Приднестровский государственный университет им. Т. Г. Шевченко

Физико-математический факультет

Кафедра прикладной математики и информатики

Курсовая работа

по дисциплине «Спецсеминар»

Моделирование систем в среде имитационного моделирования AnyLogic

Выполнил:

студент 403 гр. д/о

физ.-мат. факультета

Епифанов Андрей Сергеевич

Руководитель:

ст. преподаватель

кафедры информатики и ВТ

Великодный Вадим Игоревич

Тирасполь, 2016

Оглавление

[Аннотация 3](#_Toc482196724)

[Введение 4](#_Toc482196725)

[1. Основы моделирования систем 5](#_Toc482196726)

[1.1 Модели и моделирование 5](#_Toc482196727)

[1.2 Классификация моделей 7](#_Toc482196728)

[1.3 AnyLogic 9](#_Toc482196729)

[1.4 Пользовательский интерфейс 10](#_Toc482196730)

[1.5 Средства AnyLogic для имитационного моделирования систем 12](#_Toc482196731)

[1.6 Элементы построения модели 15](#_Toc482196732)

[2. Реализация модели движения маршрутных транспортных средств по установленному маршруту 17](#_Toc482196733)

[2.1 Постановка задачи 17](#_Toc482196734)

[2.2 Пошаговое решение задачи 18](#_Toc482196735)

[Заключение 29](#_Toc482196736)

[Список используемой литературы 30](#_Toc482196737)

# Аннотация

Настоящая работа посвящена изучению имитационного моделирования в среде AnyLogic. В результате была разработана небольшая модель с транспортной развязкой по которой движутся маршрутные такси.

# Введение

Моделирование является самым распространенным средством изучения объектов, явлений и процессов реального мира. Моделирование необходимо в тех случаях, когда изучать реальные объекты и процессы затруднительно или практически невозможно и существенно упрощает и удешевляет разработку и оптимизацию сложных и дорогих систем.

**Целью** работы является изучить среду имитационного моделирования Anylogic и разработать модель движения общественного транспорта.

Для достижения цели были поставлены следующие **задачи**:

- изучить литературу по теме исследования,

- изучить работу со средой Anylogic,

- разработать имитационную модель движения транспорта.

# Основы моделирования систем

## Модели и моделирование

В своей деятельности человек очень часто использует модели, то есть создает образ того объекта, явления или процесса, с которым ему предстоит работать.

**Модель** – это некий новый упрощенный объект, который отражает существенные особенности реального объекта, процесса или явления.  
Анализ модели и наблюдение за ней позволяют познать суть реально существующего, более сложного объекта, процесса, явления, называемо прототипом или оригиналом.

**Моделирование** представляет из себя метод решения задач, при котором исследуемую систему заменяем на более простой объект, который описывает нашу систему и называется моделью. Моделирование применяется в тех случаях, когда проведение экспериментов над системой невозможно, например, из-за высокой стоимости или длительности проведения эксперимента в реальном времени.

**Имитационной моделью** называется компьютерная программа, которая описывает структуру и демонстрирует поведение системы во времени. Имитационная модель позволяет получать подробную статистику о разных аспектах функционирования системы в зависимости от входных данных. **Имитационное моделирование** – это разработка и постановка экспериментов на моделях. Целью моделирования является принятие целесообразных управленческих решений. Компьютерное моделирование становится обязательным этапом в принятии ответственных решений во всех областях деятельности человека в связи с усложнением систем, в которых человек должен действовать и которыми он должен управлять. Знание принципов и возможностей имитационного моделирования, умение создавать и применять модели являются необходимыми требованиями к инженеру, менеджеру, бизнес-аналитику.

Современные системы моделирования поддерживают все из новейших информационных технологий, включая развитые графические оболочки для целей конструирования моделей и интерпретации выходных результатов моделирования, мультимедийные средства, анимацию в реальном масштабе времени, объектно-ориентированное программирование, Internet - решения и др.

## Классификация моделей

Объектов моделирования существует огромное количество. И для того, чтобы ориентироваться в их многообразии необходимо все это классифицировать, то есть каким-либо образом упорядочить, систематизировать.

При классификации объектов по «родственным» группам необходимо правильно выделить некий единый признак (параметр, а затем объединить те объекты, у которых он совпадает). Рассмотрим наиболее распространенные признаки, по которым можно классифицировать модели.

I. С учетом фактора времени:

динамические;

статические.

II. По области использования:

учебные;

опытные;

игровые;

научно-технические;

имитационные.

III. По области знаний:

математические;

химические;

физические;

географические;

ит.д.

IV. По способу реализации:

компьютерные;

некомпьютерные.

V. По способу представления:

материальные;

информационные

вербальные;

графические;

математические;

табличные;

специальные.

Процесс построения информационных моделей с помощью формальных языков (математических, логических и т.д.) называется **формализацией**   
Более полное определение формализации – это приведение (сведение) существенных свойств и признаков объекта моделирования к выбранной форме.

Формами представления информационной модели могут быть: словесное описание, таблица, схема, чертеж, формула, алгоритм, компьютерная программа и т.д.

## AnyLogic

Пакет AnyLogic является отечественным профессиональным инструментом нового поколения, предназначенный для разработки и исследования имитационных моделей. Продукт был разработан компанией (XJ Technologies), в г. Санкт-Петербург. Разработка осуществлялась на основе новейших идей в области информационных технологий, теории параллельных взаимодействующих процессов и теории гибридных систем. Благодаря этим идеям сильно упрощается построение сложных имитационных моделей, есть возможность использования одного инструмента при изучении различных стилей моделирования.

Программный инструмент AnyLogic является объектно-ориентированным. Другой базовой концепцией является представление модели как набора взаимодействующих, параллельно функционирующих активностей. Активный объект в AnyLogic является объектом со своим собственным функционированием, взаимодействующий с окружением. Он может включать в себя любое количество экземпляров других активных объектов. Графическая среда поддерживает проектирование, разработку, документирование модели, выполнение компьютерных экспериментов, оптимизацию параметров относительно некоторого критерия.

При создании модели возможно применение элементов визуальной графики (диаграммы состояний, сигналы, события (таймеры), порты и т.д.), планирование событий, библиотеки активных объектов.

При разработке модели на AnyLogic возможно использование концепций и средств из нескольких классических областей имитационного моделирования: динамических систем, дискретно-событийного моделирования, системной динамики, агентного моделирования. Так же в AnyLogic есть возможность интегрировать различные подходы для получения наиболее полной картины взаимодействия различных процессов.

## Пользовательский интерфейс

После запуска AnyLogic открывается рабочее окно, в котором для продолжения работы необходимо создать новый проект либо открыть уже существующий.

Окно проекта обеспечивает легкую навигацию по элементам проекта, таким как пакеты, классы и т.д. Поскольку проект организован иерархически, то он отображается в виде дерева: сам проект образует верхний уровень дерева рабочего проекта, пакеты - следующий уровень, классы активных объектов и сообщений - следующий и т.д. возможно копирование, перемещение и удаление любых элементов дерева объектов, легко управляя рабочим проектом.

Так выглядит окно проекта, вид может быть индивидуальным, путем закрывания, открывания и перемещения вкладок.



В AnyLogic для всех элементов есть собственное окно свойств, в котором указаны свойства (параметры) этого элемента. Для того чтобы открыть свойства элемента, достаточно выделить этот элемент. В окне свойств содержится несколько вкладок, каждая из которых имеет элементы управления, такие как поля ввода, флажки, переключатели, кнопки и т.д., с помощью которых можно просматривать и изменять свойства элементов. Число вкладок и их внешний вид зависит от типа выбранного элемента.



В **Окне палитры**имеются элементы, которые могут быть добавлены на диаграмму. Элементы разбиты по группам, отображающиеся на разных вкладках. Чтобы добавить объект палитры на диаграмму, нужно щелкнуть по элементу в палитре, а затем щелкнуть по диаграмме.

В активном объекте могут содержаться переменные. Они могут быть либо внутренними, либо интерфейсными. Так же могут иметься переменные, моделирующие, меняющиеся во времени величины. Переменные могут быть связаны с переменными других активных объектов. Тогда при изменении значения одной переменной будет немедленно меняться и значение переменной другого объекта. Это обеспечивает непрерывное объектов.

Запуск и отладка модели производится с помощью меню Модель и панели инструментов:



## Средства AnyLogic для имитационного моделирования систем

Создание модели осуществляется в графическом редакторе AnyLogic с использованием многочисленных средств поддержки, упрощающих работу. После чего модель компилируется встроенным компилятором AnyLogic и запускается на выполнение. В процессе выполнения модели можно видеть ее поведение, менять параметры, выводить результаты в различных формах и выполнять эксперименты с моделью. Для вычислений и описания логики поведения объектов в AnyLogic используется язык Java. Основными строительными блоками модели AnyLogic являются активные объекты, которые позволяют моделировать любые объекты реального мира. Классы являются хорошим средством, который позволяет структурировать сложную систему. Класс определяет шаблон, в соответствии с которым строятся отдельные экземпляры класса. Эти экземпляры определяются как объекты других активных объектов. Активный объект является экземпляром класса активного объекта. Чтобы создать модель AnyLogic, необходимо создать классы активных объектов (или использовать объекты библиотек AnyLogic) и задать их взаимосвязи. AnyLogic превращает создаваемые графически классы активных объектов в классы Java, благодаря этому мы имеем возможность воспользоваться всеми преимуществами объектно-ориентированного моделирования.

Активные объекты могут содержать вложенные объекты, причем уровень вложенности не ограничен. Это позволяет производить декомпозицию модели на любое количество уровней детализации.

Активные объекты имеют четко определенные интерфейсы взаимодействия. Они взаимодействуют со своим окружением только посредством своих интерфейсных элементов. Это помогает нам легче создавать системы со сложной структурой, а также активные объекты можно использовать повторно. После создания класса активного объекта, появляется возможность создавать любое количество объектов - экземпляров данного класса. Каждый активный объект имеет структуру (совокупность включенных в него активных объектов и их связи), а также поведение, определяемое совокупностью переменных, параметров и т.п. Каждый экземпляр активного объекта в работающей модели имеет свое собственное поведение, имеет свои значения параметров, работает независимо от других объектов, взаимодействуя с ними и с внешней средой. При построении модели могут быть использованы средства визуальной разработки, задания численных значений параметров, аналитических записей соотношений переменных и аналитических записей условий наступления событий. Основой программирования в AnyLogic является визуализация - построение с помощью графических объектов и пиктограмм иерархий структуры и поведения активных объектов является надстройкой над языком Java - одним из самых мощных и в то же время самых простых современных объектно-ориентированных языков. Все объекты, которые определены пользователем, компилируются в конструкции языка Java, а затем происходит компиляция всей собранной программы на Java, в исполняемый код. Хотя программирование сведено к минимуму, разработчику модели необходимо иметь некоторое представление об этом языке (например, знать синтаксически правильные конструкции).

Средствами задания поведения объектов являются переменные, события и диаграммы состояний. Переменные отражают изменяющиеся характеристики объекта. События могут наступать с заданным интервалом времени и выполнять заданное действие. Диаграммы состояний позволяют увидеть поведение объекта во времени под воздействием событий или условий, они состоят из графического изображения состояний и переходов между ними.

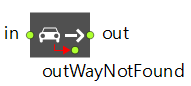
Для того что бы показать поведение сложной системы, в Anylogic есть средства, позволяющие представить моделируемую систему в живой форме динамической анимации. Визуализация процесса функционирования системы позволяет проверить адекватность модели, выявить ошибки при задании логики. Благодаря анимации можно легко создавать виртуальный мир, который управляется динамическими параметрами по законам, определенным пользователем с помощью уравнений и логики моделируемых объектов. Графические элементы, добавленные на анимацию, называются динамическими, так как все их параметры: видимость, цвет и т.п. - можно сделать зависимыми от переменных и параметров модели, которые в свою очередь меняются со временем при выполнении модели.

## Элементы построения модели

Для выполнения данной курсовой работы нам понадобятся такие элементы библиотек, как:

 Car Source. Создает автомобили и пытается поместить их в указанное место дорожной сети. Автомобиль можно поместить на указанную дорогу или парковку (это задается параметром *Появляется*).

Если в качестве места появления автомобиля указана дорога, необходимо дополнительно указать направление дороги. Автомобили будут появляться в начале указанной дороги, после чего будут начинать движение в указанном направлении. По умолчанию, выбран параметр *Случайная полоса*, поэтому автомобили будут появляться на случайных полосах дороги (если на указанной дороге больше одной полосы). Можно явно указать номер полосы, на которой будут появляться автомобили. Для этого необходимо снять флажок с опции *Случайная**полоса* и указать номер в параметре *Номер**полосы*;

CarMoveTo. Блок, который управляет движением автомобиля. Автомобиль может ехать, только когда он находится в блоке CarMoveTo. Автомобиль пытается рассчитать путь от своего текущего места до указанного места назначения, когда поступает в блок CarMoveTo.

В качестве цели движения могут выступать: *дорога*, *парковка*, *автобусная остановка* или *стоп-линия*. Указанное место назначения должно находиться в той же дорожной сети, что и автомобиль. Если от текущего местоположения автомобиля к указанному месту нет пути, автомобиль покидает блок через порт *outWayNotFound*;

http://127.0.0.1:59329/help/topic/com.anylogic.help/html/_RoadTraffic/reference/images/carDispose.png Car Dispose. Удаляет машины из модели;

 SelectOutput. Объект направляет входящие заявки в один из двух выходных портов в зависимости от выполнения заданного условия;

Delay. Задерживает заявки на заданный период времени.

# Реализация модели движения маршрутных транспортных средств по установленному маршруту

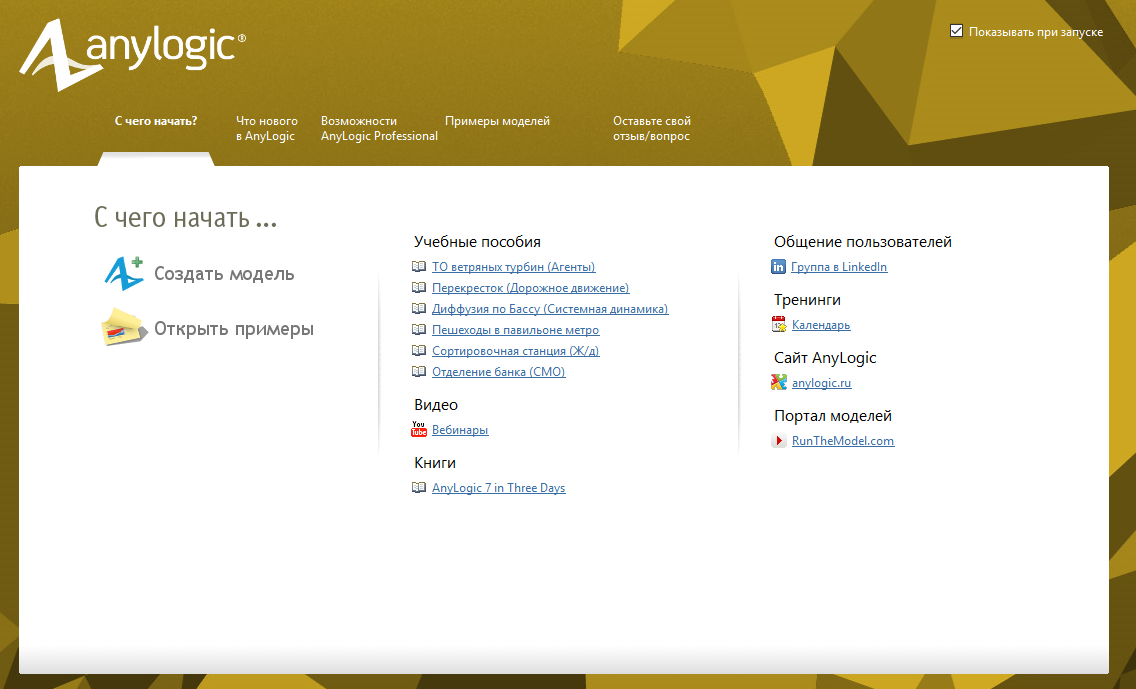
## Постановка задачи

В данной задаче мы исследуем движение маршрутных транспортных средств по г. Тирасполь. В результате должны получить дорожную сеть, по которой маршрутные такси будут двигаться по остановкам.

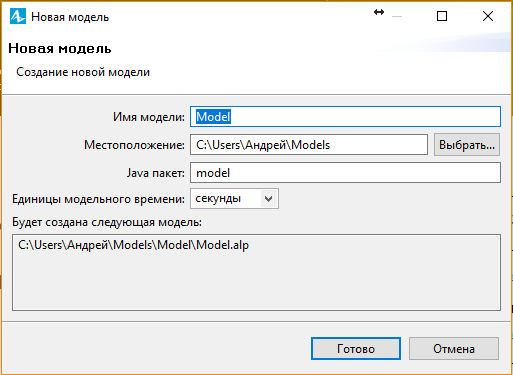
Для решения данной задачи будет использоваться инструмент имитационного моделирования Anylogic.

## Пошаговое решение задачи

После запуска Anylogic перед нами появится начальная страница



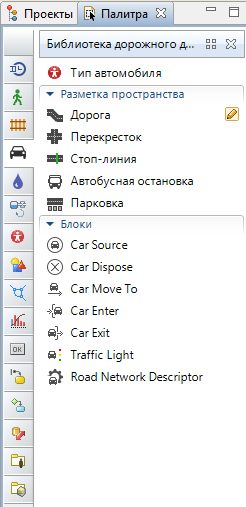
Жмем “Создать модель”, после чего появится окно создания новой модели



Указываем имя модели, ее расположение на диске, Java пакет оставляем без изменений и единицы модального времени выбираем секунды, далее жмем кнопку “готово" и приступаем к работе.

В окне Палитра заходим в Раздел презентации, выбираем объект Изображение и перетаскиваем в рабочую область, откроется окно выбора изображения, выбираем заранее приготовленную карту города.

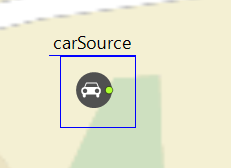
Настало время рисовать дорожную сеть, для этого в окне Палитра заходим в раздел Библиотека дорожного движения и элементами разметки пространства рисуем дорожную сеть с автобусными остановками и светофорами.



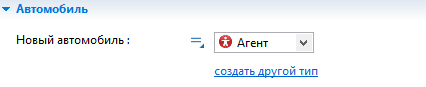
В результате должно получиться что-то вроде этого:



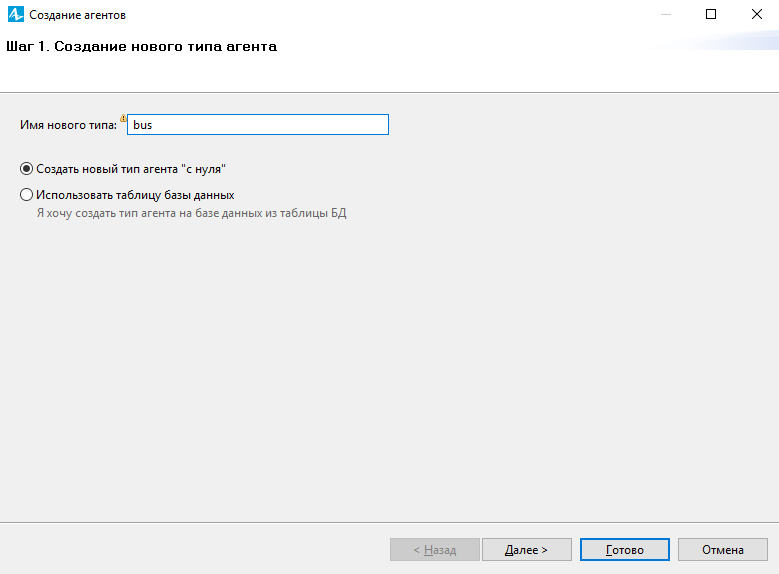
Далее добавим элемент CarSource из библиотеки дорожного движения и создадим новый тип агента для автобусов, агента назовем Bus.

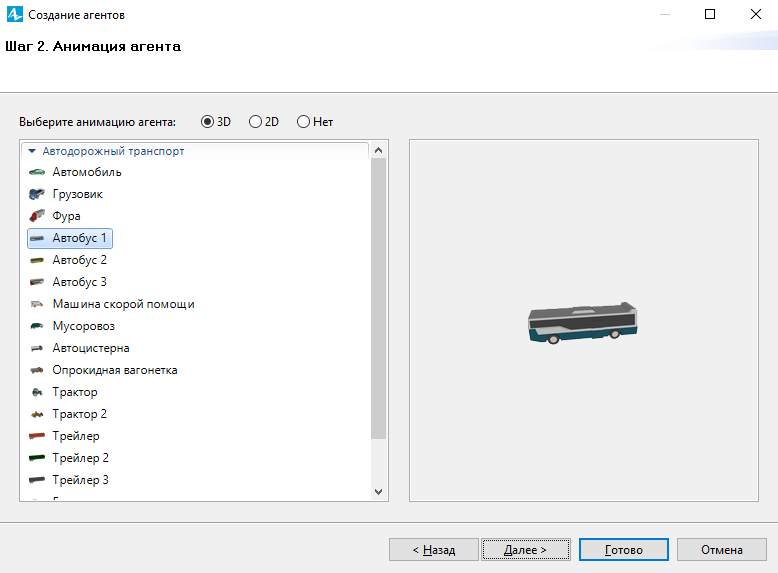


Для этого откроем свойства CarSource и во вкладке Автомобиль

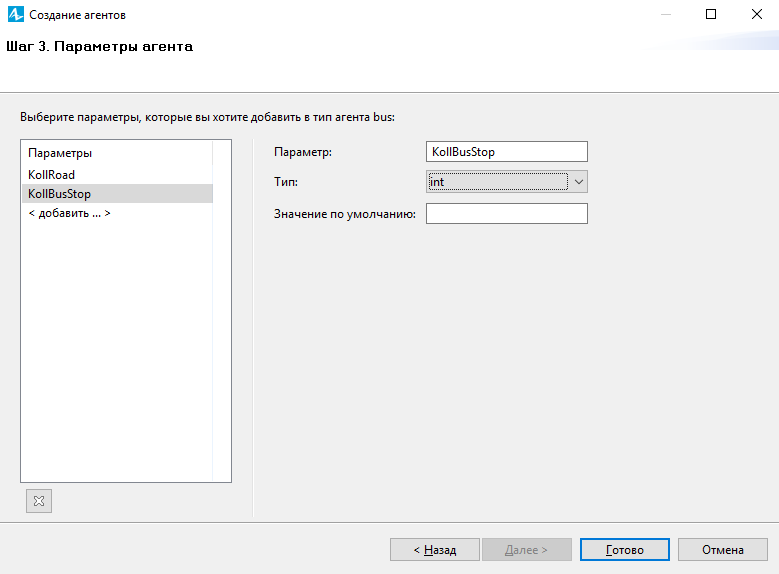


Нажмем создать другой тип





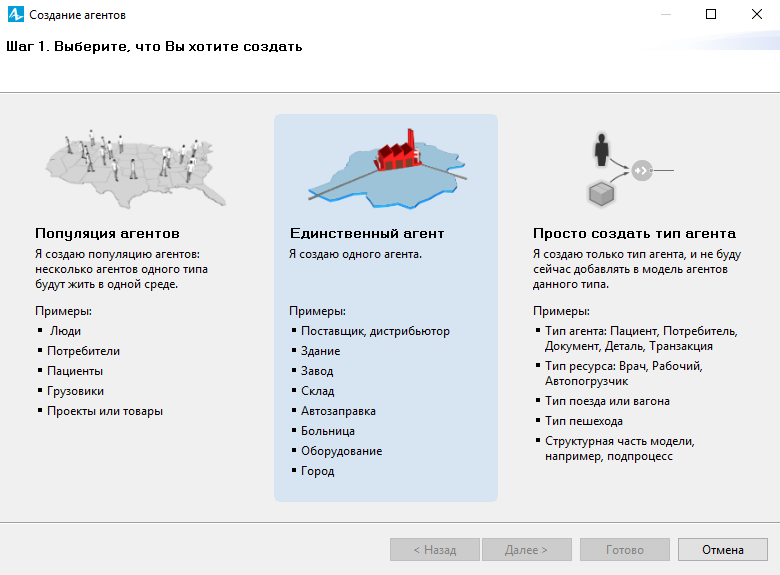
Добавим два параметра **KollRoad** и **KollBasStop**, тип у них будет целочисленный, о которых мы поговорим чуточку позже, после чего нажимаем Готово.

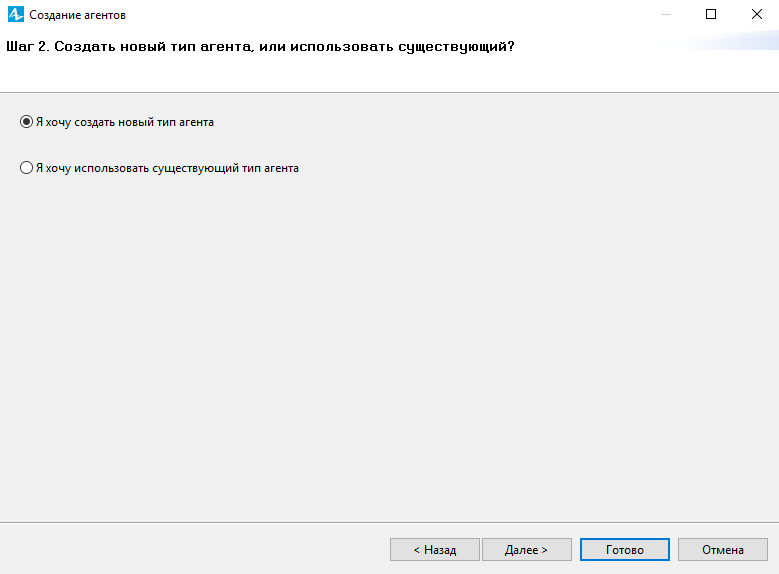


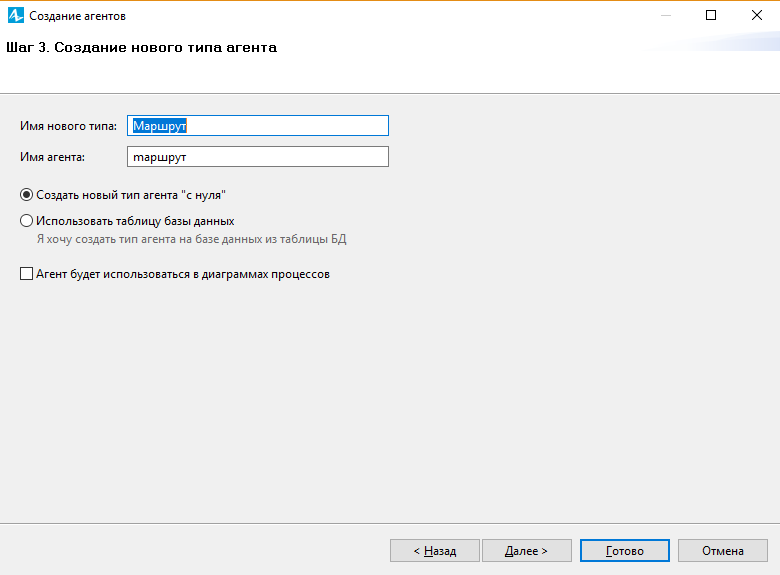
Теперь в свойствах CarSource мы можем выбрать созданного нами агента Bus.



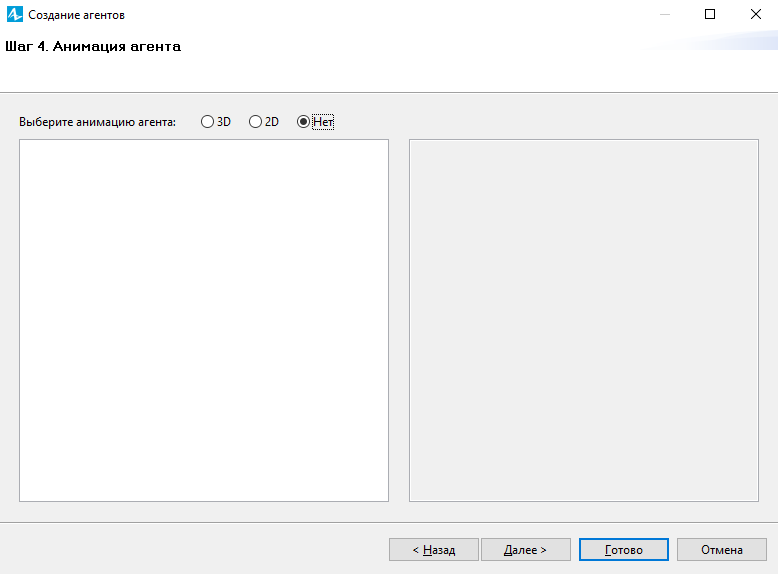
Для того чтобы наш автобус двигался по маршруту и останавливался на остановках, нам потребуется описать модель поведения автобуса, для этого мы создадим нового агента, назовем его **Маршрут**.



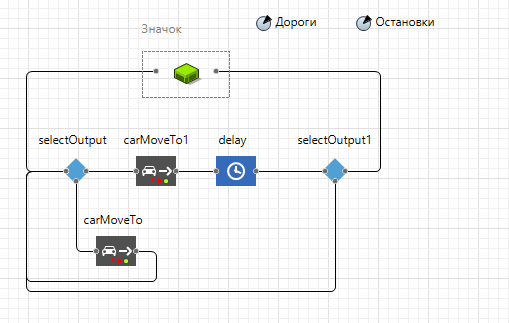




Никакой анимации у этого агента не будет.

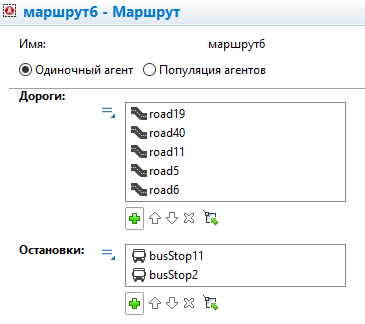


Модель поведения автобусов будет выглядеть следующим образом.

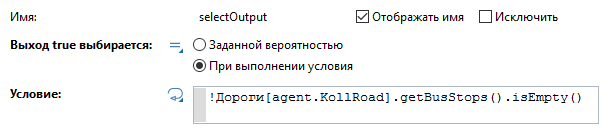


Идея заключается в том, что автобус будет двигаться по заданным дорогам и останавливаться на заданных остановках, которые заданы соответственно в параметрах **Дороги** и **Остановки** представляющие из себя одномерный массив.

Выглядит это так:

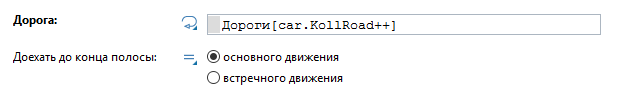


А теперь попорядку разберемся, что здесь происходит. Модель, обозначенная зеленым значком, имеет два порта, вход и выход, на вход к нам приходит автобус, далее он поступает в блок **selectOutput** раветвляется в зависимости от условия:

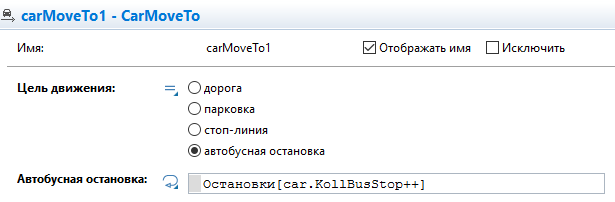


Как видим у нас появился параметр **KollRoad**, который мы создавали ранее в агенте **Bus**, этот параметр хранит в себе количество дорог который прошел автобус, **KollBusStop –** соответственно колличество остановок.

Если дорога на которую автобус собирается ехать имеет остановку, то мы переходим к блоку **CarMoveTo**, который отправляет автобус по дороге без остановок и увеличивает колличество проехавших дорог на единицу.



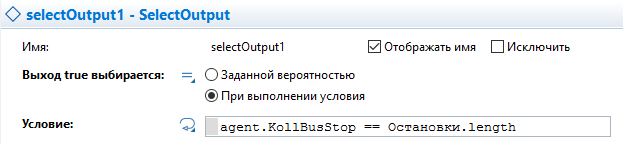
А если же дорога на которую автобус собирается ехать имеет остановку, то мы переходим к блоку **CarMoveTo1**, который в свою очередь отправляет автобус на остановку, и увеличивает колличество прошедших остановок на единицу.



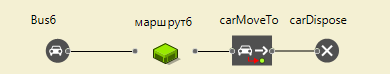
Затем идет блок **delay**, который заставляет автобус стоять на остановке некоторое колличество времени, которое вычисляется по триангулярному распределению.



После чего снова идет проверка на условие, если мы уже прошли все остановки, мы идем на выходи, если нет, то идем к превому условию и проходим все еще раз.



Что бы воспользоваться этим решением, выстраиваем вот такую конструкцию.



В параметрах **carSource** указываем где будет появляться автомобиль, в маршруте указываем по каким дорогам будет ехать автобус и места остановок где он будет останавливаться, по достижениею автобусом последней остановки, отправляем его на уничтожение.

# Заключение

В результате моделирования движения маршрутных транспортных средств так же можно собрать необходимую информацию о моделируемой системе, которая позволит предсказать рациональность или иррациональность проезда по тому или иному маршруту.

# Список используемой литературы

1. Мезенцев К.Н. Моделирование систем в среде AnyLogic.
2. Киселёва М.В. Имитационное моделирование систем в среде AnyLogic. - Екатеринбург: УГТУ-УПИ, 2009.
3. Карпов Ю. Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic. - СПб : БХВ-Петербург, 2006.
4. Рыжиков Ю. И. Имитационное моделирование. Теория и технологии. - СПб : М. : КОРОНА принт : Альтекс- А, 2004.