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](#_Toc472699096)

[Введение 4](#_Toc472699097)

[1. Теоретические основы моделирования систем в среде имитационного моделирования AnyLogic 5](#_Toc472699098)

[1.1 Средства AnyLogic для имитационного моделирования систем 7](#_Toc472699099)

[1.2 Средства описания поведения объектов 9](#_Toc472699100)

[1.3 Анимация поведения модели 9](#_Toc472699101)

[1.4 Пользовательский интерфейс 10](#_Toc472699102)

[1.5 Основные элементы построения модели 12](#_Toc472699103)

[2. Реализация модели движения маршрутных транспортных средств по установленному маршруту 15](#_Toc472699104)

[Заключение 25](#_Toc472699105)

[Список используемой литературы 26](#_Toc472699106)

# Аннотация

В ходе данной работы мы должны усвоить, что такое моделирование и научиться пользоваться средствами Anylogic для решения задач, связанных с имитационным моделированием.

# Введение

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

Отличительная черта моделирования - выделение основных свойств системы, интересующих разработчиков и исследователей, и их оценка (качественная и количественная) с учетом варьируемых поправок и ограничений. Именно это обстоятельство делает моделирование основным и необходимым этапом в разработке любых систем и изучении процессов и явлений реального мира.

Любая модель может быть формализована и изучена с помощью того или иного математического аппарата или метода. С появлением ЭВМ математические модели стали применяться практически во всех задачах моделирования, но появилась и альтернатива - имитационное моделирование, позволяющее на основе исходных заданных характеристик системы имитировать ее поведение во времени и получать необходимые характеристики.

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

Для достижения цели были поставлены следующие задачи:

- изучить литературу по теме исследования,

- изучить работу со средой Anylogic,

- разработать имитационную модель движения транспорта.

# 1. Теоретические основы моделирования систем в среде имитационного моделирования AnyLogic

Моделирование - метод решения задач, при использовании которого исследуемая система заменяется более простым объектом, описывающим реальную систему и называемым моделью. Моделирование применяется в случаях, когда проведение экспериментов над реальной системой невозможно или нецелесообразно, например, из-за высокой стоимости или длительности проведения эксперимента в реальном масштабе времени.

Имитационная модель - это компьютерная программа, которая описывает структуру и воспроизводит поведение реальной системы во времени. Имитационная модель позволяет получать подробную статистику о различных аспектах функционирования системы в зависимости от входных данных. Имитационное моделирование - разработка компьютерных моделей и постановка экспериментов на них. Целью моделирования в конечном счете является принятие обоснованных, целесообразных управленческих решений. Компьютерное моделирование становится сегодня 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 весь арсенал 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. В данном пособии описываются методы и приемы построения моделей с помощью инструментальной системы AnyLogic.

Пакет AnyLogic - отечественный профессиональный инструмент нового поколения, который предназначен для разработки и исследования имитацион-ных моделей. Разработчик продукта - компания "Экс Джей Текнолоджис" (XJ Technologies), г. Санкт-Петербург.был разработан на основе новых идей в области информационных технологий, теории параллельных взаимодействующих процессов и теории гибридных систем. Благодаря этим идеям чрезвычайно упрощается построение сложных имитационных моделей, имеется возможность использования одного инструмента при изучении различных стилей моделирования.

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

При разработке модели можно использовать элементы визуальной графики: диаграммы состояний (стейтчарты), сигналы, события (таймеры), порты и т.д.; синхронное и асинхронное планирование событий; библиотеки активных объектов.

Удобный интерфейс и многочисленные средства поддержки разработки моделей в AnyLogic делают не только использование, но и создание компьютерных имитационных моделей в этой среде моделирования доступными даже для начинающих. При разработке модели на AnyLogic можно использовать концепции и средства из нескольких классических областей имитационного моделирования: динамических систем, дискретно-событийного моделирования, системной динамики, агентного моделирования. Кроме того, AnyLogic позволяет интегрировать различные подходы с целью получить более полную картину взаимодействия сложных процессов различной природы. В данном пособии описываются три имитационные модели: дискретно-событийная, системно-динамическая и агентная. Для каждой модели приводится подробная постановка проблемы, разбирается структура модели, описывается процесс построения модели в среде AnyLogic и изучается ее поведение.

# 1.1 Средства AnyLogic для имитационного моделирования систем

Две фазы моделирования. AnyLogic используется для разработки имитационных исполняемых моделей и последующего их прогона для анализа. Разработка модели выполняется в графическом редакторе AnyLogic с использованием многочисленных средств поддержки, упрощающих работу. Построенная модель затем компилируется встроенным компилятором AnyLogic и запускается на выполнение. В процессе выполнения модели пользователь может наблюдать ее поведение, изменять параметры модели, выводить результаты моделирования в различных формах и выполнять разного рода компьютерные эксперименты с моделью. Для реализации специальных вычислений и описания логики поведения объектов AnyLogic позволяет использовать мощный современный язык Java. Активные объекты, классы и экземпляры активных объектов. Основными строительными блоками модели AnyLogic являются активные объекты, которые позволяют моделировать любые объекты реального мира. Класс в программировании является мощным средством, позволяющим структурировать сложную систему. Класс определяет шаблон, в соответствии с которым строятся отдельные экземпляры класса. Эти экземпляры могут быть определены 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 интерпретирует создаваемые графически классы активных объектов в классы Java, поэтому можно пользоваться 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 облегчает создание систем со сложной структурой, а также делает активные объекты повторно используемыми. Создав класс активного объекта, можно создать 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, стейтчартов и т.п. Каждый экземпляр активного объекта в работающей модели имеет свое собственное поведение, он может иметь свои значения параметров, функционирует независимо от других объектов, взаимодействуя с ними и с внешней средой. Визуальная разработка модели. При построении модели используются средства визуальной разработки (введения состояний и переходов стейтчарта, введения пиктограмм переменных и т.п.), задания численных значений параметров, аналитических записей соотношений переменных и аналитических записей условий наступления событий. Основной технологией программирования в AnyLogic является визуальное программирование - 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, определенные пользователем при разработке модели с помощью графического редактора, компилируются в конструкции языка Java, а затем происходит компиляция всей собранной программы на Java, задающей модель, в исполняемый код. Хотя программирование сведено к минимуму, разработчику модели необходимо иметь некоторое представление об этом языке (например, знать синтаксически правильные конструкции).

# 1.2 Средства описания поведения объектов

Основными средствами описания 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 (или стейтчарты) позволяют визуально представить поведение объекта во времени под воздействием событий или условий, они состоят из графического изображения состояний и переходов между ними (т.е. по сути это конечный автомат). Любая сложная логика поведения объектов модели может быть выражена с помощью комбинации стейтчартов, дифференциальных и алгебраических уравнений, переменных, таймеров и программного кода на Java. Алгебраические и дифференциальные уравнения записываются аналитически. Интерпретация любого числа параллельно протекающих процессов в модели AnyLogic скрыта от пользователя.

# 1.3 Анимация поведения модели

Имеет удобные средства представления функционирования моделируемой системы в живой форме динамической анимации, что позволяет "увидеть" поведение сложной системы. Визуализация процесса функционирования моделируемой системы позволяет проверить адекватность модели, выявить ошибки при задании логики. Средства анимации позволяют пользователю легко создавать виртуальный мир (совокупность графических образов, ожившую мнемосхему), управляемый динамическими параметрами модели по законам, определенным пользователем с помощью уравнений и логики моделируемых объектов. Графические элементы, добавленные на анимацию, называются динамическими, поскольку все их параметры: видимость, цвет и т.п. - можно сделать зависимыми от переменных и параметров модели, которые меняются со временем при выполнении модели.

# 1.4 Пользовательский интерфейс

После запуска AnyLogic открывается рабочее окно, в котором для продолжения работы надо создать новый проект или 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. Можно копировать, перемещать и удалять любые элементы дерева объектов, легко управляя рабочим проектом.



Рисунок 1. Окно проекта.

В редакторе AnyLogic для каждого выделенного элемента модели существует свое окно свойств, в котором указываются свойства (параметры) этого элемента. При выделении какого-либо элемента в окне редактора снизу появляется окно свойств, показывающее параметры данного выделенного элемента. Окно свойств содержит несколько вкладок. Каждая вкладка содержит элементы управления, такие как поля ввода, флажки, переключатели, кнопки и т.д., с помощью которых можно просматривать и изменять свойства элементов модели. Число вкладок и их внешний вид зависит от типа выбранного элемента.



Рисунок 2. Окно свойств.

**Окно палитры**содержит элементы (графические объекты), которые могут быть добавлены на структурную диаграмму. Элементы разбиты по группам, отображаемым на разных вкладках. Чтобы добавить объект палитры на диаграмму, щелкните сначала по элементу в палитре, а затем щелкните по диаграмме.

Активный объект может содержать переменные. Переменные могут быть либо внутренними, либо интерфейсными. Активный объект может иметь переменные, моделирующие, меняющиеся во времени величины. Переменные могут быть вынесены в интерфейс активного объекта и связаны с переменными других активных объектов. Тогда при изменении значения одной переменной будет немедленно меняться и значение связанной с ней зависимой переменной другого объекта. Этот механизм обеспечивает непрерывное и/или дискретное взаимодействие объектов.

Запускать и отлаживать модель можно с помощью меню Модель и панели инструментов:



Рисунок 3. Панель инструментов.

# 1.5 Основные элементы построения модели

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

 Source. Создает заявки. Обычно используется в качестве начальной точки потока заявок. Заявки могут быть либо базового для заявок класса Entity, либо любого класса пользователя, унаследованного от этого базового класса. Вы можете сконфигурировать объект так, чтобы он создавал заявки других типов, указав конструктор нужного класса в параметре Новая заявка, а также задать действие, которое должно выполняться перед тем, как новая заявка покинет объект, и связать с заявкой определенную фигуру анимации;

 SelectOutput. Объект направляет входящие заявки в один из двух выходных портов в зависимости от выполнения заданного (детерминистического или заданного с помощью вероятностей) условия. Условие может зависеть как от заявки, так и от каких-то внешних факторов. Поступившая заявка покидает объект в тот же момент времени;

Delay. Задерживает заявки на заданный период времени. Время задержки вычисляется динамически, может быть случайным, зависеть от текущей заявки или от каких-то других условий. Это время может, в частности, вычисляться как длина фигуры, заданной в качестве фигуры анимации этого объекта, поделенной на "скорость" заявки. Одновременно могут быть задержаны сразу несколько заявок (не более заданной вместимости объекта capacity);

Sink. Уничтожает поступившие заявки. Обычно используется в качестве конечной точки потока заявок. Для того, чтобы заявки удалялись из модели и уничтожались, нужно соединить выходной порт последнего блока процессной диаграммы с портом объекта Sink или Exit;

NetworkEnter. Используется при моделировании транспортных сетей. Регистрирует заявку в сети и помещает ее в заданный узел сети. После добавления в сеть заявка может перемещаться по сети и использовать сетевые ресурсы. Заявка не может одновременно находиться сразу в нескольких сетях, поэтому перед добавлением в другю сеть она должна быть вначале удалена из текущей сети с помощью объекта NetworkExit. Кроме того, Вы может задать скорость перемещения заявки по сети (которую впоследствии можно будет изменить). Операция добавления в сеть занимает нулевое время. Сразу после добавления в сеть заявка будет отображена на анимации (в случайном месте внутри начального узла);

 NetworkMoveTo. Используется при моделировании транспортных сетей. Перемещает заявку в новое место сети. Если к заявке присоединены какие-то ресурсы, то они перемещаются вместе с заявкой. При этом независимо от скорости ресурсов перемещаться такая группа из заявки и ее ресурсов будет со скоростью заявки. Время, которое заявка проведет в этом объекте, будет равно длине кратчайшего из возможных путей из текущего местоположения заявки в узел назначения, поделенной на скорость заявки (обратите внимание, что Вы можете динамически изменять эту скорость с помощью метода entity. setNetworkSpeed ()). Заявка будет отображаться на анимации сети движущейся вдоль выбранного маршрута.

NetworkExit. Используется при моделировании транспортных сетей. Удаляет заявку из сети. Заявка при этом перестает отображаться на анимации сети. При удалении заявки из модели (например, с помощью объекта Sink) она обязательно должна быть предварительно удалена из сети (если на момент удаления она в ней находится);

Network. Используется при моделировании транспортных сетей. Задает топологию сети и управляет сетевыми ресурсами. В одной модели может быть несколько сетей, и каждая сеть задается одним объектом Network. Топология сети задается группой фигур: прямоугольники задают узлы сети, а линии и ломаные - сегменты.

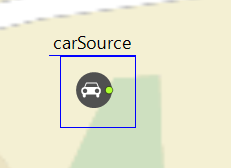
 Queue. Моделирует очередь заявок, ожидающих приема объектами, следующими за данным в потоковой диаграмме, или же хранилище заявок общего назначения.

# 2. Реализация модели движения маршрутных транспортных средств по установленному маршруту

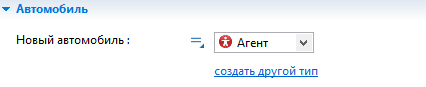
И так, для начала создадим проект и набросаем нашу дорожную сеть с автобусными остановками, перекрестками и светофорами.



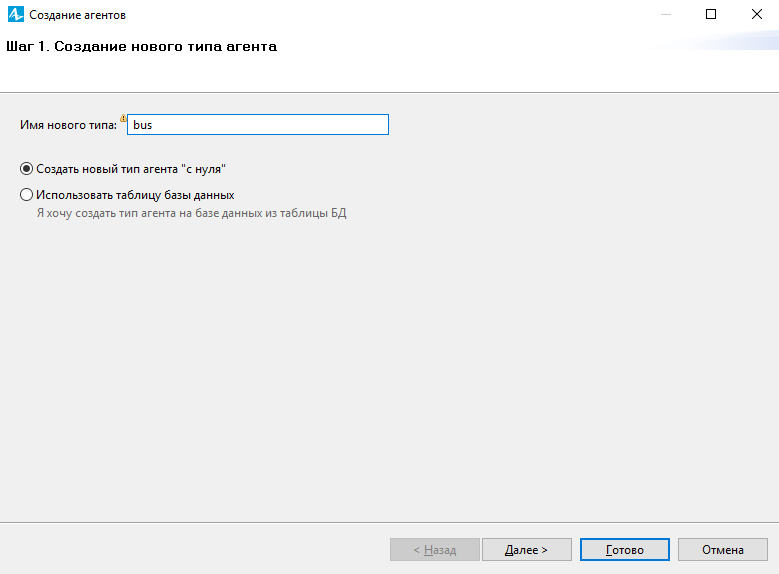
Далее добавим элемент CarSource из библиотеки дорожного движения и создадим новый тип агента для автобусов, агента назовем Bus.

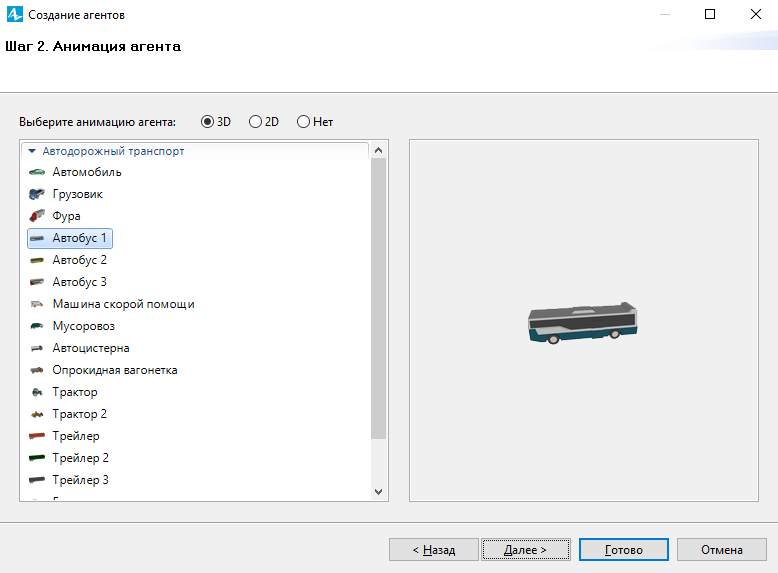


Для этого откроем свойства CarSource и во вкладке Автомобиль

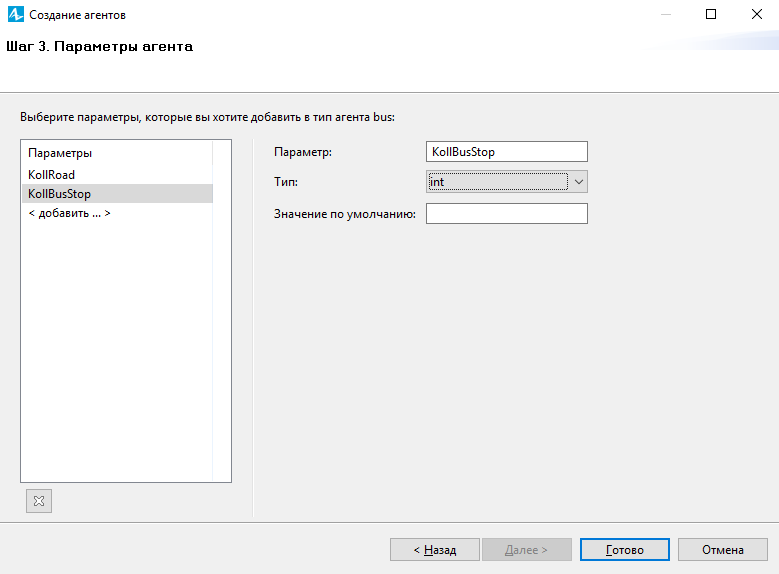


Нажмем создать другой тип





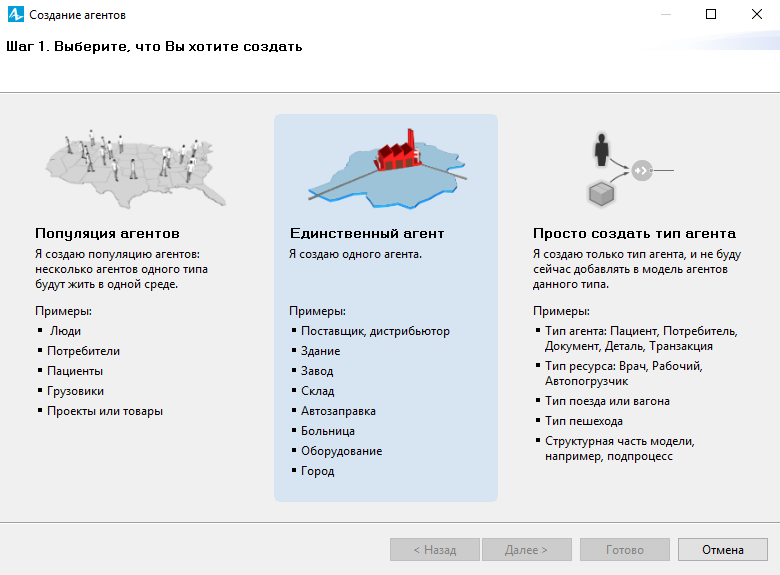
Добавим два параметра **KollRoad** и **KollBasStop**, тип у них будет целочисленный, о которых мы поговорим чуточку позже, после чего нажимаем Готово.

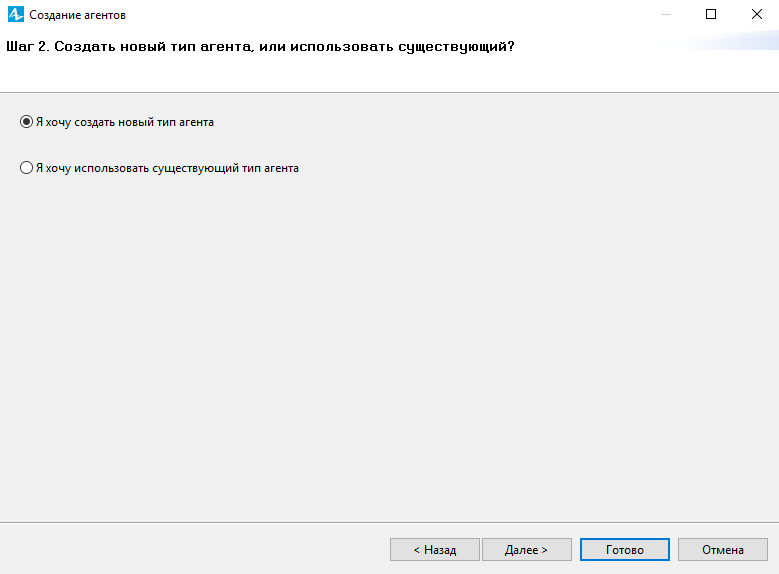


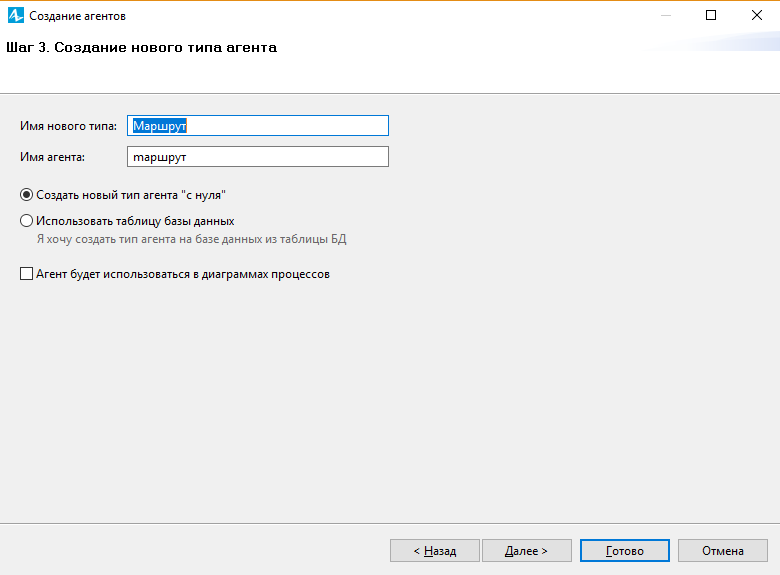
Теперь в свойствах CarSource мы можем выбрать созданного нами агента Bus.



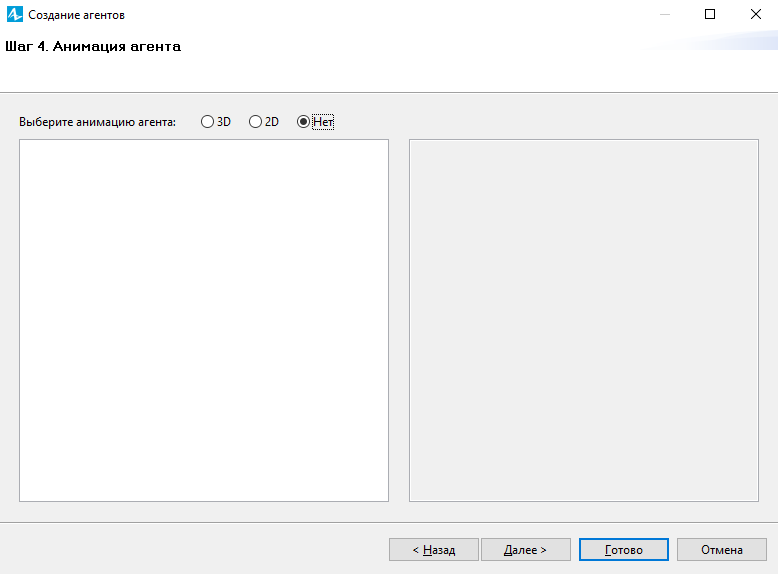
Для того чтобы наш автобус двигался по маршруту и останавливался на остановках, нам потребуется описать модель поведения автобуса, для этого мы создадим нового агента, назовем его **Маршрут**.



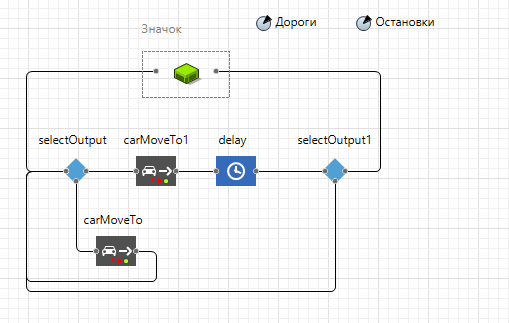




Никакой анимации у этого агента не будет.

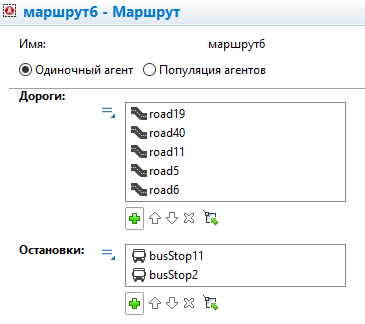


Модель поведения автобусов будет выглядеть следующим образом.

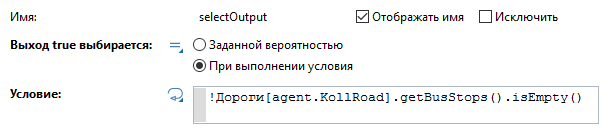


Идея заключается в том, что автобус будет двигаться по заданным дорогам и останавливаться на заданных остановках, которые заданы соответственно в параметрах **Дороги** и **Остановки** представляющие из себя одномерный массив.

Выглядит это так:

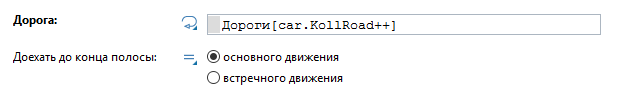


А теперь попорядку разберемся, что здесь происходит. Модель, обозначенная зеленым значком, имеет два порта, вход и выход, на вход к нам приходит автобус, далее он поступает в блок **selectOutput** раветвляется в зависимости от условия:

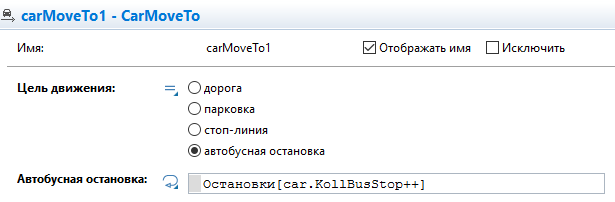


Как видим у нас появился параметр **KollRoad**, который мы создавали ранее в агенте **Bus**, этот параметр хранит в себе количество дорог который прошел автобус, **KollBusStop –** соответственно колличество остановок.

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.



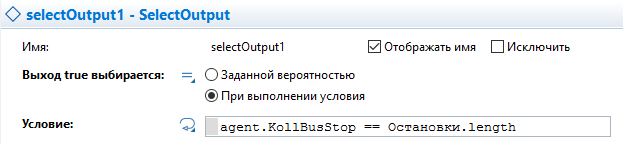
А если же дорога на которую автобус собирается ехать имеет остановку, то мы переходим к блоку **CarMoveTo1**, который в свою очередь отправляет автобус на остановку, и увеличивает колличество прошедших остановок на единицу.



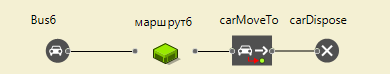
Затем идет блок **delay**, который заставляет автобус стоять на остановке некоторое колличество времени, которое вычисляется по триангулярному распределению.



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 так же возможен сбор необходимой информации о моделируемой системе, что позволяет предсказывать рациональность или иррациональность проектируемой модели или выявить ошибки в проектировании системы.

# Список используемой литературы

1. К.Н. Мезенцев - Моделирование систем в среде AnyLogic <http://www.anylogic.ru/mezenczev>.

2. М.В. Киселёва - Имитационное моделирование систем в среде AnyLogic <http://www.anylogic.ru/upload/Books\_ru/Kisileva\_Simulation\_Modeling\_In\_AnyLogic.pdf>.

3. Карпов Ю. - Имитационное моделирование систем. Введение в моделирование с AnyLogic.