

ДИСЦИПЛИНА	Программные средства имитационного моделирования систем (полное наименование дисциплины без сокращений)
ИНСТИТУТ	ИТ
КАФЕДРА	Прикладной математики полное наименование кафедры)
ВИД УЧЕБНОГО МАТЕРИАЛА	Практики (в соответствии с пп.1-11)
ПРЕПОДАВАТЕЛЬ	Есипов Иван Владимирович (фамилия, имя, отчество)
СЕМЕСТР	7, 2024-2025 (указать семестр обучения, учебный год)



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное

Учреждение высшего образования

МИРЭА – Российский технологический университет

Институт Информационных Технологий

Кафедра Прикладной математики

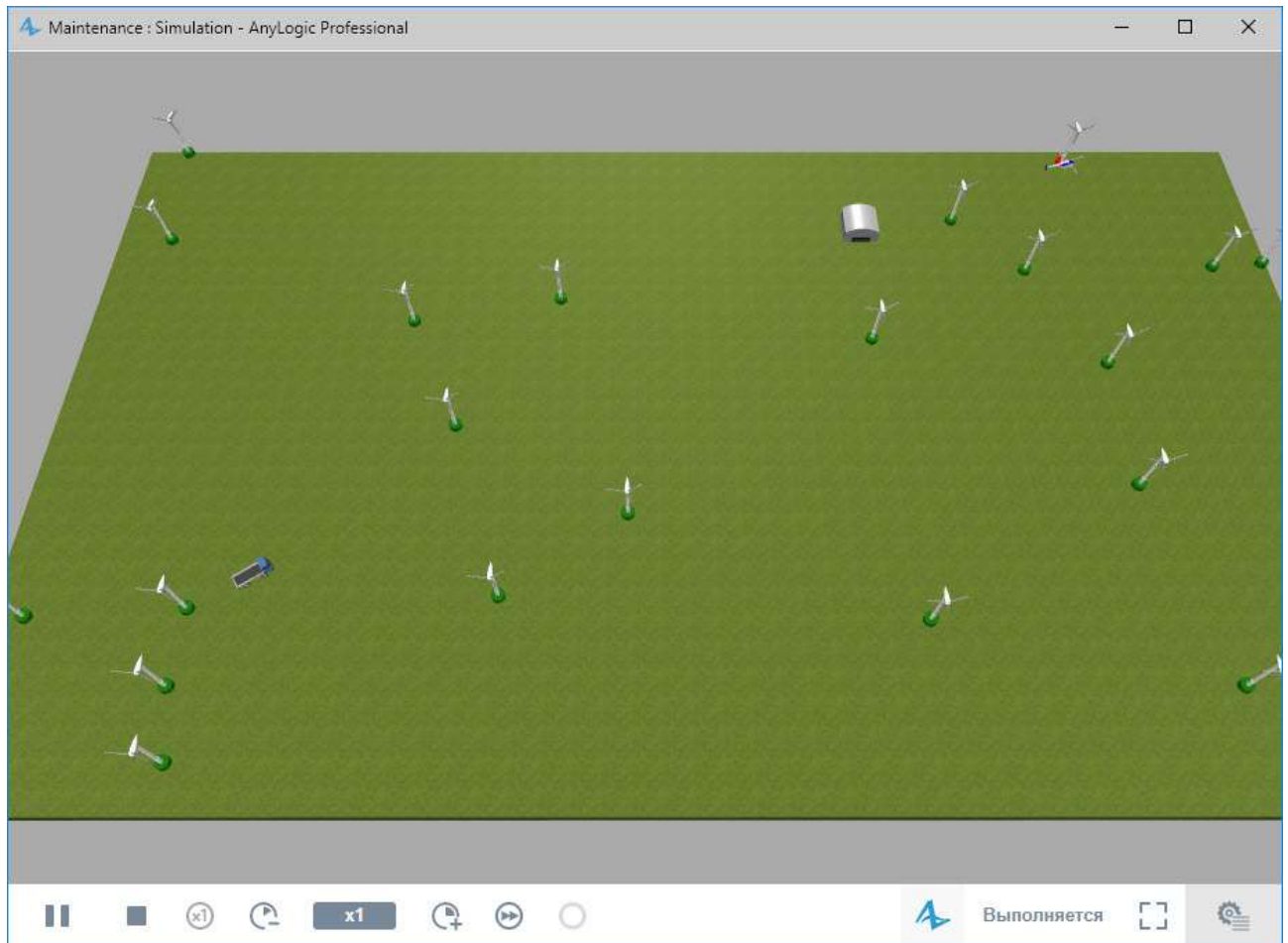
Практическая работа №7

Тема практической работы

«Модель обслуживания ветряных турбин»

Москва 2024

Давайте промоделируем, как сервисный центр производит обслуживание ветряных турбин. Имеется десять ветряных турбин, расположенных случайным образом в непрерывном пространстве, которые требуют обслуживания.



Мы будем различать два вида сервисных работ:

1. **Периодически проводимое техническое обслуживание**
 - ТО должно проводиться каждые две недели.
 - Сервисная бригада выезжает к турбине на грузовике.
 - Время проведения работ равно 10 часам.
2. **Срочное устранение поломок**
 - Среднее время между поломками — 50 дней.
 - Сервисная бригада вылетает к турбине на вертолете.
 - Время устранения поломки равномерно распределено от 10 до 20 часов.

Ветряные турбины обслуживаются одним сервисным центром. Центр владеет парком транспортных средств, состоящим из двух вертолетов и пяти грузовиков.

Промоделируем один год. При желании вы можете учесть затраты по видам деятельности, добавить стоимость бригад, логику замены деталей деталями различной стоимости и т.д.

Шаг 1. Создание различных типов агентов

Под агентом в агентном моделировании понимается элемент модели, который может иметь поведение, память (историю), контакты и т.д. Агенты могут моделировать людей, компании, проекты, автомобили, города, животных, корабли, товары и т.д.

Вы можете создавать внутри агента переменные, диаграммы состояний, задавать события, потоковые диаграммы системной динамики, а также добавлять внутрь агента объекты библиотек AnyLogic. Вы можете создать в одной модели столько типов агентов, сколько разных типов агентов вам нужно промоделировать.


Создание агента обычно начинается с определения его интерфейса для связи с внешним миром. В случае систем с большим количеством агентов с динамическими связями (например, в моделях социальных сетей) агенты могут взаимодействовать друг с другом путем вызова методов друг у друга.

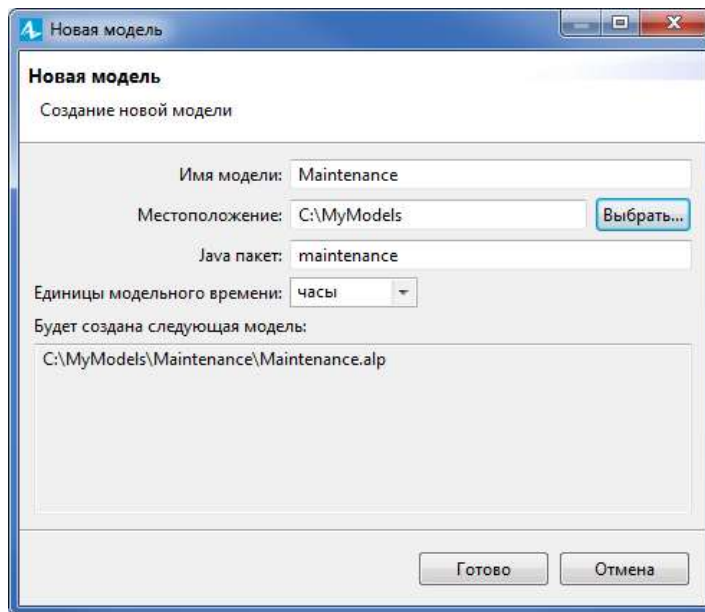
Начальное состояние и поведение агента могут быть реализованы различными способами. Состояние (накопленная история) агента может быть представлено с помощью переменных, либо состояния диаграммы состояний. Поведение может быть либо пассивным (агенты реагируют только на прибытие сообщений или на вызов методов и не имеют собственных событий, запланированных на будущее) или активным, когда внутренняя динамика агента (события, запланированные через заданные таймауты или процессы системной динамики) является причиной действий, совершаемых агентом. В последнем случае внутри агентов скорее всего должны быть заданы события и/или диаграммы состояний.

В нашей модели мы будем использовать [параметры](#), [переменные](#), [коллекции](#), [функции](#), [события](#), [списки вариантов](#), а также [диаграммы состояний](#).



Создание модели


Создайте новую модель

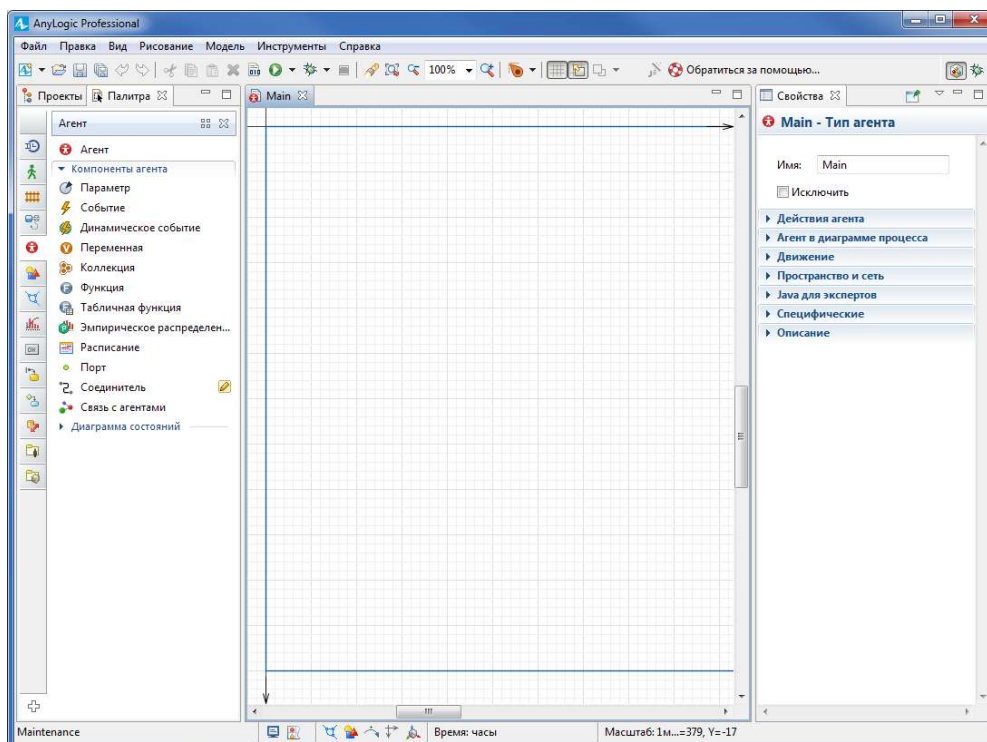
1. Щелкните по кнопке **Создать**  на панели управления. Откроется диалоговое окно **Новая модель**.
2. Задайте имя модели. Введите `Maintenance` в поле **Имя модели**.



3. Укажите местоположение, где вы хотите хранить файлы модели. Используйте кнопку **Выбрать...**, чтобы выбрать нужную папку, или введите путь к папке в поле **Местоположение**.
4. В качестве **Единиц модельного времени** выберите **часы**.
5. Щелкните **Готово**.

Будет создана новая модель. AnyLogic автоматически создаст тип агента  **Main** и простой эксперимент  **Simulation**.

В центре рабочего пространства вы увидите графический редактор. Он отображает диаграмму типа агента  **Main**. **Рамка** синего цвета задает область диаграммы, которая будет отображена в **окне модели** при ее запуске (а также размеры этого окна).



Слева от графического редактора вы можете видеть панель **Проекты**, объединенную с панелью **Палитра**. Панель **Проекты** обеспечивает доступ к моделям AnyLogic, открытым в данный момент в рабочем пространстве. Дерево элементов модели позволяет легко ориентироваться в ее структуре. Панель **Палитра** содержит все графические элементы, которые вы можете добавлять на диаграмму типа агента, просто перетаскивая их в графический редактор. Элементы сгруппированы в отдельные палитры.

Справа вы можете найти панель **Свойства**. Панель **Свойства** отображает и позволяет изменять свойства выбранного в данный момент элемента (группы элементов) модели. Когда вы выделяете какой-либо элемент, например, в панели **Проекты** или в графическом редакторе, панель **Свойства** отображает свойства выделенного элемента.

Вы можете открывать и закрывать конкретные панели с помощью меню **Вид**.

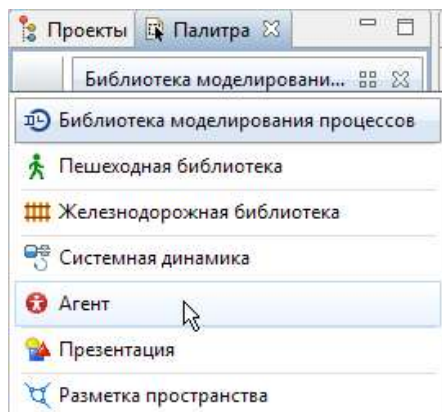
Теперь начнем разрабатывать нашу модель.

Создание агентов

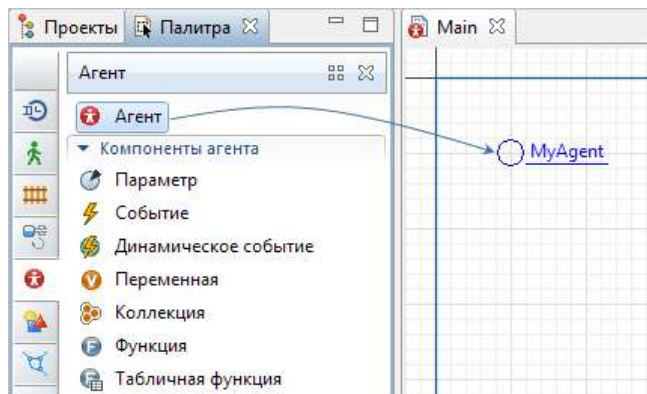
Вначале создадим одного агента - сервисный центр, затем пустую популяцию для обслуживающего транспорта, три популяции для моделирования турбин, грузовиков и вертолетов, а также тип агента, представляющего запросы на обслуживание. Далее мы сможем задать процессы внутри типов агентов - на их диаграммах.

Добавьте сервисный центр

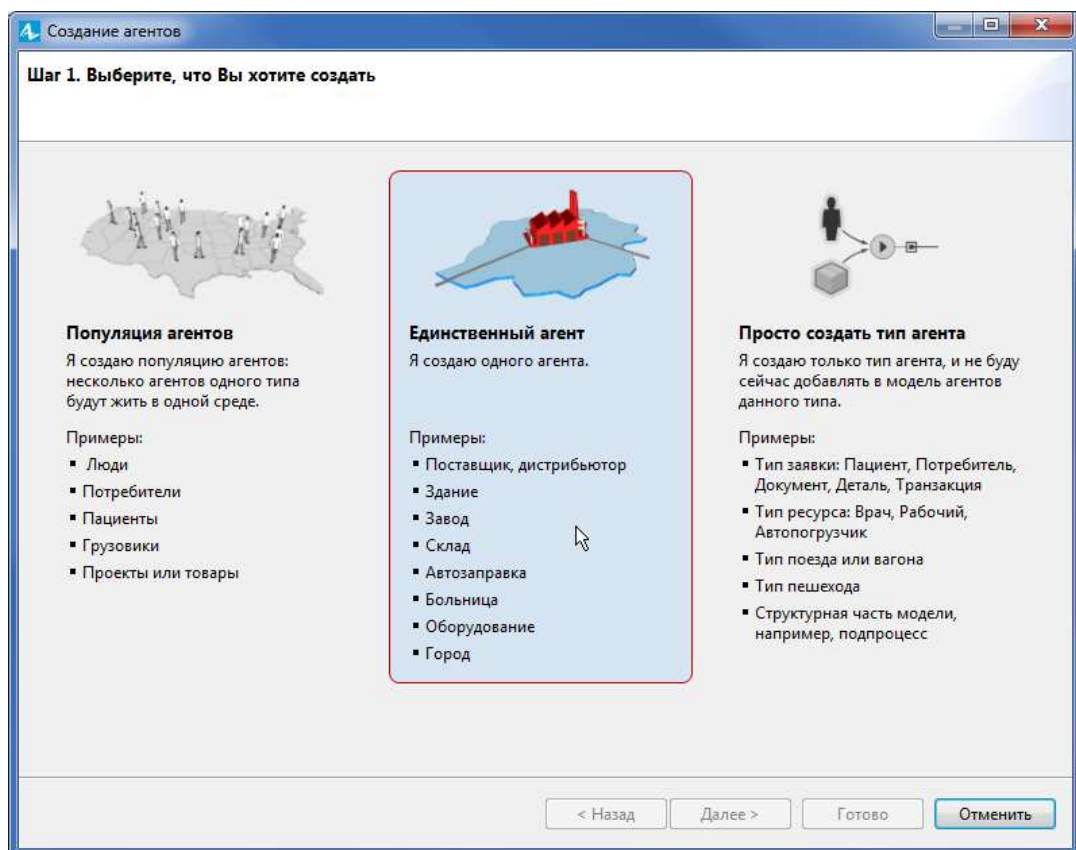
1. В панели **Палитра** наведите мышь на вертикальную полосу навигации (она располагается по левому краю панели), и выберите палитру **Агент**.



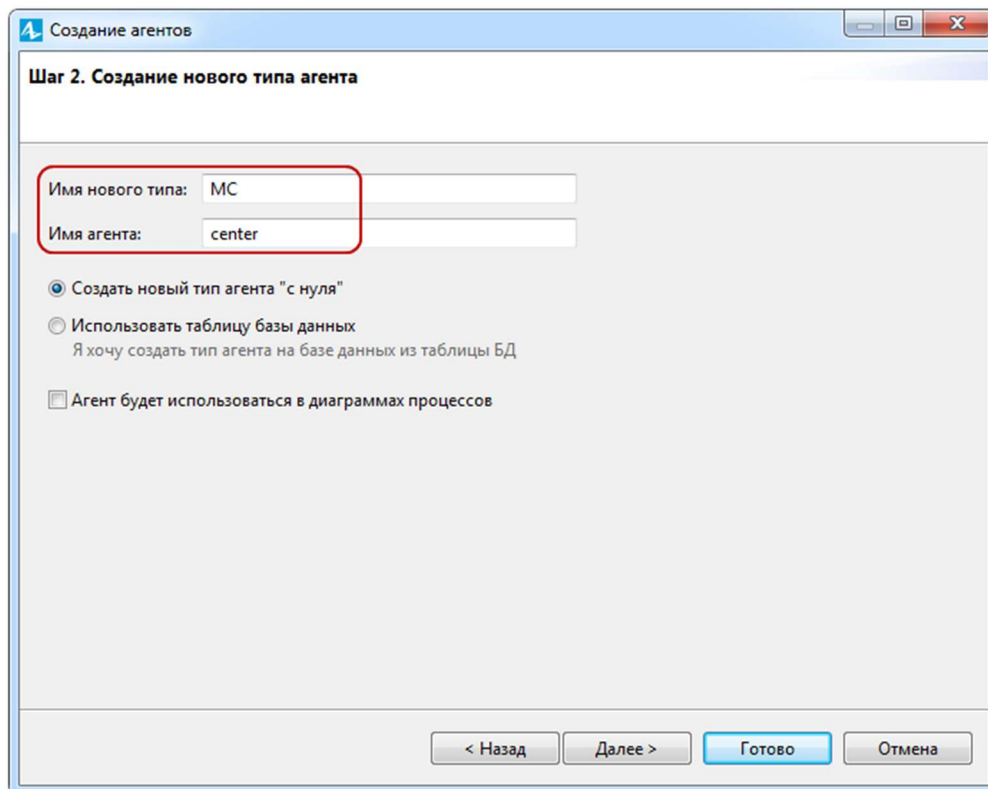
2. Перетащите элемент **Агент** (с иконкой человека) из палитры на диаграмму типа агента (с иконкой человека) **Main**.
Окно мастера **Создание агентов** откроется автоматически.



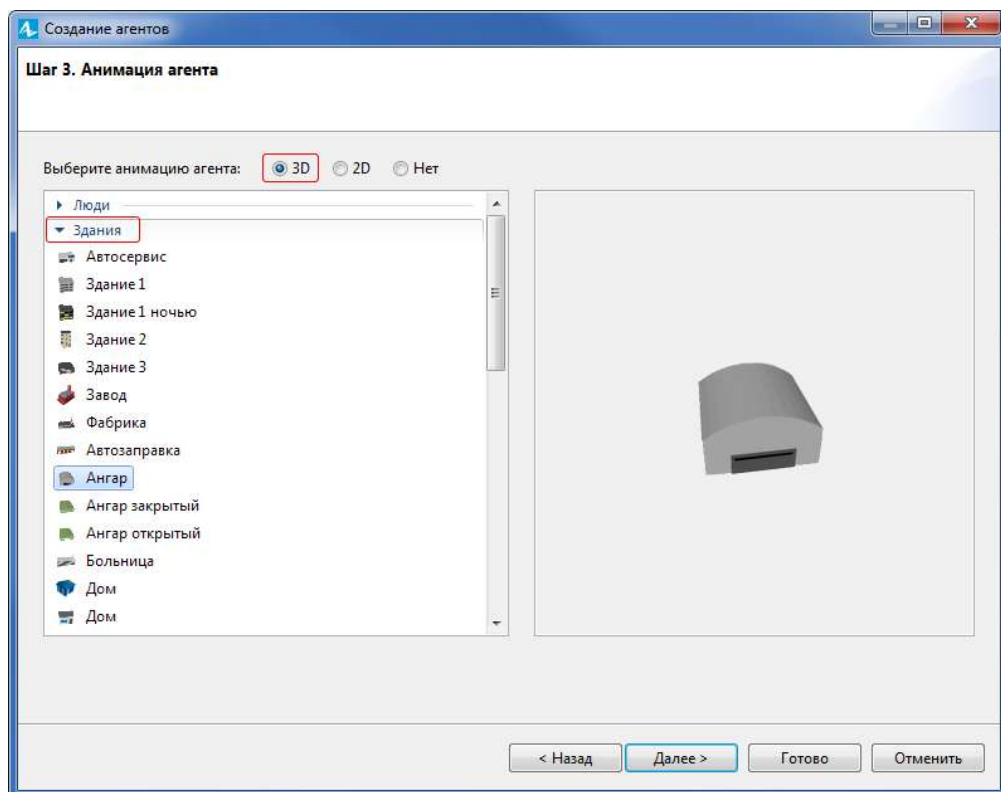
3. Выберите опцию **Единственный агент** на первом шаге мастера. Нам нужно создать только один сервисный центр, который будет отправлять транспорт к турбинам по запросам на плановое обслуживание или при авариях.




4. Мы не будем использовать данные из базы данных, поэтому оставьте выбранным опцию **Создать новый тип агента "с нуля"**. Задайте **Имя нового типа**: МС и имя самого агента center в поле ниже. Щелкните **Далее**, чтобы продолжить.

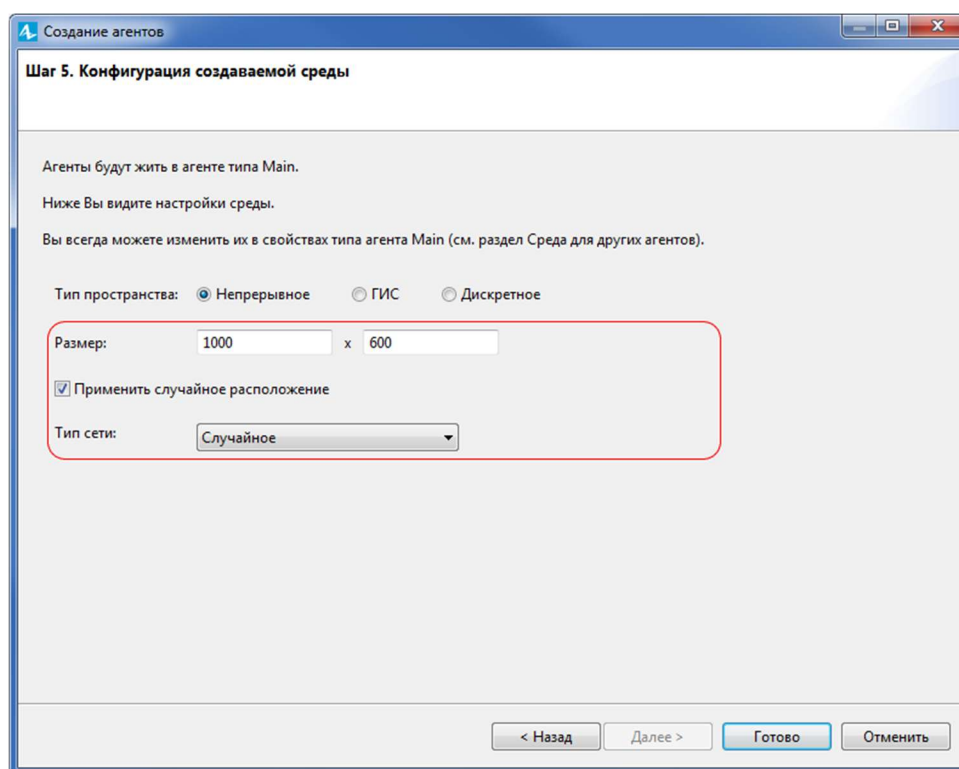


5. Выберите тип анимации **3D**, затем фигуру анимации **Ангар** из секции **Здания** и щелкните **Далее**.








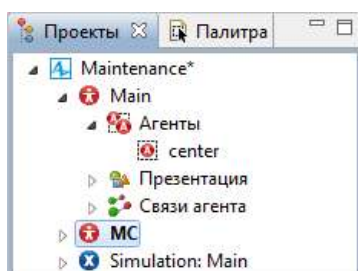
6. Пропустите четвертый шаг по добавлению параметров, просто щелкните **Далее**.
7. Мы хотим, чтобы все наши агенты "жили" в непрерывном пространстве, размерами 1000x600, в сети со случайным расположением агентов.


Популяции агентов, которые мы позже так же добавим на диаграмму  Main, будут жить в той же среде.




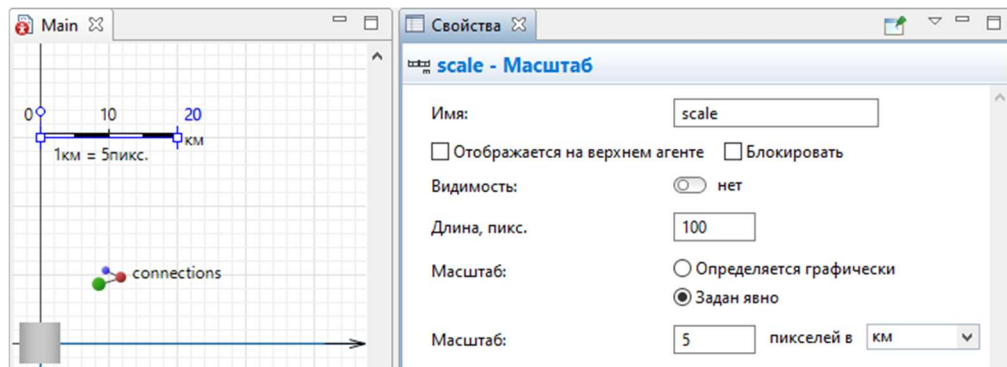
8. Щелкните **Готово**.

После создания нового типа агента , вы можете найти его в дереве модели на том же уровне, что и тип агента  Main, а сам агент  **center** находится в ветке  **Main** >  **Агенты**.



Фигура анимации агента отображается как на диаграмме его типа, так и на  Main. Нам необходимо изменить масштаб всей модели, а также масштаб отображения этой фигуры на анимации.

1. Чтобы изменить масштаб модели, сделайте двойной щелчок мышью по типу агента  Main в дереве элементов модели, чтобы открыть его диаграмму. Затем **передвиньте диаграмму** вниз, чтобы видеть элементы, находящиеся над осью X.
2. Здесь вы найдете объект **Масштаб**. Выделите его, чтобы открыть его свойства.
3. Установите **Масштаб** в режиме **Задан явно**, не изменяя длину линейки шкалы. В свойствах элемента, задайте **Масштаб**: 5 пикселей на 1 км.



Теперь давайте изменим масштаб фигуры анимации сервисного центра (ангара).

1. Сделайте двойной щелчок мышью по типу агента **МС** в дереве элементов модели, чтобы открыть его диаграмму. Найдите и выделите линейку масштаба этого типа агента.
2. Очевидно, что при выбранном масштабе фигура ангара будет отображаться на анимации модели такой крошечной, что мы не сможем ее разглядеть. Поэтому мы зададим для этого типа агента более крупный масштаб, чтобы мы могли легко увидеть его фигуру на сцене анимации нашей модели. В свойствах фигуры анимации ангара снимите флажок **Автоматически изменять размер для соответствия масштабу агента**.
3. Переключите опцию **Масштаб** в режим **Задан явно**. Ниже, задайте масштаб: **10 пикселей в км**.
4. Выберите фигуру анимации сервисного центра и в свойствах задайте **Доп. масштабирование: 75%**.

Добавьте ветряные турбины

1. Перетащите элемент **Агент** из палитры **Агент** на диаграмму типа агента **Main**.
2. В этот раз выберите опцию **Популяция агентов**. Щелкните **Далее** на шаге 2 (по умолчанию выбрана нужная нам опция **Я хочу создать новый тип агента**). На следующем шаге мастера создания агентов введите имя нового типа: **Turbine** и тогда автоматически появится имя популяции: **turbines**. Щелкните **Далее**.

Создание агентов

Шаг 2. Создание нового типа агента

Имя нового типа:

Имя агента:

☒ Создать новый тип агента "с нуля"
☐ Использовать таблицу базы данных
 Я хочу создать тип агента на базе данных из таблицы БД

☐ Агент будет использоваться в диаграммах процессов

3. Выберите **3D** фигуру анимации **Ветряная турбина** из секции списка объектов **Энергетика**.


Создание агентов

Шаг 4. Анимация агента

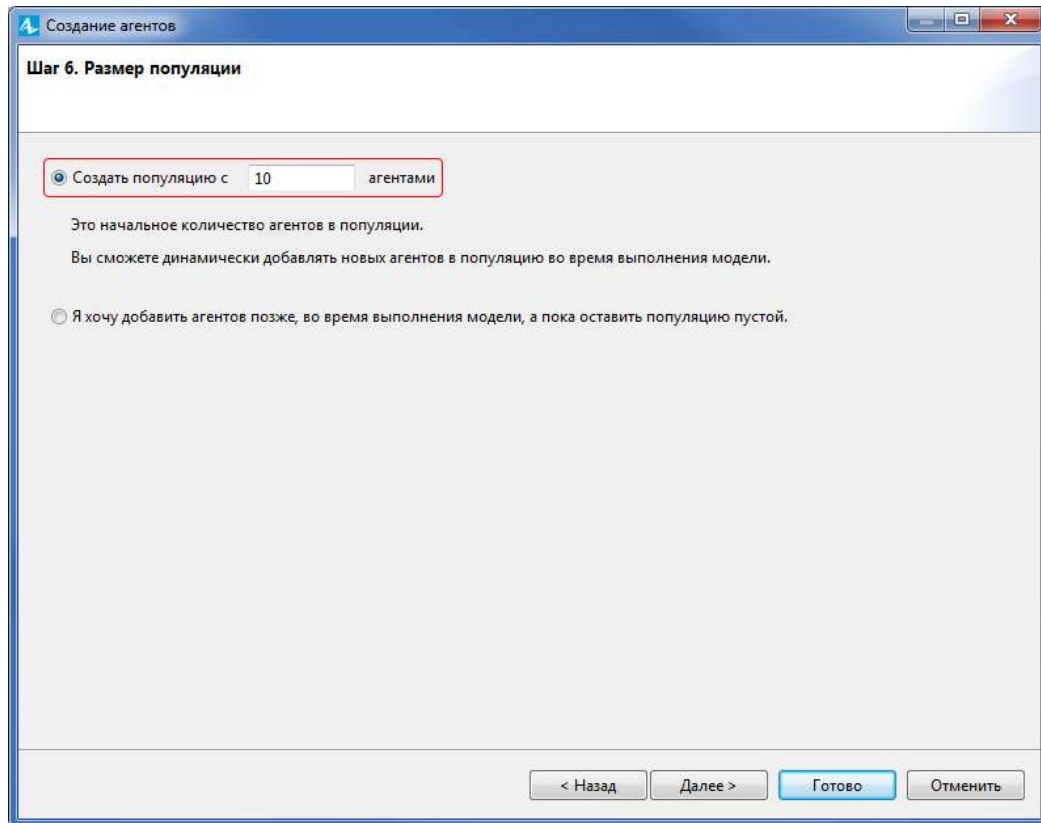
Выберите анимацию агента: ☒ 3D ☐ 2D ☐ Нет




Энергетика

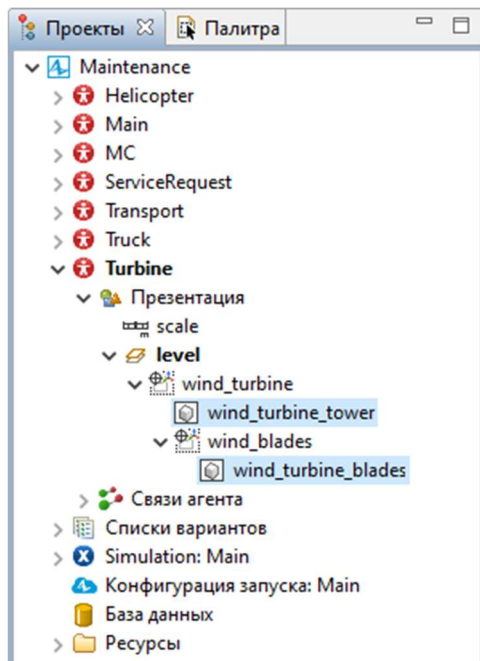
- Солнечная батарея
- Труба
- Градирня
- Градирня 2
- Генератор
- Нефтяной бур
- Нефтяной бур 2
- Платформа бура
- Ветряная турбина**
- Ветряная турбина Башня
- Ветряная турбина Лопасти
- Опора ЛЭП
- Опора ЛЭП 2
- Контроллер 1
- Контроллер 2



4. Нам не нужно сейчас создавать какие-либо параметры, так что пропустите шаг 5.
5. На шаге 6, задайте количество агентов, которые мы хотим видеть в этой популяции: 10 агентов.





5. Вы можете щелкнуть **Готово** уже на шаге 6. Настройки шага 7, **Конфигурация создаваемой среды**, будут автоматически заполнены, так как мы добавляем популяцию агентов в тип агента  Main, где мы уже задавали среду при создании агента сервисного центра.
6. Откройте диаграмму типа агента  Turbine и измените масштаб. Поскольку мы хотим, чтобы элементы нашей модели изображались на анимации чуть крупнее, и их можно было разглядеть, зададим чуть увеличенный масштаб. Пусть длина линейки соответствует 10 километрам, и масштаб будет равен **1 км = 10 пикс.**
7. Вы увеличили масштаб, и фигура анимации перестала отображаться на графическом редакторе. Это произошло потому, что в 3D объектах, входящих в группу, которой является эта фигура, по умолчанию выбрана опция **Автоматически изменять размер для соответствия масштабу агента**. Чтобы получить доступ к свойствам этих объектов, перейдите в панель **Проекты** и раскройте папку **Презентация** агента  Turbine.



8. В группе **wind_turbine** вы найдете два 3D объекта: **wind_turbine_tower** и **wind_turbine_blades**. В свойствах обоих объектов снимите флажок **Автоматически изменять размер для соответствия масштабу агента** и выберите в поле **Доп. масштабирование: 50%**.
9. Поскольку мы отключили автоматическое масштабирование, нам придется вручную настроить расположение элементов, составляющих группу **wind_turbine** относительно друг друга. Для этого выберите в дереве модели группу **wind_blades** и в поле свойств **Z** поместите следующее выражение:
`22.25 * Scale.DEFAULT_SCALE.pixelsPerUnit(METER)`

При вводе Java-кода пользуйтесь [Мастером подстановки кода](#), доступным по нажатию в поле ввода клавиш Ctrl + пробел (macOS: Alt + пробел). С его помощью вы узнаете список доступных функций и элементов модели и сможете просто выбрать нужное вам имя из списка, тем самым избежав возможных ошибок при написании имени.

Добавьте транспорт

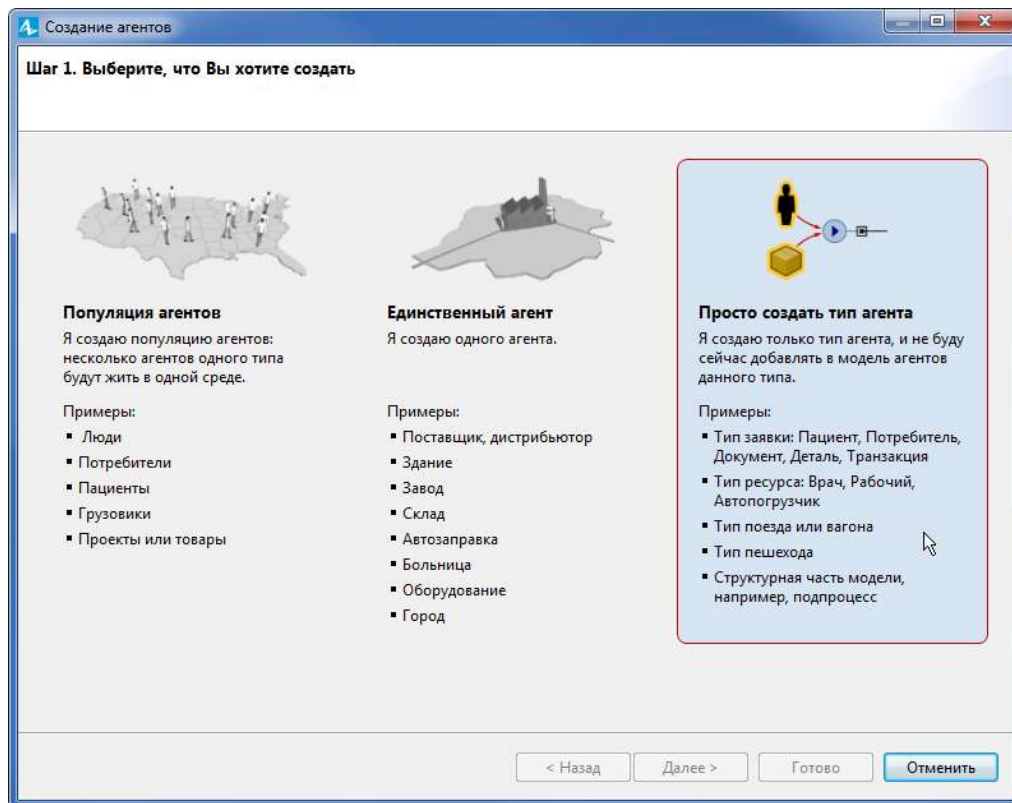
1. Перетащите элемент **Агент**  из палитры **Агент** на диаграмму типа агента  Main.
2. Снова выберите опцию **Популяция агентов**. Щелкните **Далее** на шаге 2 (**Я хочу создать новый тип агента**). На следующей странице мастера (шаг 3) введите в поле **Имя нового типа:** `Transport`, то же введите в поле **Имя популяции:** `transport`. Щелкните **Далее**.
3. Это пустая популяция, которой не нужна анимация. Мы будем использовать этот тип агента, чтобы задать логику модели. Выберите **Нет** для типа анимации, пропустите шаг создания параметров, выберите опцию **Я хочу добавить агентов позже, во время выполнения модели, а пока оставить популяцию пустой** и щелкните **Готово**.


Добавьте в модель грузовики и вертолеты

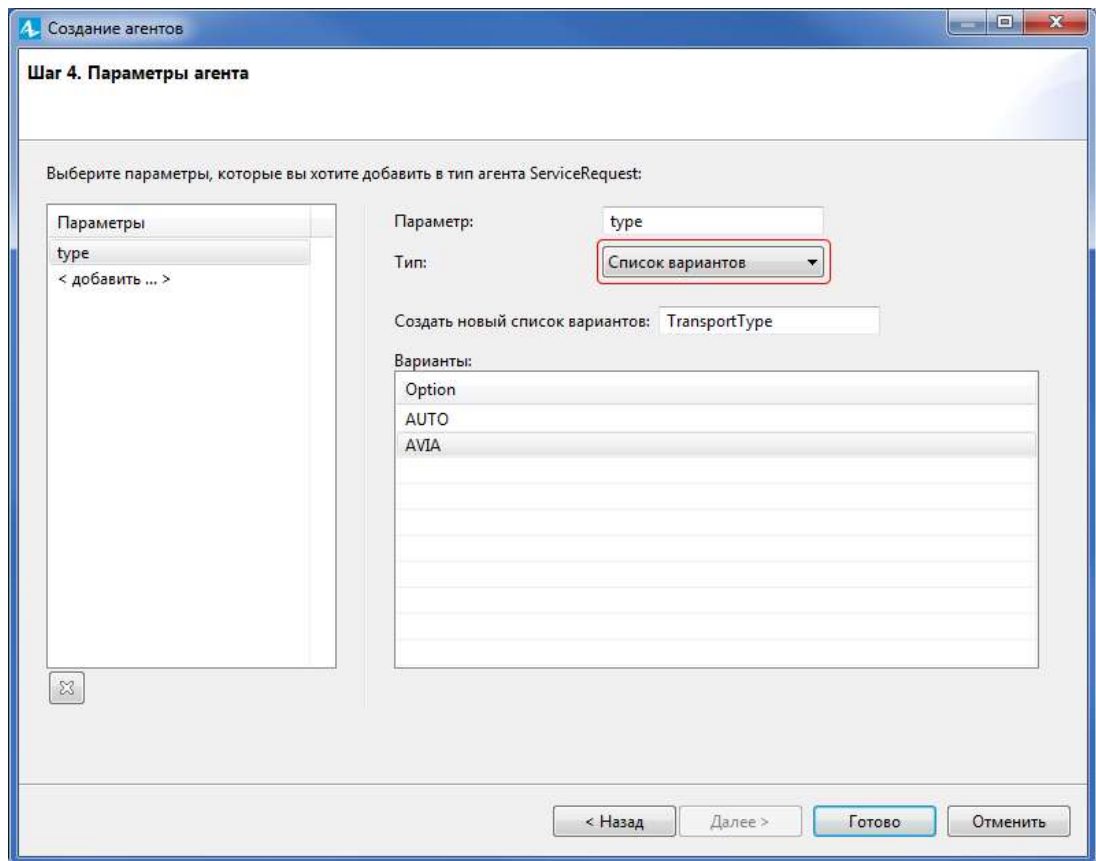
1. Перетащите элемент **Агент** из палитры **Агент** на диаграмму типа агента **Main**.
2. Снова выберите опцию **Популяция агентов**. Щелкните **Далее** на шаге 2 (**Я хочу создать новый тип агента**). На следующей странице мастера (шаг 3) введите в поле **Имя нового типа**: **Truck**, пусть имя популяции автоматически заполнится как **trucks**. Щелкните **Далее**.
3. Выберите фигуру **3D** анимации **Грузовик** из секции **Автодорожный транспорт** на следующем шаге. Нам не нужно добавлять сейчас параметры в мастере. Всего в популяции будет 5 агентов, а настройки среды уже будут заполнены.
4. После того, как создадите тип агента **Truck**, выделите **Truck** в дереве элементов модели и перейдите в панель **Свойства**. Сначала откройте секцию свойств **Размеры и движение**. Мы считаем, что грузовики в среднем движутся со скоростью **70 км в час**.
5. Откройте секцию свойств **Специфические** и укажите, что тип агента **Truck** **наследуется** от типа агента **Transport**. Для этого выберите в параметре **Расширяет тип агента**: **Transport**.
6. Зададим для грузовика чуть увеличенный масштаб. Для этого в свойствах элемента масштаб на диаграмме **Truck**, снимите флажок **Унаследовано от родительского класса**. Пусть длина линейки соответствует 10 километрам, и масштаб будет равен **1 км = 10 пикс**.
7. Теперь тип агента **Truck** полностью задан. Нам необходимо добавить еще одну популяцию для вертолетов. Вернитесь на **Main** и снова перетащите элемент **Агент** из палитры **Агент** и выберите **Популяция агентов** на первом шаге.
8. Введите имя нового типа **Helicopter**, оставьте имя популяции **helicopters**. Вы можете найти фигуру **3D** анимации **Вертолет** в секции **Военного назначения**. Мы хотим использовать 2 вертолета для обслуживания турбин. Эта популяция живет в той же среде, что и все остальные.
9. **Helicopter** также **Расширяет тип агента**: **Transport**. Мы предполагаем, что вертолеты движутся со скоростью, равной **200 км в час**.
10. Зададим для вертолета масштаб **1 км = 5 пикселей** (так же, как и чуть ранее, запретив наследование масштаба от родительского класса).

Добавьте запросы на обслуживание

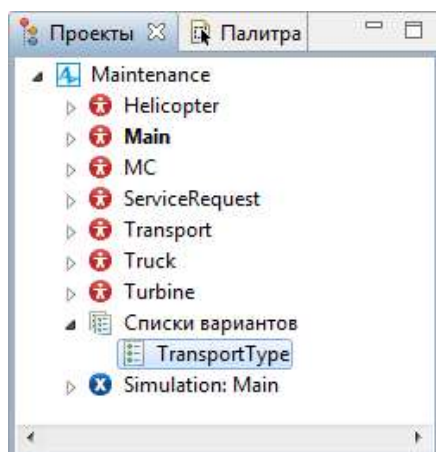
1. Перетащите элемент **Агент** из палитры **Агент** на диаграмму типа агента **Main**.
2. Выберите опцию **Просто создать тип агента**. Нам нужен тип агента, чтобы задать логику модели специальными параметрами, которые мы создадим на его диаграмме.






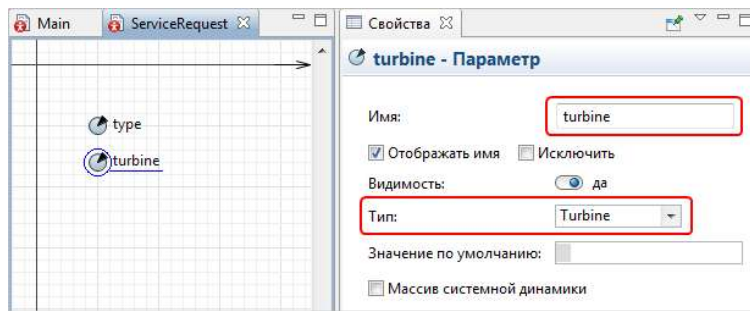
3. Введите имя типа `ServiceRequest` в соответствующем поле и щелкните **Далее**. Выберите **Нет** для типа анимации и перейдите на следующий шаг.
4. На шаге 4 мы создадим в мастере **параметр**, а также список вариантов.  **Список вариантов** - это элемент, который позволяет задавать параметры агента, которые имеют ограниченный выбор вариантов. В нашем случае, нам необходимо различать грузовики и вертолеты в общем транспортном парке. Назовите параметр `type` и выберите его **Тип: Список вариантов**. Так как в этой модели мы еще не создавали списков вариантов, по умолчанию мастер предложит вам **Создать новый список вариантов**: введите имя `TransportType`. Добавьте варианты в таблицу, по одному в каждой строке: `AUTO`, `AVIA`.




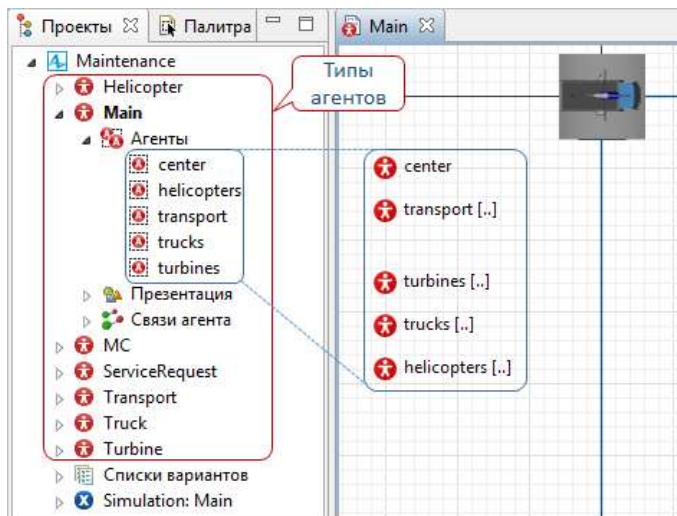
5. Щелкните **Готово**. Список вариантов не имеет отдельного значка, который можно выбрать щелчком в графическом редакторе. Вы можете найти секцию **Список вариантов** в дереве модели:




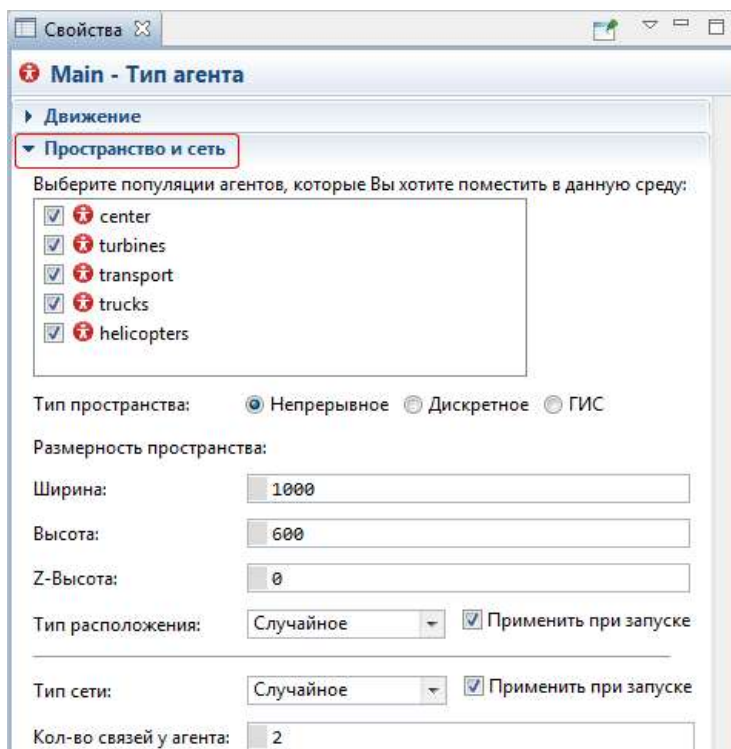
6. После того, как вы щелкните **Готово**, AnyLogic откроет диаграмму типа агента  **ServiceRequest**. Здесь вы можете увидеть параметр **type**, который мы создали в мастере.
7. Добавьте еще один **Параметр**  из палитры **Агент**. Назовите его **turbine** и выберите его **Тип** из выпадающего списка: **Turbine**. Теперь тип агента  **ServiceRequest** полностью задан.



Все агенты появятся на диаграмме  Main. Вы можете заметить, что популяции отмечены символами [...]. Переместите агентов за рамку окна презентации, а центры всех фигур анимации поместите в начало координат.

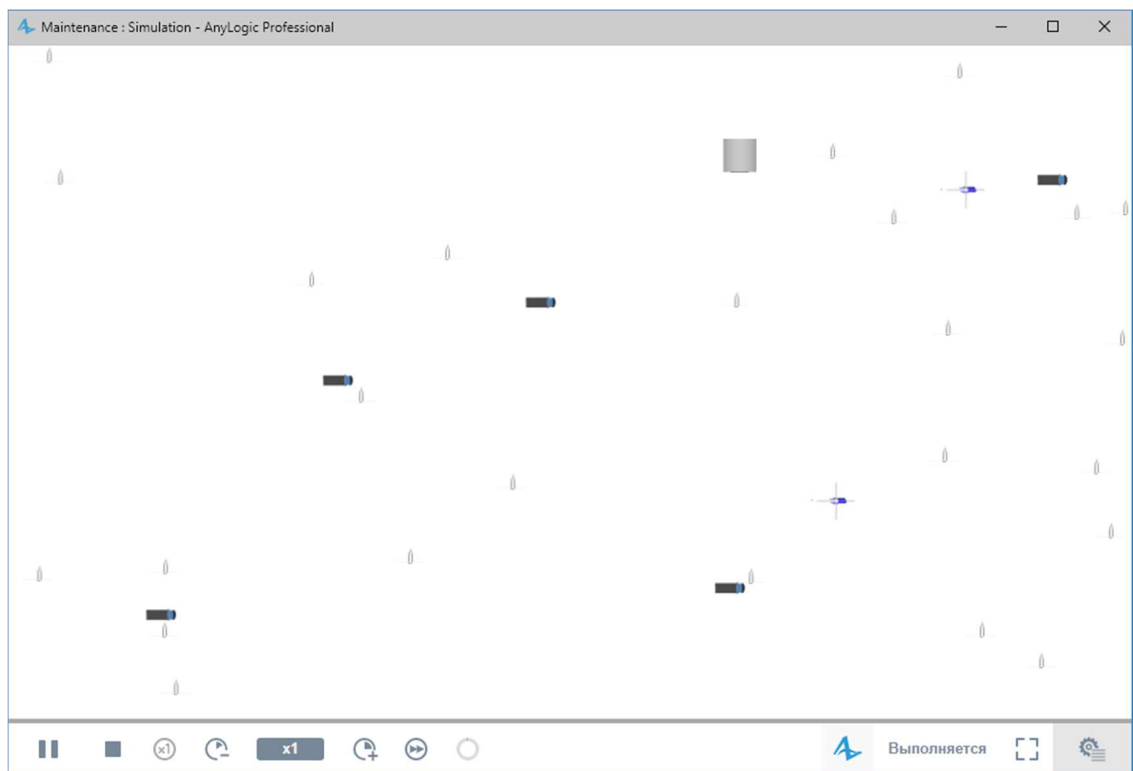


Вы можете найти настройки **среды** в секции **Пространство и сеть** свойств типа агента  Main:



Запустите модель

1. Постройте свой проект, щелкнув кнопку **Построить модель**  на панели управления. Если в модели есть какие-либо ошибки, то построение будет неудачным, и вы увидите панель **Ошибки**, в которой будут перечислены все ошибки в модели. Вы можете открыть место ошибки двойным щелчком по ее описанию в списке, чтобы исправить ее. После того, как модель будет успешно построена, вы можете запустить ее. Запуская эксперимент, вы автоматически обновляете модель.
2. Выберите эксперимент, который хотите запустить, открыв выпадающий список рядом с кнопкой запуска модели **Запуск**  на панели управления. Ваш эксперимент называется  Maintenance/Simulation. Потом вы сможете запускать этот эксперимент, просто щелкая кнопку **Запуск** , поскольку она будет запускать последний запущенный эксперимент.
3. Запустив модель, вы автоматически начнете ее выполнение и увидите окно презентации. На ней отображается презентация агента верхнего уровня модели. Мы только начали разрабатывать модель. В данный момент мы можем видеть сервисный центр, турбины и разные типы транспорта, случайно расположенные в непрерывном пространстве.



Далее давайте зададим местоположение транспорта, грузовиков и вертолетов в сервисном центре.

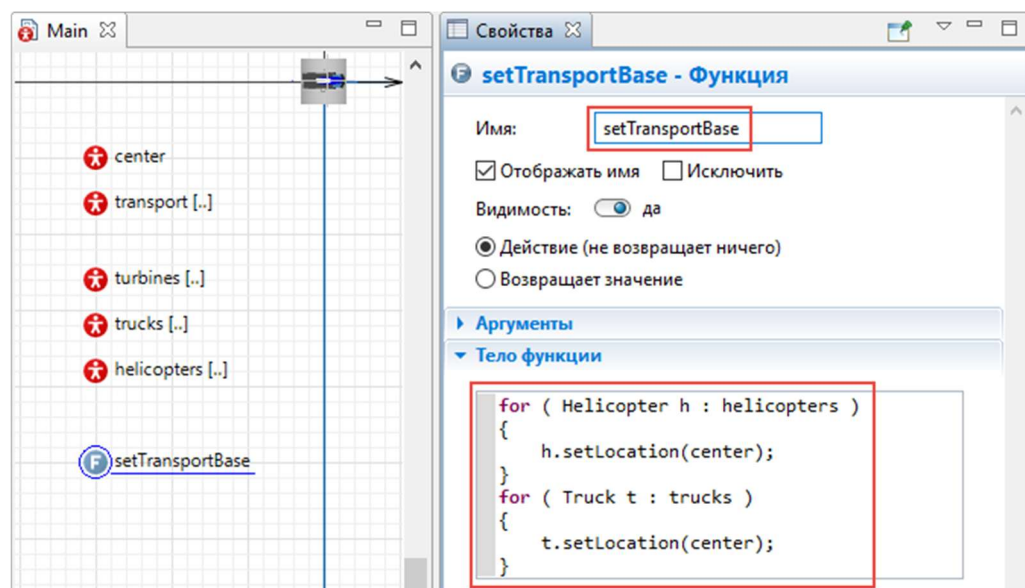
Шаг 2. Задание транспортной базы

Разрабатывая дальше модель, мы будем задавать логику и процессы модели на диаграммах каждого агента в отдельности. Для начала мы добавим алгоритм Java, с

помощью которого при запуске модели все наши грузовики и вертолеты будут размещены в сервисном центре.

Поместите транспорт в сервисный центр

1. Откройте диаграмму **Main** в графическом редакторе.
2. Перетащите элемент **Функция** из палитры **Агент** на диаграмму **Main**.
3. Назовите функцию `setTransportBase`. Эта функция ничего не возвращает, поэтому оставьте выбранной опцию **Действие (не возвращает ничего)**.
4. Разверните секцию свойств **Тело функции** и введите код Java как указано на изображении:



```
for (Helicopter h : helicopters)
{
    h.setLocation(center);
}
for (Truck t : trucks)
{
    t.setLocation(center);
}
```

Мы создали функцию, которая разместит транспорт в сервисном центре. Алгоритм этой функции содержит два [цикла for](#).

Первый цикл выполняет итерирование по всем агентам, входящим в популяцию вертолетов. В строке инициализации цикла `for` используется следующий синтаксис: в круглых скобках вы сначала указываете имя типа агентов, находящихся в популяции (`Helicopter`). Затем `h` — имя локальной переменной, которую мы задаем на этом участке кода. Вы можете использовать любое другое допустимое имя (`hel`, `a`, `item`, и так далее). После этого необходимо указать `helicopters` — имя популяции агентов, итерирование по которой мы будем выполнять.


Поскольку иногда мы хотим совершить не одно, а несколько действий с каждым агентом популяции, необходимо сказать компилятору Java, какие именно выражения Java должны выполняться во время каждой итерации цикла. Для этого мы помещаем

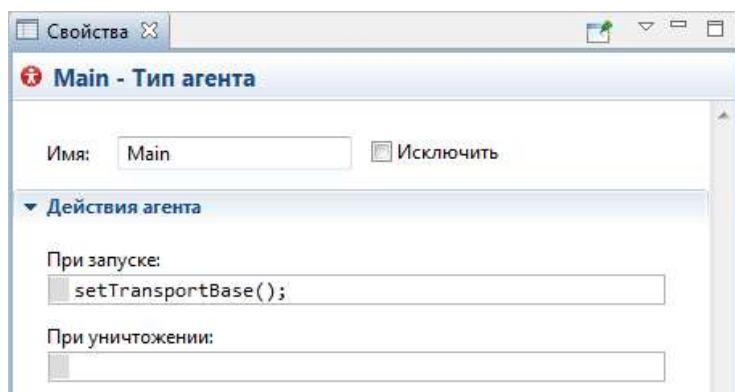
необходимые действия в фигурные скобки. В нашем случае это всего лишь одна строка кода: `{ h.setLocation(center) }`

С помощью этого кода мы выполняем следующее: мы задаем местоположение для агента популяции, по которому выполняется итерирование в данный момент, используя ранее заданную локальную переменную `h`. Местоположение (в нашем случае это сервисный центр `center`) передается в качестве аргумента функции `setLocation()`.

Второй цикл `for` выполняет то же самое для популяции `trucks`.

Теперь, чтобы функция заработала, ее нужно вызвать из кода.

Выделите  `Main` в дереве модели (или сделайте щелчок мышью в пустом месте графического редактора этого типа агента) и перейдите в панель **Свойства**. Разверните секцию свойств **Действия агента** и поместите вызов нашей функции в поле **При запуске**:




`setTransportBase();`

Снова запустите модель. Вы увидите, что все грузовики и вертолеты находятся в сервисном центре. Мы зададим движение транспорта и поведение других агентов на следующих этапах нашего учебного пособия.



Шаг 3. Настройка логических процессов транспорта

На этом этапе мы зададим начальную конфигурацию типа агента  Transport.



Если у агента можно выделить несколько состояний, выполняющих различные действия при происхождении каких-то событий, или если у агента есть несколько качественно различных поведений, последовательно сменяющих друг друга при происхождении определенных событий, то поведение такого объекта может быть описано в терминах [диаграммы состояний](#). Диаграмма состояний позволяет графически задать пространство состояний алгоритма поведения объекта, а также события, которые являются причинами срабатывания переходов из одних состояний в другие, и действия, происходящие при смене состояний.

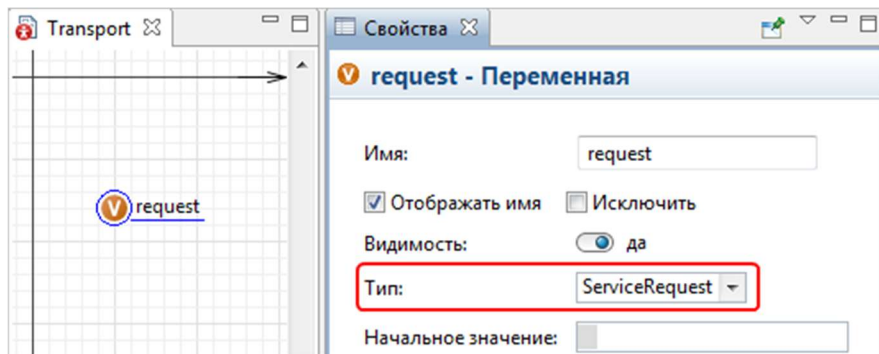
С помощью диаграмм состояний можно графически задать дискретные поведения объектов любой сложности, куда более разнообразные, чем элементарные состояния свободен/занят (idle/busy), открыт/закрыт (open/closed), исправен/неисправен (up/down) и т.п., предлагаемые большинством блочных инструментов моделирования.

Диаграмма состояний представляет собой состояния, соединенные переходами. Переходы могут сработать в результате заданного в качестве условия перехода события - это может быть истечение заданного таймаута, получение диаграммой состояний сообщения, выполнение заданного логического условия и т.д. Срабатывание перехода приводит к переходу управления диаграммы состояний в то

состояние, в которое ведет этот переход. Состояния могут быть иерархическими, т.е. содержать другие состояния и переходы.

Добавьте необходимые переменные

1. Перейдите в панель **Проекты** и откройте двойным щелчком тип агента  Transport.
2. Добавьте элемент **Переменная**  из палитры **Агент**.
3. Назовите переменную `request` и выберите в качестве ее типа `ServiceRequest`:

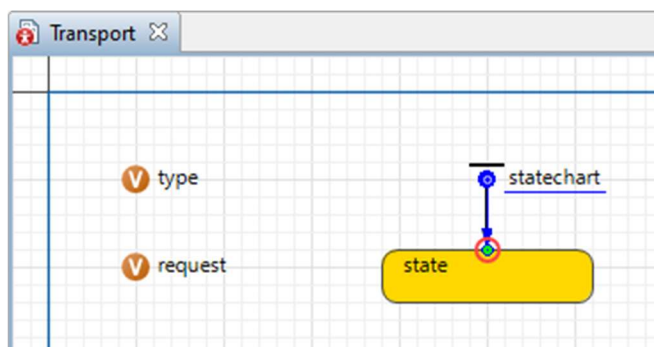


4. Таким же образом добавьте еще одну переменную. Назовите эту переменную `type` и в качестве ее типа выберите `TransportType`.

Нарисуйте диаграмму состояний транспорта

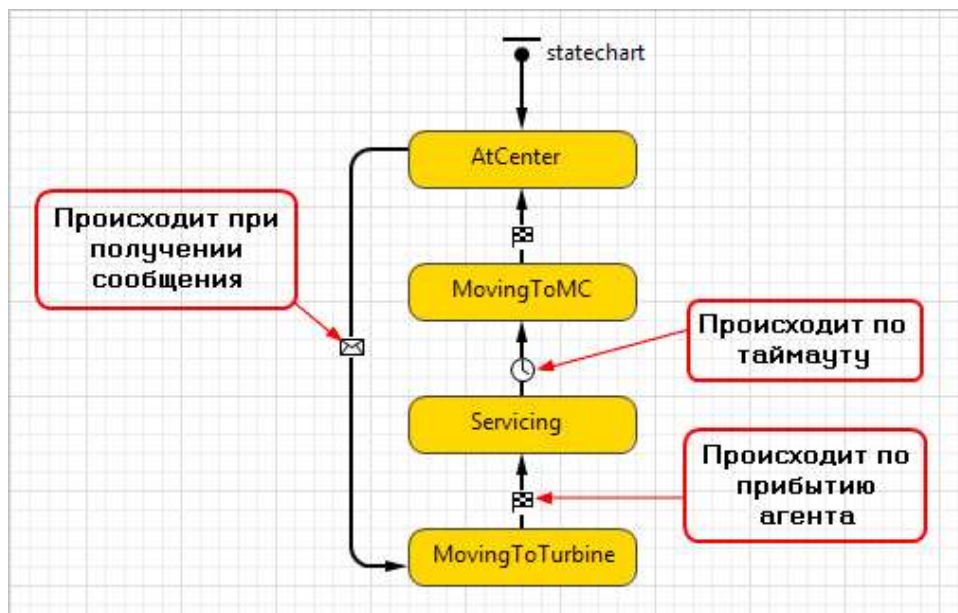
Мы используем элементы палитры **Диаграмма состояний**, чтобы задать поведение этого типа агента.

1. Перетащите элемент **Начало диаграммы состояний**  в графический редактор типа агента  Transport. Этот элемент задает начало всей диаграммы.
2. Добавьте **Состояние** , перетащив этот элемент из палитры **Диаграмма состояний**, и поместите его в конец стрелки начала диаграммы состояний.
3. Чтобы добавить переход, сделайте двойной щелчок по элементу **Переход**  в палитре. Его иконка поменяется на , это значит, что элемент теперь в режиме рисования. Вы можете рисовать переходы, делая щелчки по состояниям. Удостоверьтесь, что элементы соединяются друг с другом. Когда они соединены, AnyLogic отображает зеленую точку в месте соединения.



4. На этом шаге мы создадим "пустую" диаграмму состояний, которая состоит из четырех состояний, шести переходов разных типов и одного ветвления.
5. Состояния называются AtCenter (в сервисном центре), MovingToMC (движение к сервисному центру), Servicing (обслуживание турбины), MovingToTurbine (движение к турбине).

Это, собственно, все действия и передвижения грузовиков и вертолетов в нашей модели. Пожалуйста, следуйте структуре диаграммы, представленной на рисунке ниже.



6. Внутренние настройки и условия некоторых состояний и переходов должны будут ссылаться на элементы модели, которые мы еще не создали. С другой стороны, нам уже сейчас необходима эта диаграмма, поскольку на нее ссылаются некоторые из тех элементов, которые мы зададим позднее. Поэтому в следующих фазах мы продолжим разрабатывать другие типы агентов, а логику типа Transport закончим в самом конце разработки модели.

Давайте теперь зададим выбор типа транспортного средства в зависимости от запроса и доступность этих средств в сервисном центре.

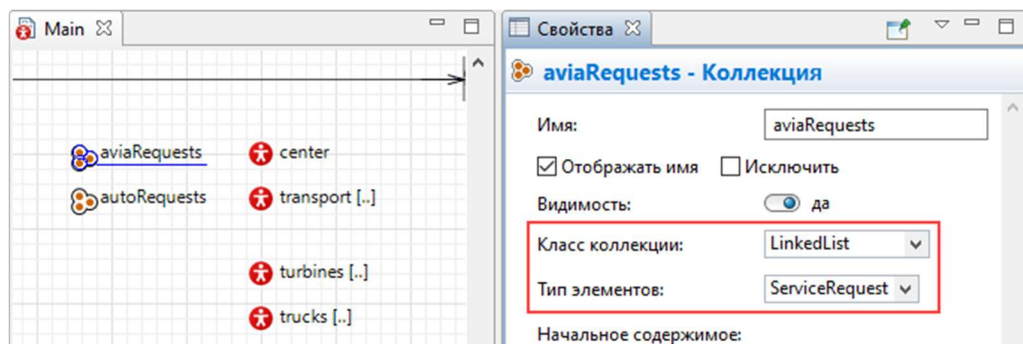
Шаг 4. Настройка поведения сервисного центра

Задайте запросы на транспорт для сервисного центра

Агент может содержать [переменные](#). Переменные обычно используются для моделирования изменяющихся характеристик объекта или для хранения результатов работы модели.

Коллекция представляет собой группу объектов, известных как ее элементы. Некоторые коллекции позволяют хранение нескольких одинаковых элементов, некоторые - нет. Некоторые коллекции упорядочены, некоторые - нет. Коллекция используется для задания объекта данных, объединяющего в себе сразу несколько однотипных элементов. С помощью коллекций вы можете хранить, извлекать и управлять агрегированными данными. Обычно коллекции представляют элементы данных, которые образуют группу, например, очередь (в этом случае элементы представляют собой людей, ожидающих в очереди) или автопарк (элементы задают автомобили), или телефонный справочник (коллекция хранит соответствие имен и телефонных номеров).

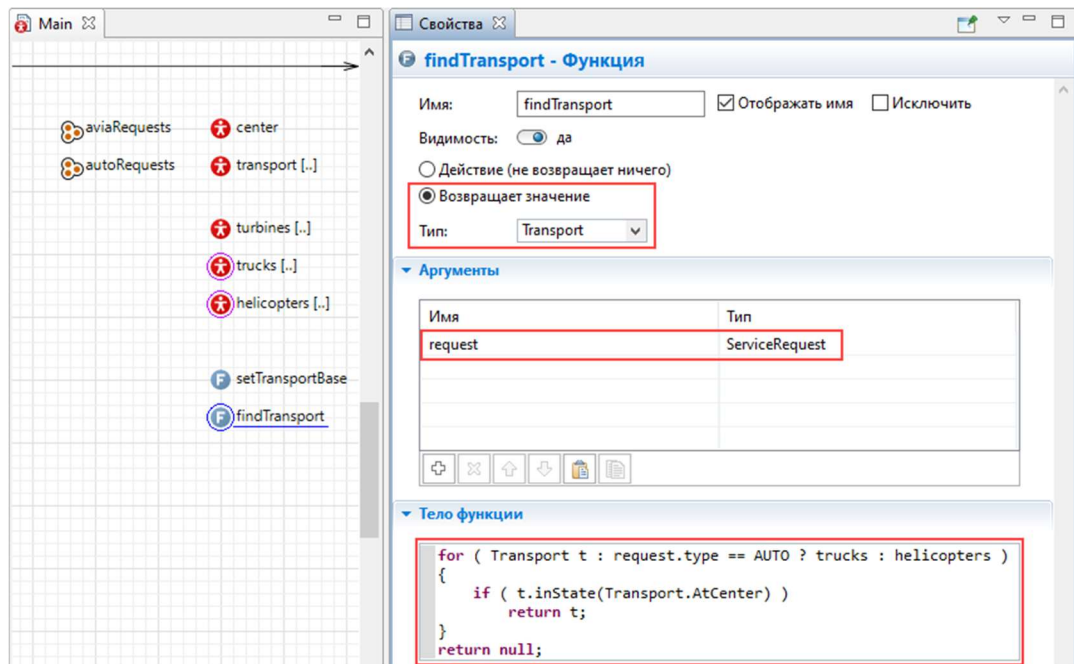
1. Откройте диаграмму **Main**. Добавьте два элемента **Коллекция** из палитры **Агент**.
2. Назовите коллекции `aviaRequests` и `autoRequests`. Обе коллекции имеют **Класс коллекции** `LinkedList` и **Тип элементов** `ServiceRequest`.



Теперь мы создадим функцию, которая будет искать свободный транспорт, чтобы выполнить сервисные работы.

Задайте логику управления транспортным парком

1. Перетащите элемент **Функция** из палитры **Агент** на диаграмму **Main**.
2. Назовите функцию `findTransport`.
3. Эта функция будет выполнять поиск свободного транспорта. Обнаружив свободный транспорт, функция должна его вернуть, поэтому выберите опцию **Возвращает значение** и задайте **Тип** значения `Transport`.
4. Функция должна проанализировать тип входящих запросов на сервисные работы. Для этого нам необходимо добавить один аргумент функции - с его помощью мы сможем передать функции запрос на сервисные работы. Разверните секцию **Аргументы** в свойствах функции и задайте в таблице аргумент **Типа** `ServiceRequest`.
5. Нам осталось задать алгоритм функции. Разверните секцию **Тело функции** в панели **Свойства** и введите код Java, как указано на изображении:



```

for (Transport t : request.type == AUTO ? trucks : helicopters)
{
    if (t.inState(Transport.AtCenter))
        return t;
}
return null;



```

В теле функции мы задаем цикл `for`. Этот цикл не всегда выполняет итерирование агентов одной популяции. Здесь он анализирует тип запроса на сервисные работы и проверяет, выполняется ли условие `request.type == AUTO`. Если условие выполняется, значит, поступил запрос на автомобильный транспорт, и далее цикл производит итерирование популяции `trucks`. Если же проверка условия возвращает значение `false`, это значит, что поступил запрос на вертолет. В таком случае цикл выполняет итерацию другой популяции: `helicopters`.

Внутри цикла мы проверяем, свободен ли в данный момент итерируемый агент. Для этого мы используем функцию агента `inState()`. Если управление диаграммой состояний агента на данный момент находится в состоянии **AtCenter**, это значит, что мы нашли свободный транспорт. В таком случае выполнение функции завершается на строке `return t`; и функция возвращает найденное значение (свободный транспорт). Если свободного транспорта нет, эта строка не будет выполняться и после цикла `for` выполнится следующая строка кода: `return null`; Таким образом мы сообщаем модели, что грузовик не найден (функция вернула значение `null` - свободный транспорт не существует).

В AnyLogic иногда нужно будет вводить код в параметрах различных элементов модели. Важно понимать, где именно вы "находитесь", вписывая код (какому типу агента принадлежит этот элемент), и как получить доступ к другим элементам из этого поля.

Элементы модели, принадлежащие тому же типу агента, доступны просто по именам.



Чтобы получить доступ к полю вложенного объекта, вы должны поставить точку "." после имени объекта и затем написать имя этого поля. Например, чтобы получить доступ к состоянию **AtCenter** диаграммы состояний, которая находится в агенте  **Transport**, из агента  **Main**, мы вызываем `Transport.AtCenter`.

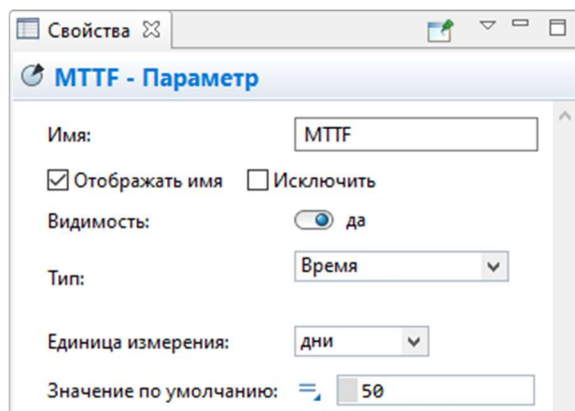
Далее давайте настроим тип агента  **Turbine**.

Шаг 5. Настройка поведения турбин

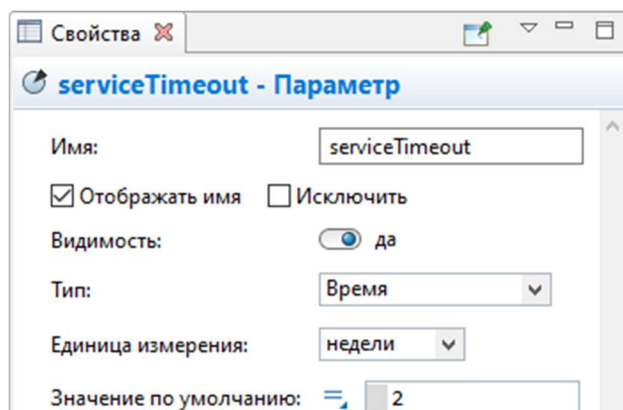
На этом этапе мы зададим поведение турбин с помощью диаграммы состояний: когда и как турбины выходят из строя или требуют планового техобслуживания. Также мы настроим анимацию турбин: пусть лопасти турбины перестают крутиться при экстренной поломке. К тому же, мы хотели бы видеть цветовую индикацию состояния турбины.

Задайте временные интервалы работы турбин



1. Двойным щелчком откройте тип агента  **Turbine** из дерева элементов модели. Начните с добавления двух элементов **Параметр**  из палитры **Агент** в графический редактор.
2. Параметр с именем **MTTF** (среднее время до аварии) имеет тип **Время** и его **Значение по умолчанию** равняется **50 дням**. **Дни** вы можете выбрать в свойстве **Единица измерения**.

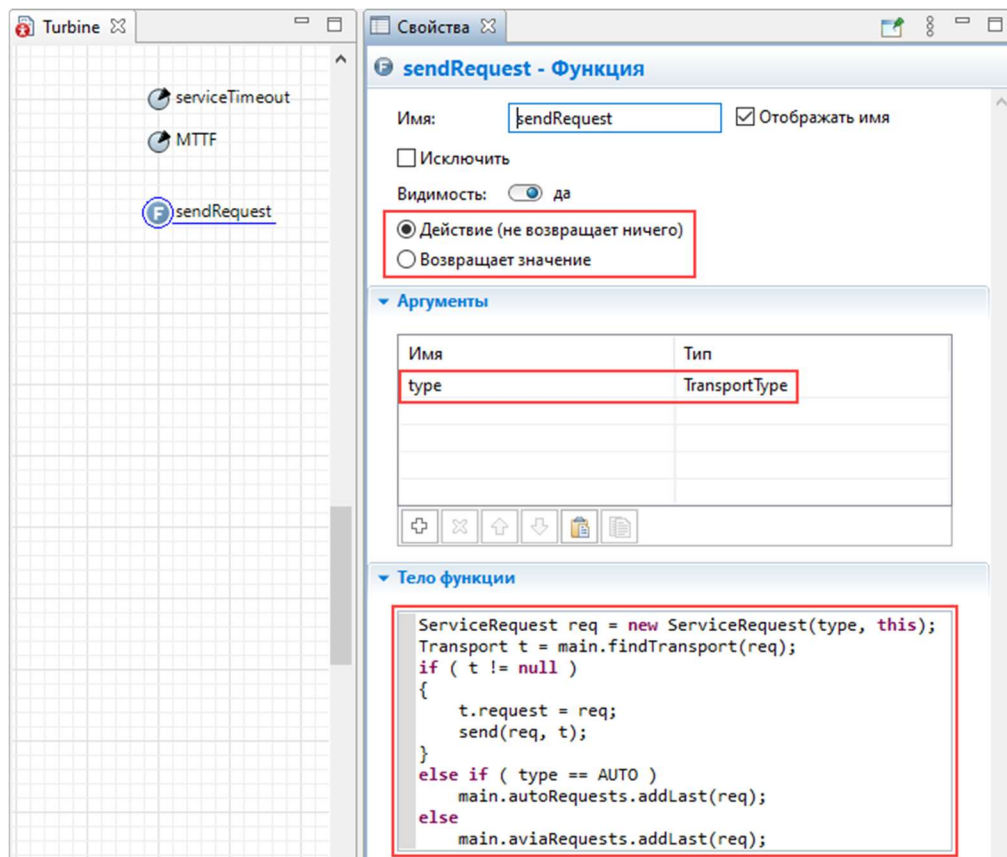


3. Второй параметр, **serviceTimeout**, также имеет тип **Время**; его **Значение по умолчанию** равняется **2 неделям**.





Добавьте функции, управляющие запросами на транспорт



1. Добавьте **Функцию**  из палитры **Агент** и назовите ее `sendRequest`.
2. Задайте ее свойства, как указано на рисунке ниже. В секции свойств **Аргументы** добавьте аргумент `type` типа `TransportType`. Тело функции ссылается на функцию `findTransport()`, которую мы ранее создали на диаграмме  `Main`. Когда турбина отправляет запрос на обслуживание, сервисный центр должен отправлять соответствующий тип транспорта: `AUTO` или `AVIA`.

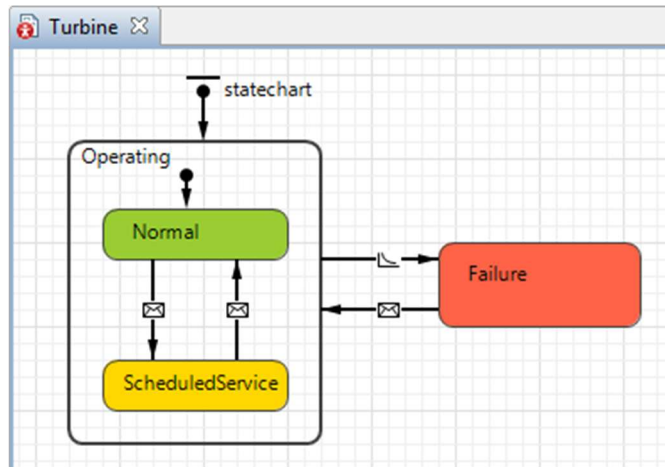


```
ServiceRequest req = new ServiceRequest(type, this);
Transport t = main.findTransport(req);
if ( t != null )
{
    t.request = req;
    send(req, t);
}
else if (type == AUTO)
    main.autoRequests.addLast(req);
else
    main.aviaRequests.addLast(req);
```

Задайте состояния турбины



1. Теперь мы готовы приступить к созданию диаграммы состояний турбины. Откройте палитры **Диаграмма состояний** и перетащите на диаграмму  `Turbine` элемент  **Начало диаграммы состояний**.

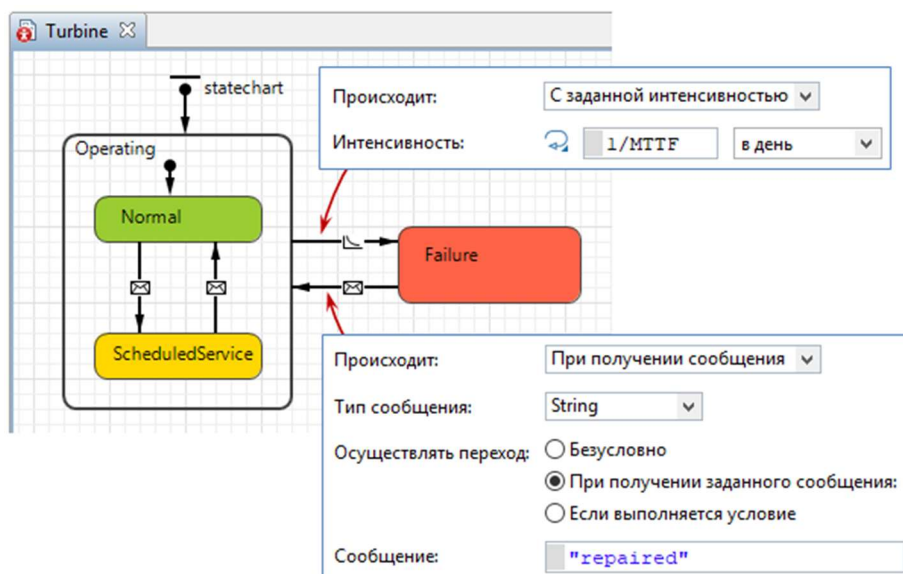
2. Добавьте состояния  и переходы , как показано на рисунке. Внутри сложного состояния Operating используйте **Указатель начального состояния**, ведущий в состояние Normal:





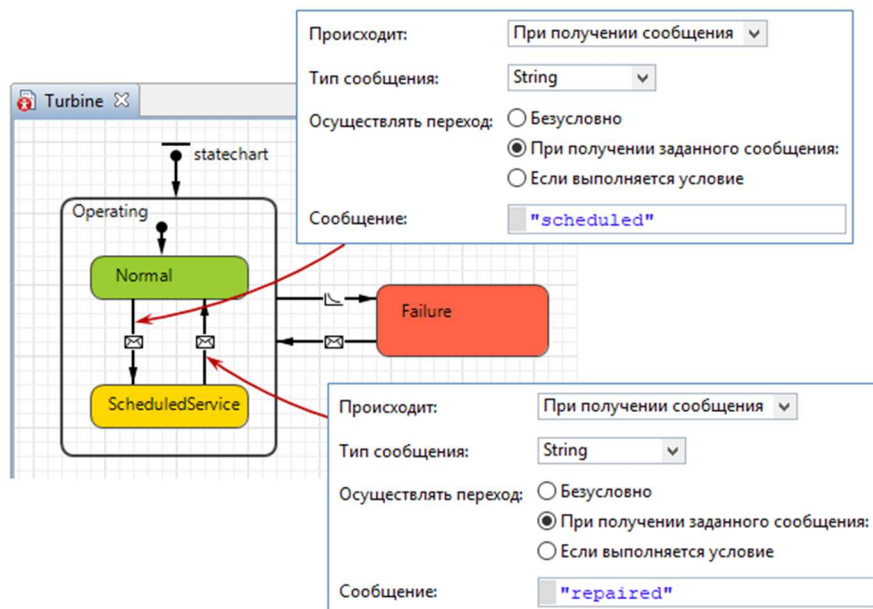
3. Для каждого состояния турбины задается свое **Действие при входе**. Когда происходит авария, турбина находится в состоянии Failure (авария) и отправляет запрос на обслуживание AVIA транспортом (вертолетом). Когда подходит время планового обслуживания, турбина отправляет запрос на AUTO транспорт — грузовик.


СОСТОЯНИЕ	ДЕЙСТВИЕ ПРИ ВХОДЕ
Failure	sendRequest(AVIA);
ScheduledService	sendRequest(AUTO);

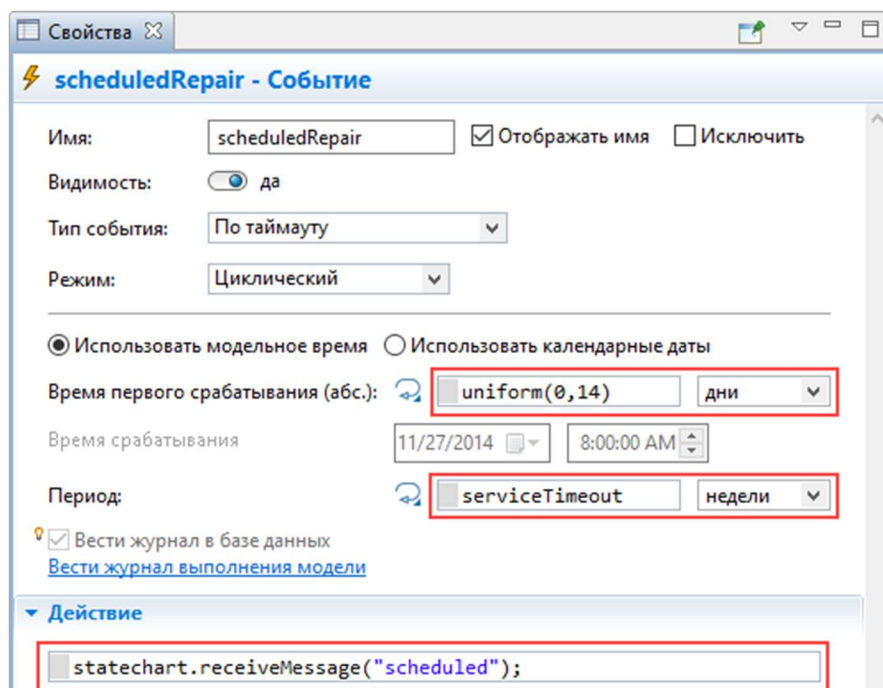
4. Переход от состояния Operating (работает) к состоянию Failure происходит **С заданной интенсивностью** , которая равняется 1/MTTF в день - один раз за среднее время до аварии. Переход обратно из состояния Failure в состояние Operating происходит **При получении заданного сообщения**  "repaired" (исправлено).



5. Иногда работающая турбина получает сообщение  "scheduled" (запланировано), и это означает, что ей требуется плановое обслуживание - при этом происходит переход в соответствующее состояние ScheduledService. Когда турбина получает сообщение  "repaired", обслуживание завершено, и она снова может вернуться в рабочее состояние Operating.

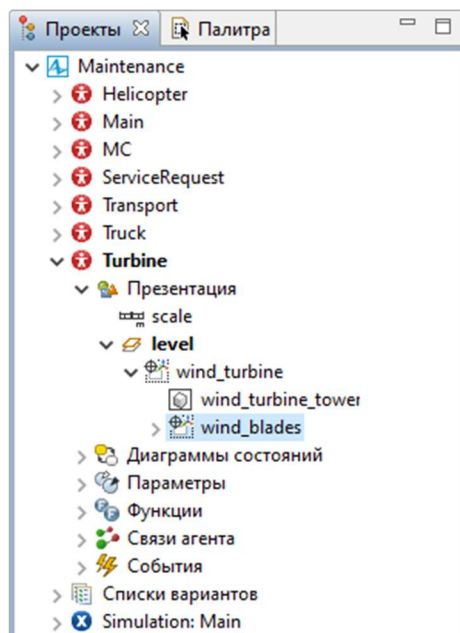


6. Добавьте циклическое  **Событие по таймауту** и назовите его scheduledRepair. Это событие будет запускаться, когда турбине требуется плановое обслуживание (согласно переменной serviceTimeout), и оно будет запускать переход от состояния Operating в состояние ScheduledService.

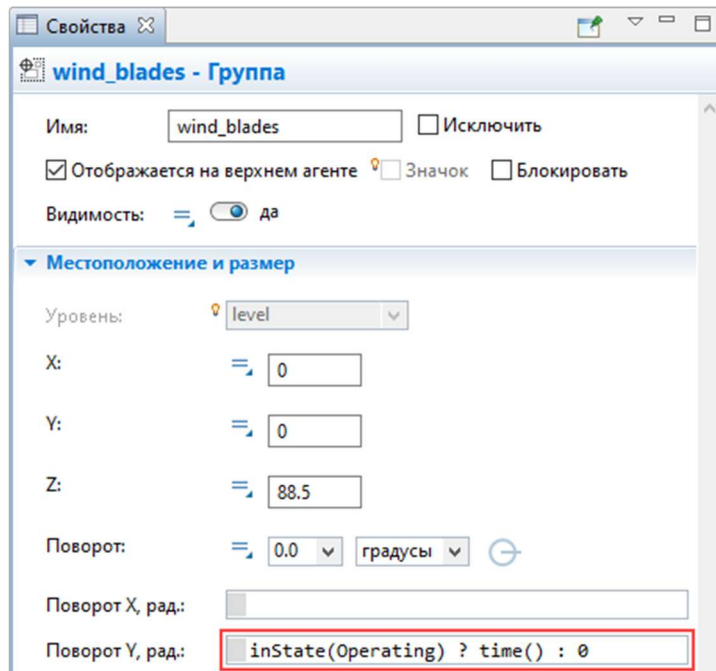


Настройте анимацию лопастей турбины

1. Нам необходимо изменить свойства лопастей турбины. Откройте панель **Проекты** и выберите группу **wind_blades**:




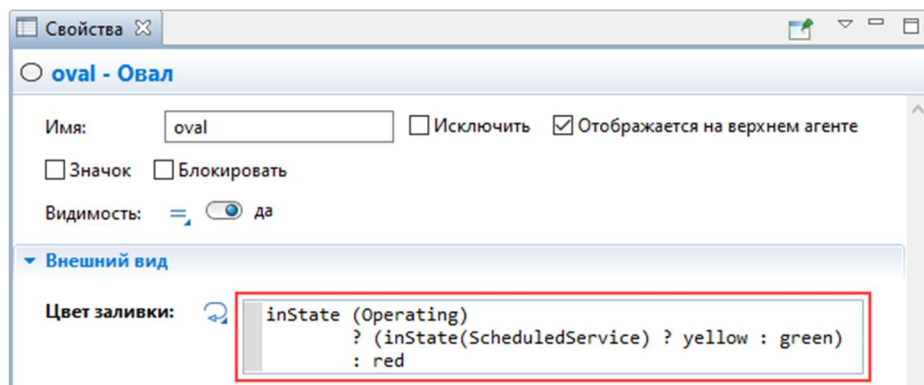
2. Перейдите в свойства этой группы. Разверните секцию **Местоположение и размер** и введите `inState(Operating) ? time() : 0` в поле свойства **Поворот Y**. Теперь лопасти будут вращаться, когда турбина находится в рабочем состоянии, а при поломке турбины будут останавливаться.



Добавьте индикацию состояния турбины

1. Откройте палитру **Презентация** и сделайте двойной щелчок по элементу **Овал**, чтобы перейти в режим рисования для этого элемента.
2. Нарисуйте круг вокруг фигуры турбины радиусом 10.

- Щелкните по фигуре круга правой кнопкой мыши и выберите **Порядок > На задний план** в контекстном меню.
- Перейдите в секцию **Внешний вид** свойств круга. Введите выражение, которое будет вычисляться во время прогона модели в поле свойства **Цвет заливки**, чтобы цвет менялся в зависимости от состояния турбины.
- Чтобы иметь возможность задать динамическое значение в поле свойства, щелкните его значок .



- Если турбина ожидает запланированного обслуживания, круг станет отображать желтый цвет, иначе - зеленый, когда турбина в рабочем состоянии; в случае, когда турбина выходит из строя, мы получим красный цвет фигуры.


Дополнительно, вы можете изменить размер элементов диаграммы состояний и изменить цвет состояний по умолчанию на более подходящий.

Запустите модель. Вы увидите, что некоторые турбины работают, некоторые ожидают планового обслуживания, а некоторые - вышли из строя.





Теперь мы можем вернуться к настройке транспорта.

Шаг 6. Завершение настройки логики транспорта

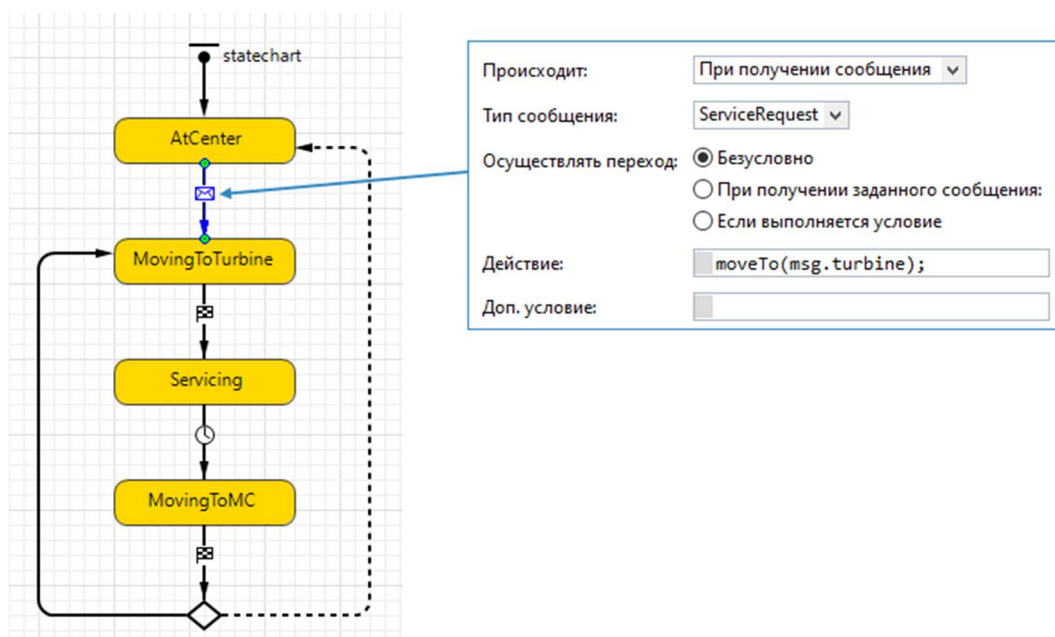
На этом шаге мы добавим необходимые настройки в диаграмму состояний типа агента  Transport. Мы зададим движение транспорта между сервисным центром и турбинами с помощью переходов различных типов.

Укажите тип транспорта

1. Откройте графическую диаграмму агента  Truck, дважды щелкнув по соответствующему элементу в дереве модели.
2. В секции свойств **Действия агента** укажите тип агента в поле **При запуске** с помощью следующего выражения: `type = AUTO;`
3. Тем же способом в свойствах агента  Helicopter укажите соответствующий тип: `type = AVIA;`

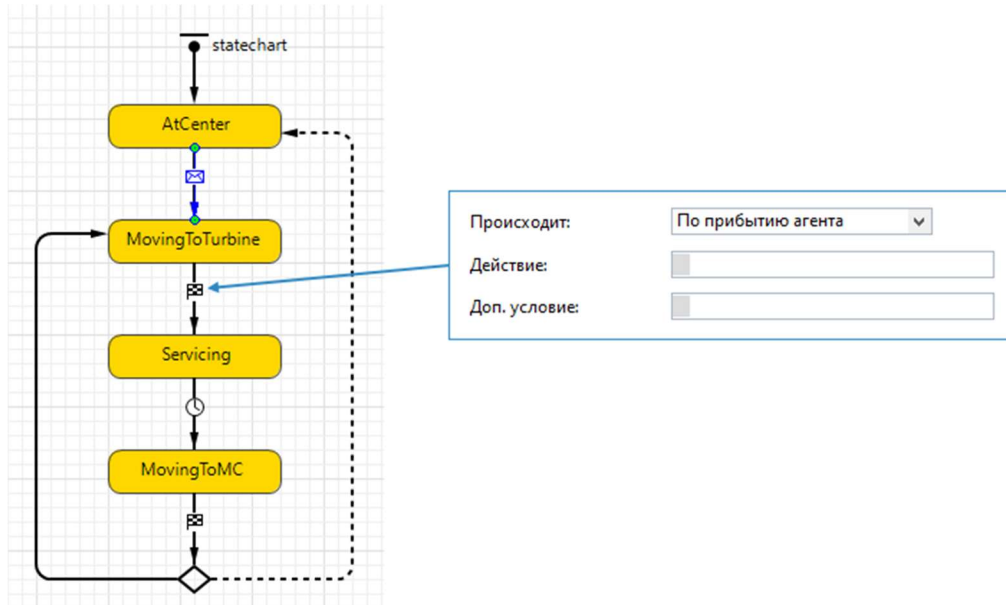
Настройте диаграмму состояний

1. Дважды щелкните по агенту  Transport, чтобы открыть его графическую диаграмму. На ней вы обнаружите ранее созданную диаграмму состояний. Теперь мы можем задать ее настройки.
2. У состояний в диаграмме (**AtCenter**, **MovingToTurbine**, **Servicing**, **MovingToMC**) нет каких-то особенных свойств. Мы будем задавать логику движения транспорта с помощью различных переходов между состояниями.
3. Переход из состояния **AtCenter** в состояние **MovingToTurbine** происходит **При получении сообщения**  типа `ServiceRequest`. Здесь необходимо указать **Действие**, которое следует выполнить транспорту: двигаться к турбине, от которой пришло сообщение.

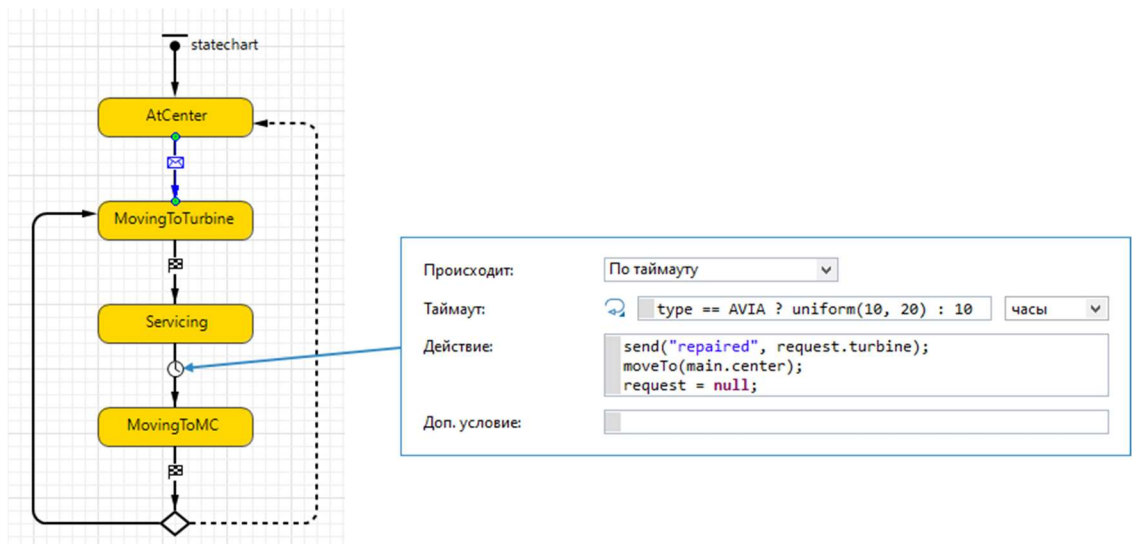



```
moveTo(msg.turbine);
```

4. Переход из состояния **MovingToTurbine** в состояние **Servicing** запускается **По прибытию агента**. Грузовик или вертолет, который достиг турбины, начинает выполнять свою задачу сразу после прибытия на место. По завершению обслуживания, о чем нас "оповещает" турбина, транспортное средство может отправляться обратно на базу.




5. Следующий переход идет из состояния **Servicing** в состояние **MovingToMC**. Действие, задающее возвращение в сервисный центр, происходит **По таймауту**.

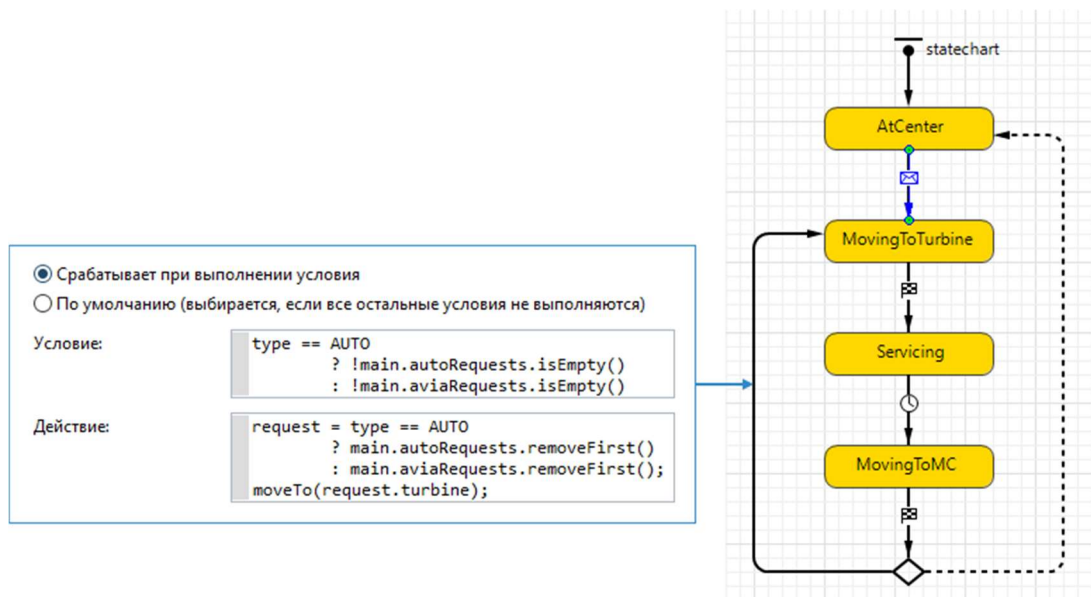


Таймаут: `type == AVIA ? uniform(10, 20) : 10`

Действие:

```
send("repaired", request.turbine);
moveTo(main.center);
request = null;
```

6. На рисунке выше вы видите, что последний переход, из **MovingToMC** в ветвление, откуда транспорт может направиться либо в состояние **MovingToTurbine**, либо вернуться в **AtCenter**, происходит **По прибытию агента** . Когда он происходит, выполняется проверка, при прохождении которой транспортное средство либо выезжает на обслуживание очередной турбины, либо отправляется обратно в сервисный центр и остается там, пока снова не понадобится для обслуживания.
7. Выберите переход из ветвления в состояние **MovingToTurbine**. Здесь нам необходимо задать **Условие** и **Действие**: код в поле **Условие** проверит, есть ли в данный момент запросы от турбины на запланированные ремонтные работы или срочный ремонт из-за поломки. Если такие запросы есть, код в поле **Действие** проверит, какой именно запрос поступил, и отправит соответствующий тип транспорта к турбине:






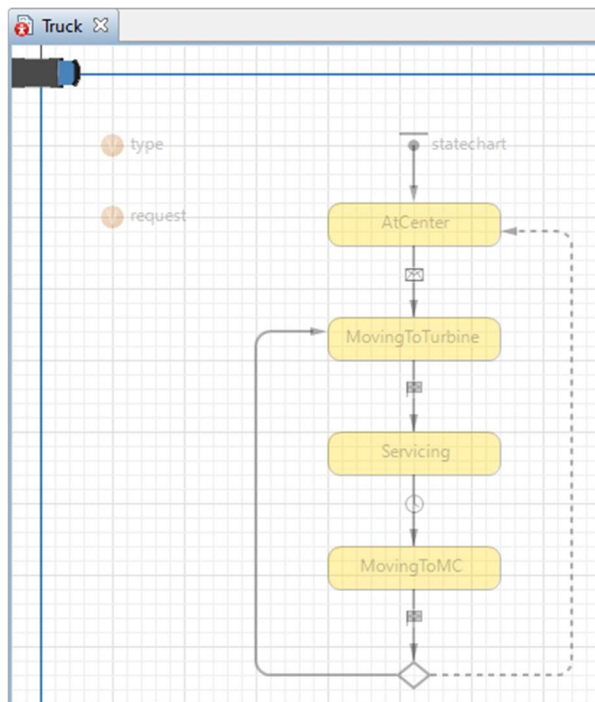
Условие:

```
type == AUTO ? !main.autoRequests.isEmpty() : !main.aviaRequests.isEmpty()
```

Действие:

```
request = type == AUTO ? main.autoRequests.removeFirst() : main.aviaRequests.removeFirst();
moveTo(request.turbine);
```

Теперь, если вы откроете тип агента  **Truck** или  **Helicopter**, вы увидите там проекцию элементов типа агента  **Transport**, который они расширяют. Эти элементы отображаются на их диаграммах для удобства, но чтобы изменять их свойства, вернитесь обратно на диаграмму самого агента:



Запустите модель. Изначально все турбины находятся в рабочем состоянии (обозначаются зеленым цветом). Затем вы увидите, как грузовик направляется к той турбине, которой требуется плановое обслуживание (желтые турбины), а вертолет направится к той турбине, которая вышла из строя (красные турбины), чтобы выполнить срочные ремонтные работы.





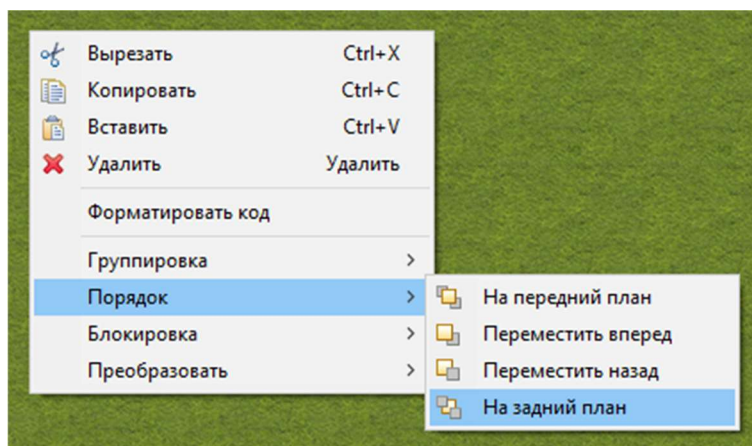
Мы закончили задавать логику модели, давайте теперь улучшим анимацию, создадим 3D анимацию и исследуем модель во время прогона.



Шаг 7. Запуск модели и исследование ее элементов

Это последний шаг нашего учебного пособия по разработке агентной модели обслуживания турбин.


Добавьте элементы презентации

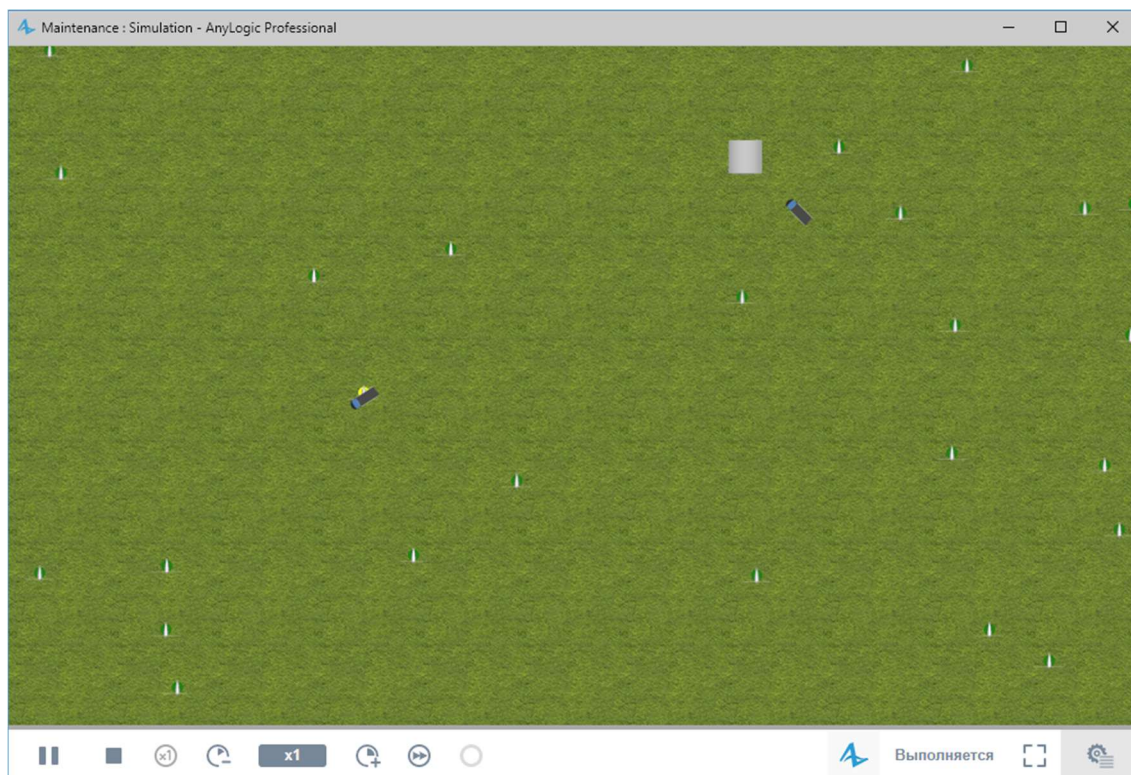
1. Вы можете добавить фон для анимации агентов - турбин и транспортных средств - на диаграмме типа агента  **Прямоугольник**. Выделите элемент **Прямоугольник**  в палитре **Презентация**, чтобы перейти в режим рисования. Затем щелкните в графический редактор и тащите прямоугольник, не отпуская кнопки мыши, пока не получите нужную форму прямоугольника.
2. Перейдите в секцию свойств фигуры **Внешний вид** и выберите текстуру **Grass** в качестве **Цвета заливки** и опцию **Нет цвета** для **Цвета линии**. Обратите внимание на свойства **Z** и **Z-Высота** в секции **Местоположение и размер**. Например, если **Z-Высота** равняется **10**, вы можете установить **Z** на **-10**, иначе другие фигуры анимации "утонут" в этом фоне, ведь они по умолчанию перемещаются на нулевом уровне.
3. Так как прямоугольник — это последняя добавленная нами фигура, он отображается поверх всех остальных. Щелкните прямоугольник правой кнопкой мыши и выберите **Порядок > На задний план**.




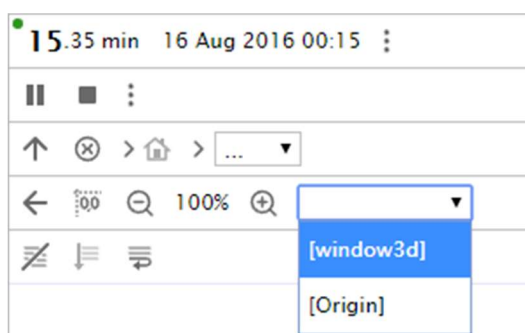
4. Вы также можете щелкнуть фигуру анимации агента  **МС** и выбрать **Порядок > На передний план**, чтобы "спрятать" транспорт за ней.
5. Чтобы задействовать 3D анимацию, откройте палитру **Презентация** и перетащите элемент **3D Окно**  в графический редактор. Затем вы можете изменить его **Свойства**: размер или цвет.

Запустите модель

1. Щелкните по кнопке панели управления **Запуск**  и запустите модель.



2. При создании 3D окна, AnyLogic добавляет [область просмотра](#), которая позволяет легко переключаться к сцене 3D анимации во время выполнения модели. Чтобы переключиться к 3D анимации в запущенной модели, откройте [панель разработчика](#), щелкнув  **Показать /скрыть панель разработчика** в правом углу [панели управления](#). В открывшейся панели разработчика, раскройте список **Выбрать область и показать** и выберите опцию **[window3d]**.



ЧТОБЫ	ВЫПОЛНИТЕ СЛЕДУЮЩИЕ ДЕЙСТВИЯ
Переместить сцену	<ol style="list-style-type: none"> 1. Нажмите левую кнопку мыши в области 3D окна и держите ее нажатой. 2. Передвиньте мышь в направлении перемещения.
Повернуть сцену	<ol style="list-style-type: none"> 1. Нажмите клавишу Alt и держите ее нажатой. 2. Нажмите левую кнопку мыши в области 3D окна и держите ее нажатой. 3. Передвиньте мышь в направлении вращения.





Приблизить/отдалить
сцену

1. Покрутите колесо мыши от/на себя в области 3D окна.

3. Вы можете перемещаться по 3D сцене с помощью мыши и следующих клавиш:



4. Вы можете управлять запуском модели с помощью [кнопок панели](#)

[управления](#): **Пауза**  и **Прекратить выполнение эксперимента** 
, **Ускорить**  или **Замедлить**  модель.

5. Кроме того, вы можете следить за разными типами агентов. В [панели](#)

[разработчика](#) раскройте список  **Выбрать агента уровнем ниже и перейти** и выберите **turbines[25]**. В квадратных скобках указано количество агентов в этой популяции. Выбрав этот элемент в списке, вы увидите диаграмму первого агента популяции турбин. Чтобы переместиться к другому агенту этой популяции, введите индекс требуемого агента справа. Таким образом можно наблюдать за любым агентом в модели.

И, пожалуй, последний штрих. Как вы можете видеть, сейчас и грузовики и вертолеты паркуются прямо в точке расположения турбины, что, конечно, придает анимации некую незавершенность.

Мы можем поправить и этот момент, слегка сдвинув анимацию турбины от ее логической точки местоположения. Для этого откройте диаграмму турбины Turbine и переместите анимацию турбины чуть вверх (например, чтобы лопасти стали располагаться прямо по оси X).

Запустите модель еще раз. Создание нашей модели завершено, и теперь вы можете добавить в эту модель необходимую аналитику и, если нужно, провести эксперимент оптимизации