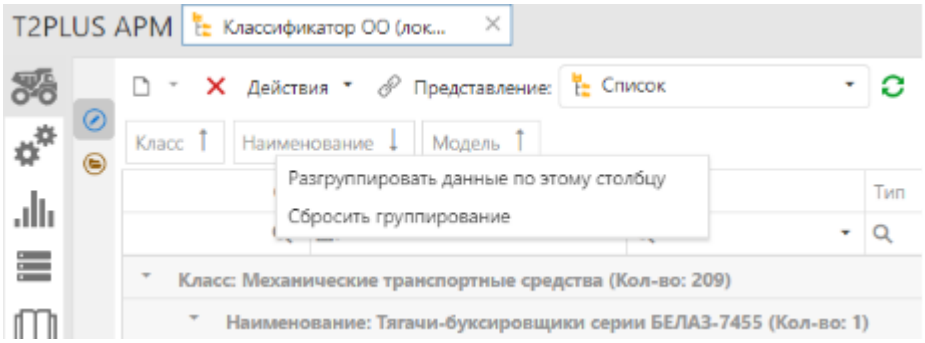


Рисунок 15. Контекстное меню области группировки

Настройка столбцов

Функция Выбор столбцов позволяет настроить видимость столбцов в интерфейсе системы.

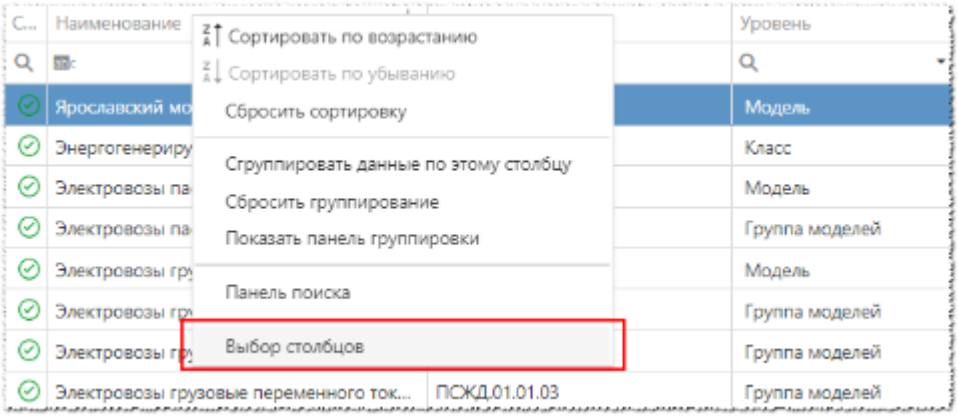


Рисунок 16. Функция "Выбор столбцов"

Для скрытия столбцов необходимо убрать пометку в перечне имеющихся в таблице столбцов.

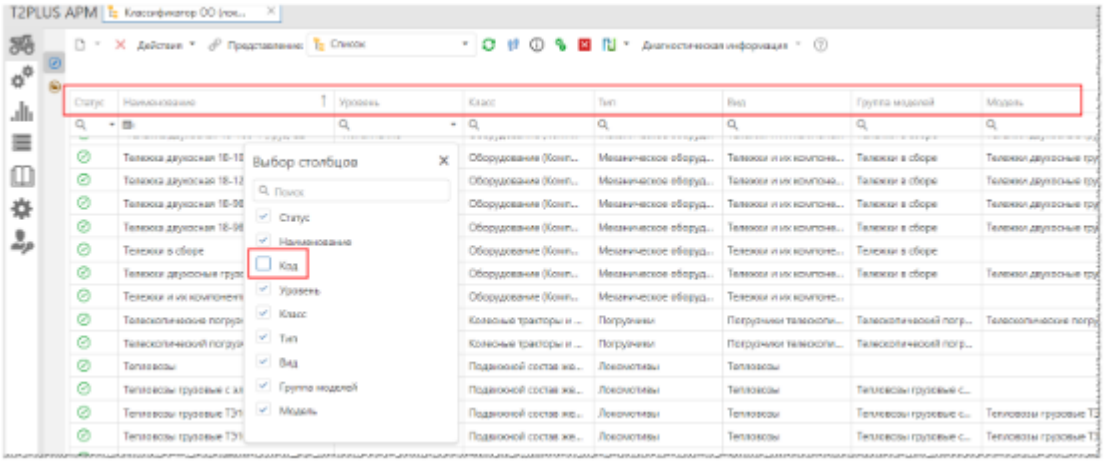


Рисунок 17. Настройка отображения столбцов

Для изменения ширины колонки вручную следует перетащить соответствующую границу колонки. Для изменения порядка следования колонок в списке требуется перетащить заголовок колонки на нужную позицию в списке.

Редакторы и способы ввода информации

Ниже приведены методы ввода информации и связанные редакторы, предусмотренные в приложении, построенном на **платформе Sezal**.

Поля, доступные для редактирования пользователям, выделены белым цветом, при нажатии на них левой клавишей мыши в поле устанавливается курсор и есть возможность ввода текста с клавиатуры (области 1, Рисунок 18). Поля, недоступные для редактирования с клавиатуры (области 2, Рисунок 18), отражены светло-серым цветом, не активны при нажатии на них левой клавишей

мыши, курсор в них не устанавливается, а появляется подсказка "*Выберите значение*". Такие поля заполняются автоматически.

При установке курсора на поле ввода можно узнать, какой редактор доступен для ввода значения.

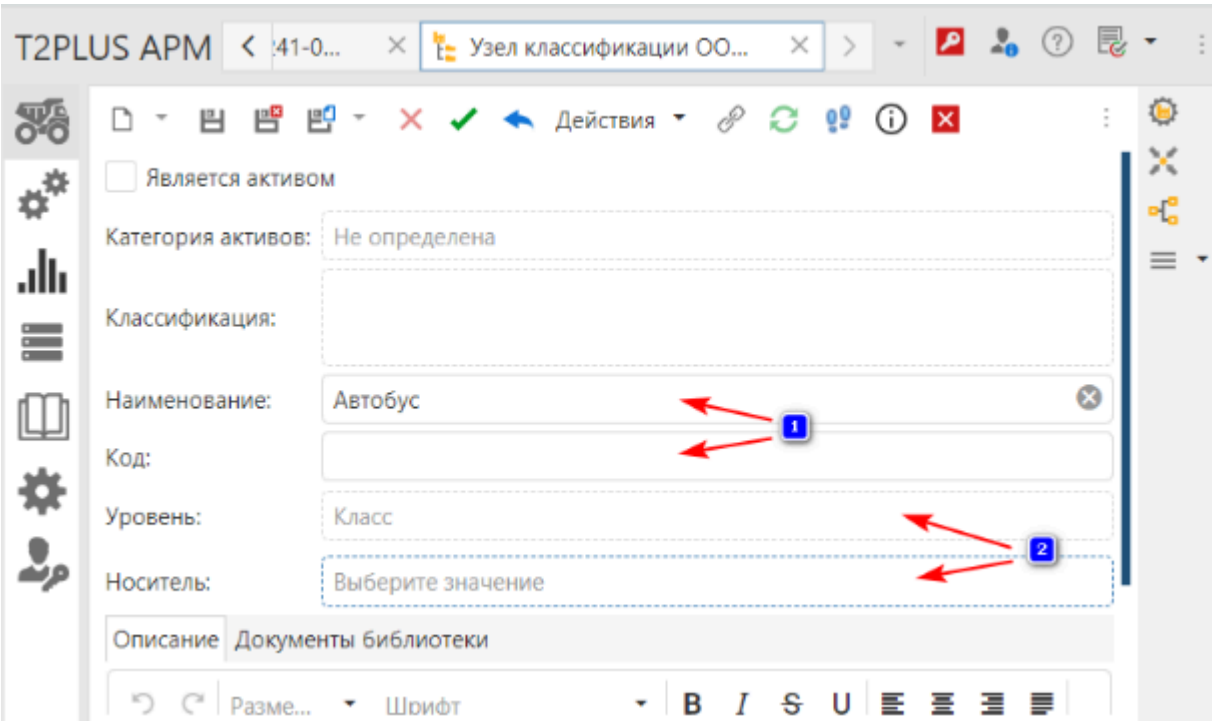


Рисунок 18. Редактируемые и нередактируемые поля

Текстовый редактор

В простой текстовый редактор текст вводится с клавиатуры. Удаляется текст клавишей Backspace.

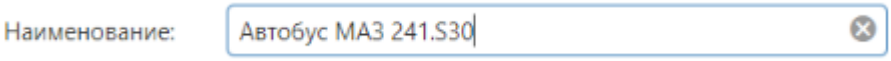


Рисунок 19. Текстовый редактор

Редактор выбора объектов из справочника

В тех случаях, когда ввод записей предопределен справочником, выбор может осуществляться описываемыми ниже способами.

Поиск в строке авто-фильтра

Для быстрого поиска необходимо выделить область ввода (область 1 на рисунке ниже) и начать вводить текст. При вводе критерия поиска автоматически фильтруются записи в списке. По умолчанию при использовании строки авто-фильтра применятся операнд поиска "*Начинается с*".

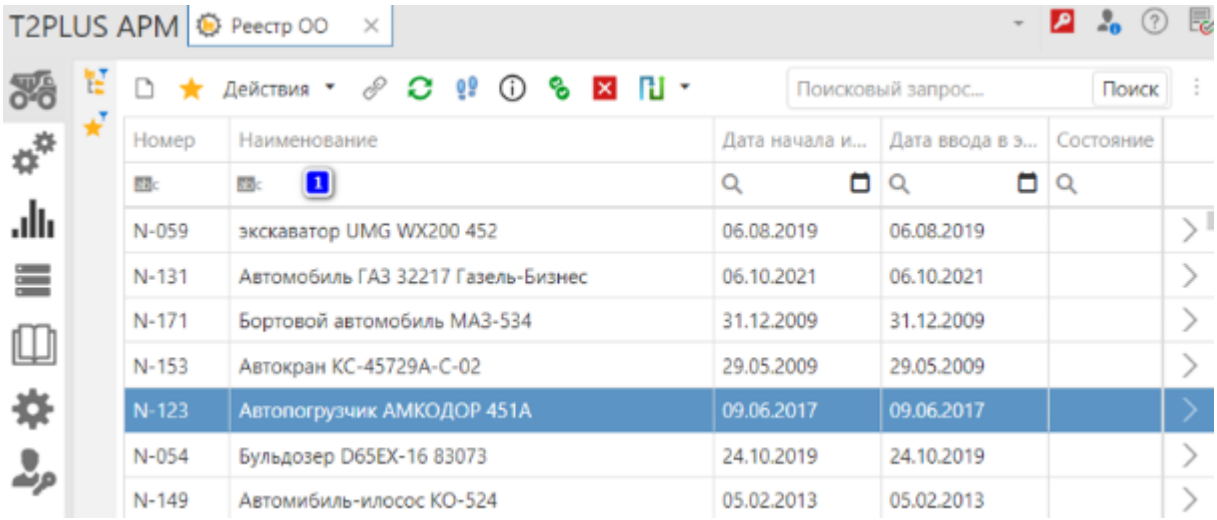


Рисунок 20. Поле ввода значений для поиска

Значения, подходящие под критерии поиска, система выведет на экран.

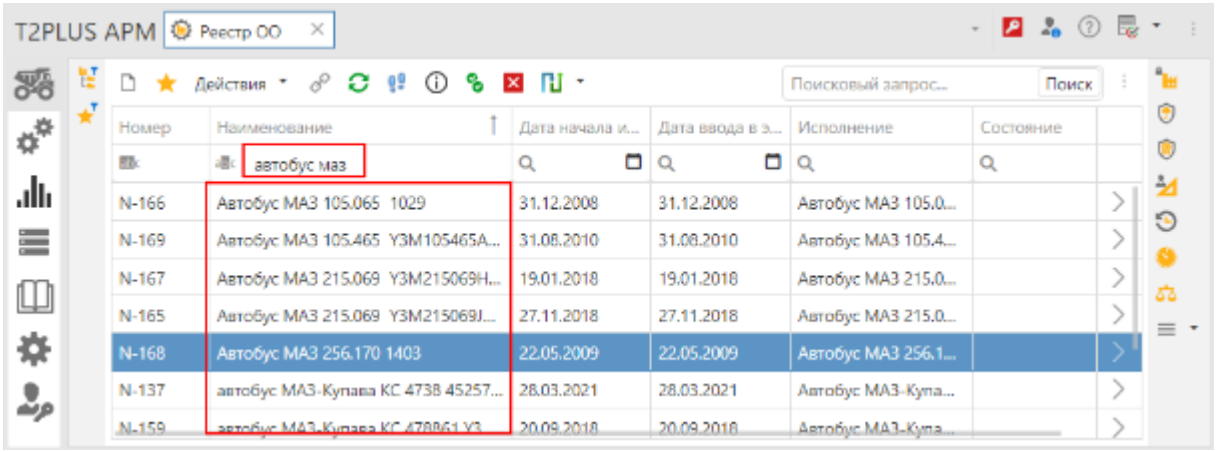


Рисунок 21. Пример быстрого поиска объекта обслуживания по наименованию

Для удаления результата поиска необходимо удалить вводимые данные из строки клавишей Backspace.

Открыть карточку требуемого объекта обслуживания можно двойным щелчком левой кнопки мыши или наведя курсор на нужную строку и нажав клавишу Enter.

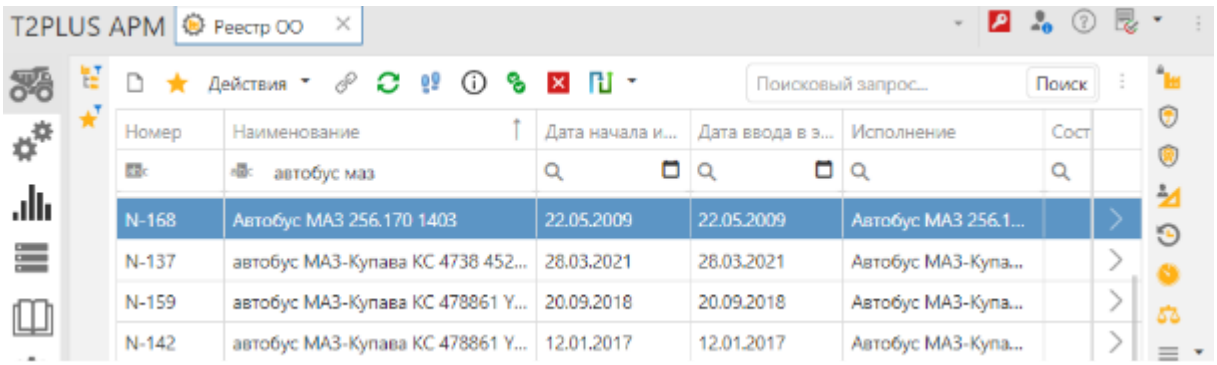


Рисунок 22. Выбор искомого объекта обслуживания

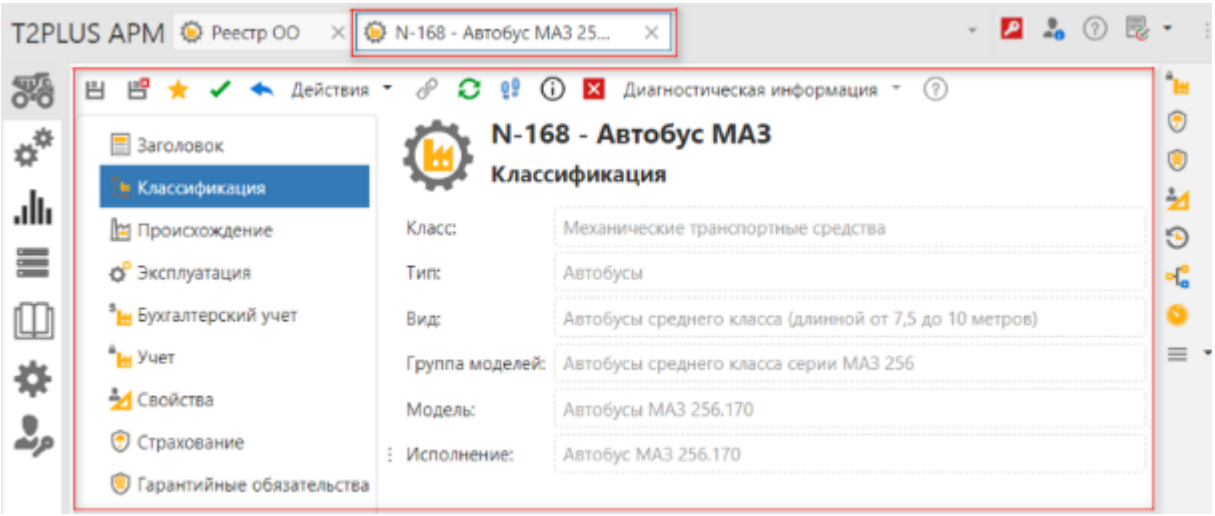


Рисунок 23. Открытие карточки объекта

Строка поиска по тексту

Поиск объектов можно осуществлять и с помощью строки *Поисковый запрос* в правом верхнем углу интерфейса системы.

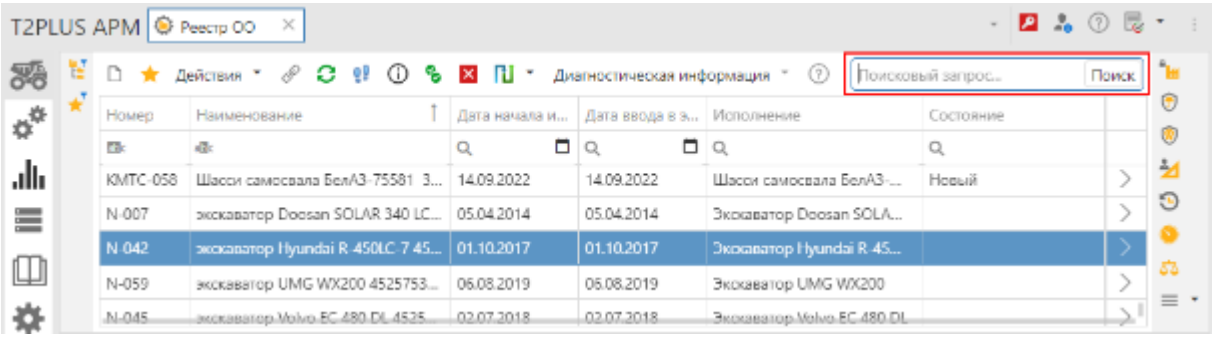


Рисунок 24. Строка поиска записей по тексту

Для нахождения требуемого элемента следует ввести в строку поиска часть наименования или номера объекта, затем нажать кнопку **Поиск** или клавишу Enter на клавиатуре.

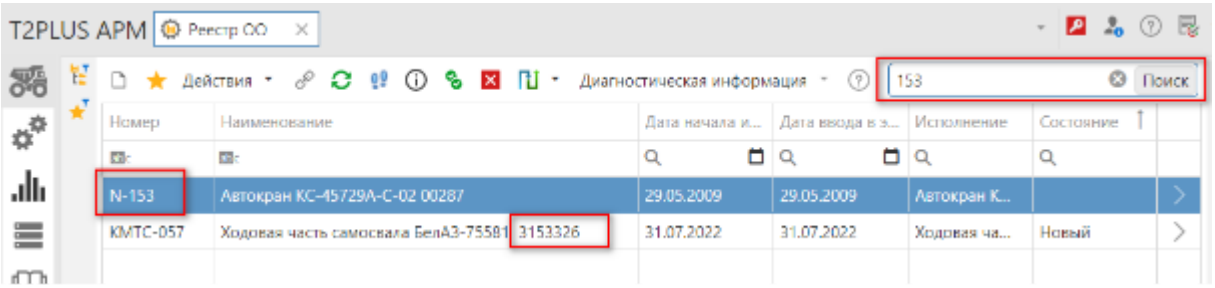


Рисунок 25. Результат поиска записей по тексту

Быстрый поиск по главному меню системы

Если необходимо быстро найти определенный пункт главного меню системы, можно использовать строку поиска по главному меню. Данная строка отражается в правом верхнем углу интерфейса окна браузера при условии, что пользователь зашел в один из модулей системы.

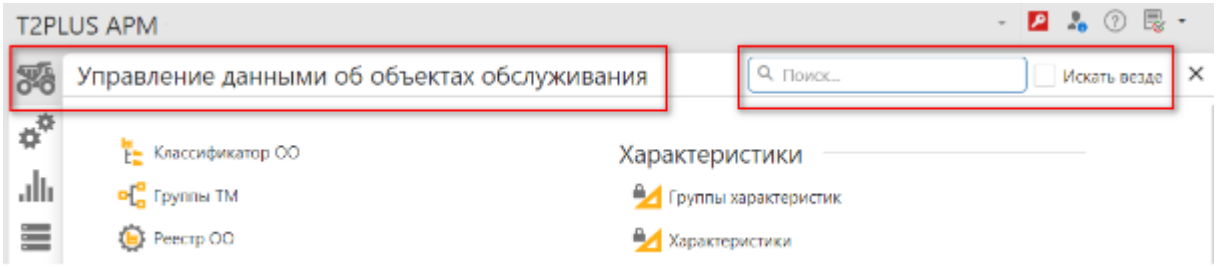


Рисунок 26. Поиск по главному меню

Чтобы найти определенный подпункт главного меню, начнем вводить его название в строку поиска с клавиатуры и увидим найденные результаты в рабочей области окна. Если установить параметр **Искать везде**, система выдаст совпадения по всем модулям, а не только по тому, в котором пользователь находится в данный момент.

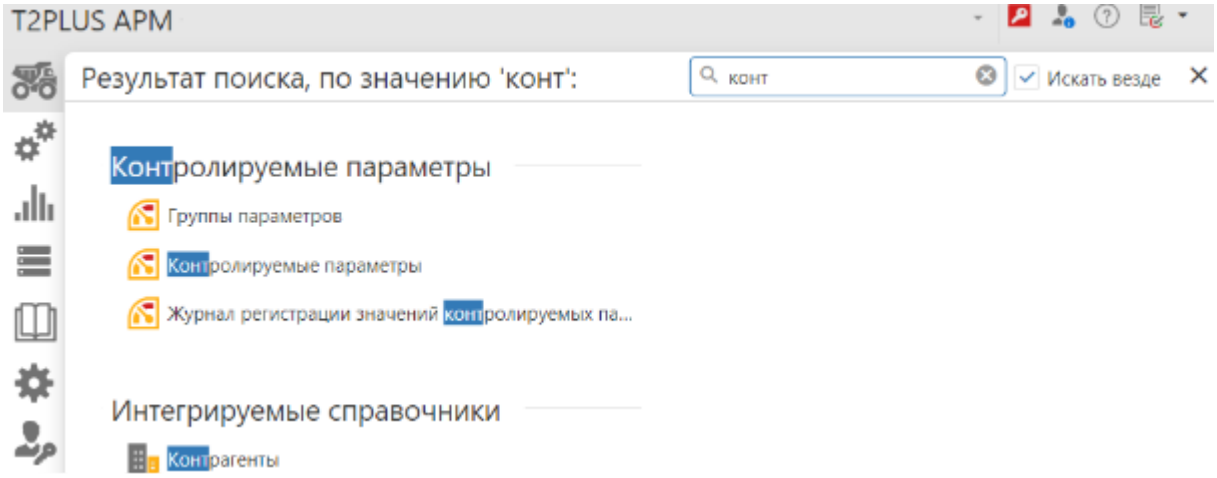


Рисунок 27. Пример поиска по главному меню

Статусная схема

Для удобства работы в системе используются следующие статусы:

- *Черновик*: назначаемый системой автоматически начальный статус при создании записи.
- *Опубликовано*: присваиваемый пользователем при выполнении определенных условий статус: полнота заполнения полей, уникальность и т. п. Возврат в статус *Черновик* возможен, только если сущность не выбрана (не используется) в другом объекте системы.
- *Архив*: конечный статус, присваиваемый пользователем с определенными правами. Запрет использования сущности при создании новых объектов системы.

В таблице ниже приведено описание ограничений на доступные действия в зависимости от статуса.

Наименование	Пиктограмма	Ограничения состояния на доступные действия			
		Редактирование полей записи	Использование при создании новых записей	Использование в ранее созданных записях	Удаление записи
Черновик			Запрещено	Отсутствует	
Опубликовано		Запрещено			Запрещен
Архив		Запрещено	Запрещено		