С. А. Олейникова

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Практикум

Печатается по решению Ученого Совета факультета информационных систем и компьютерной безопасности Воронежского государственного технического университета Протокол №1 от 21 сентября 2022 г.

Воронеж
Издательство «Научная книга»
2022

УДК 519.6(075.8)
ББК 32.973
О 53

Рецензенты:
кафедра математических методов исследования операций Воронежского государственного университета (зав. кафедрой д-р техн. наук, профессор Т. В, Азарнова);

д-р техн. наук, профессор В. Ф. Барабанов

О 53 Олейникова, С. А. Математическое моделирование: Практикум/ С. А. Олейникова. - Воронеж: Издательство «Научная книга», 2022. - 80 с.

ISBN 978-5-98222-996-0

Практикум содержит теоретические сведения и варианты заданий для лабораторных работ по курсу «Моделирование вычислительных систем». Основное внимание уделено математическим и имитационным методам построения моделей.

Предназначено для студентов направления 09.03.01 «Информатика и вычислительная техника» (профиль «Вычислительные машины, комплексы, системы и сети») очной и заочной форм обучения.

Ил. 70. Табл. 5. Библиогр.: 17 назв.

УДК 519.6(075.8)
ББК 32.973
О 53

ISBN 978-5-98222-996-0

ВВЕДЕНИЕ

Для эффективного функционирования сложных обслуживающих и производственных систем необходимо обеспечить возможность получения разнообразных стохастических характеристик. К таким характеристикам могут относиться средняя длина очереди, среднее время отклика на запрос, загрузка устройств и т.д.

Наличие сложной структуры исследуемой системы с большим количеством связей между своими элементами, а также интенсивное развитие информационных технологий обуславливает применение имитационных методов моделирования для решения данной задачи. Современные среды моделирования отличаются не только возможностью получения необходимой для пользователя информации о характеристиках системы, но развитым графическим интерфейсом, позволяющим, при необходимости, отобразить процесс функционирования систем и процесс их обслуживания в динамике. Это делает аппарат имитационного моделирования неотъемлемой частью системы управления предприятием или организацией, позволяющей получать необходимую информацию в оперативном режиме.

Данный практикум посвящен вопросам связанным с построением математических моделей сложных вычислительных и обслуживающих систем. В первую очередь, это внимание акцентируется на имитационных моделях как разновидности математического моделирования как. Предполагается, что для реализации имитационных моделей будет использована среда Anylogic, как мощный инструмент для разработки моделей любой сложности, отличающийся от своих аналогов не только развитым графическим интерфейсом, позволяющим визуализировать процессы, происходящие в исследуемых объектах, моделировать различные виды обслуживания (железнодорожные, автомобильные перевозки, пешеходное движение и т.д.), но и имеющем в своем арсенале новейшие библиотеки, позволяющие использовать агентный подход для моделирования сложных распределенных систем.

Практикум состоит из 8 частей, каждая из которых содержит описание соответствующей лабораторной работы. Каждая часть содержит необходимые для выполнения работы теоретические сведения, а также задание к работе.

Практикум может быть полезен не только обучающимся по направлению 09.03.01 (Информатика и вычислительная техника), но и любым студентам, желающим освоить среду AnyLogic для построения имитационных моделей.

ЛАБОРАТОРНАЯ РАБОТА № 1
СОЗДАНИЕ ПРОСТЕЙШЕЙ ОБСЛУЖИВАЮЩЕЙ МОДЕЛИ В ANYLOGIC

Цель лабораторной работы:

Знакомство со средой имитационного моделирования AnyLogic путем создания элементарной модели.

1.1.Краткие теоретические сведения

1.1.1 Структура простейшей модели

Среда AnyLogic является современным приложением, позволяющим формировать модели любой сложности и получать необходимую статистическую информацию. Внешний вид приложения представлен на *рис. 1.1*.

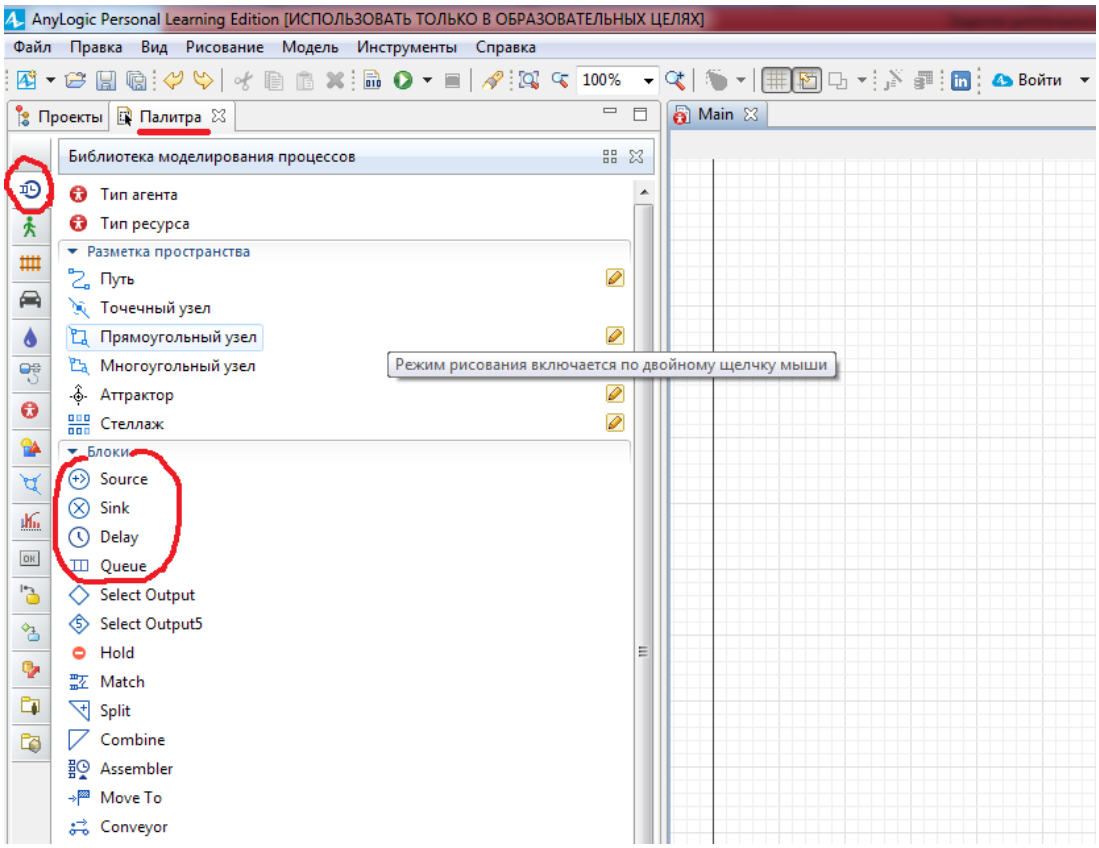


Рис. 1.1. Требуемые компоненты

В левой части представлены все библиотеки, которые доступны пользователю. При выборе очередной библиотеки (в данном случае, библиотеки моделирования процессов), получаем доступ ко всем объектам

данной библиотеки. Процесс моделирования заключается в последовательном помещении необходимых объектов в рабочую область и настройки их свойств.

Простейшая обслуживающая система включает в себя:

- входящий поток заявок;
- обслуживающее устройство (одно или несколько);
- поток обслуженных заявок (и/или поток заявок, получивших отказ).

Примерами элементарных обслуживающих систем можно являются работа банкомата, автозаправочной станции, любого магазина и т.д. Рассмотрим процесс моделирования простейших моделей средствами Anylogic. На первом этапе сформируем саму модель. Для этого потребуется вкладка «**Библиотека моделирования процессов**» из Палитры.

Поток заявок создается с помощью блока **Source** (источник). Нажав мышью соответствующий блок, необходимо перетащить его в рабочую область. После добавления любого объекта необходимо настроить его свойства. В частности, для блока Source необходимо определить, через какое время в систему будут поступать заявки. Anylogic дает возможность задать поступление с помощью следующих основных способов:

- интенсивности поступления (например, 5 заявок в минуту);
- временем между двумя последовательно поступающими в систему заявками. Это время может быть постоянной или случайной величиной (специфика случайных величин, описывающих длительность обслуживания, будет представлена позже).

Обслуживающее устройство состоит из заданного количества идентичных каналов обслуживания (например, 3 кассы магазина и т.д.). В случае, если для заявки неважно, какой именно канал будет занят ее обслуживанием, систему можно смоделировать одним устройством. Если же между отдельными каналами для заявки есть какие-то различия (например, разное время обслуживания или выбор одного из обслуживающих устройств только в том случае, если другое занято и т.д.), то необходимо использовать несколько объектов типа «Обслуживающее устройство» и специальным образом организовывать логику модели (проверки условий и т.д.). Режим функционирования любого обслуживающего устройства следующий: если в момент поступления заявки хотя бы один канал обслуживания свободен, то заявка принимается к обслуживанию, в противном случае - заявка ждет перед устройством момента, когда оно освободится, а потом занимает освободившийся канал обслуживания. Очевидно, что в этом случае возникает очередь. Для того, чтобы собрать статистику об ожидании, можно воспользоваться блоком **queue** (очередь). В качестве свойств этого блока можно указать количество мест, на которое эта очередь рассчитана (в случае, если у очереди ограничена длина). Поместив блок queue из палитры на рабочую область,

видим, что от блока source к блоку queue образовалась связь (прямая линия). Далее помещаем блок, имитирующий функционирование устройств. Это блок **delay**. Его основным свойством является число каналов обслуживания. После обслуживания заявки должны покидать систему. В Anylogic это осуществляется посредством блока **sink**. В результате получается следующая структура модели (*рис. 1.2*).

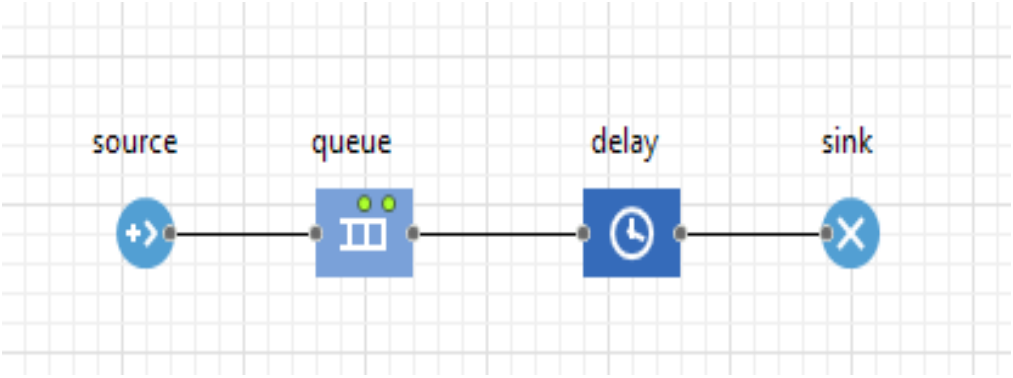


Рис. 1.2. Пример простейшей модели

На этом первый этап моделирования завершен. После него необходимо приступить к так называемому прогону модели. Эта имитация процесса прохождения заявок по всем заданным блокам в соответствии с имеющимися свойствами этих блоков. Нажав кнопку F5, переходим к прогону. В процессе прогона пользователю отображается движение заявок от одного блока к другому с указанием числа заявок, вошедших в каждый блок (*рис. 1.3*).

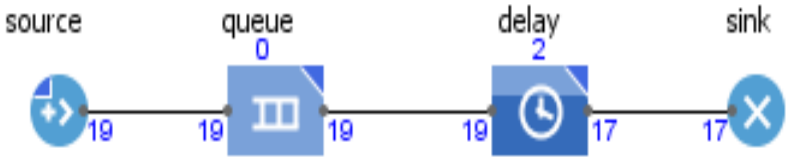


Рис. 1.3. Прогон модели

Из данного рисунка видно, что блоком Source было создано 19 заявок, 19 заявок входили в очередь и 19 заявок вышли из нее (т.е. в данный момент очередь пуста); 19 заявок посетили блок delay, имитирующий обслуживающее устройство, а вышли из него 17 заявок; 2 заявки в данный момент находятся в этом устройстве. Все 17 заявок, покинувших устройство delay, были уничтожены блоком sink и покинули систему.

Щелкнув по любому объекту модели, можно увидеть статистику о его функционировании. На *рис. 4* представлена статистика о работе трехканального устройства.

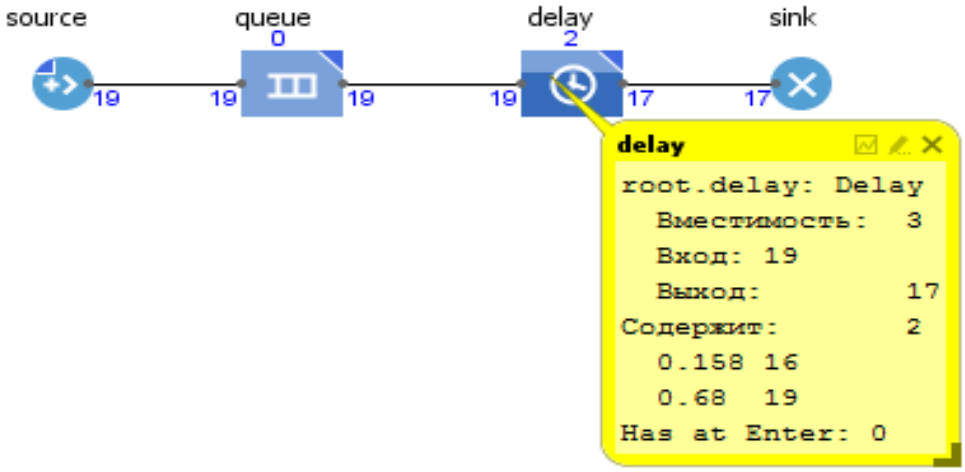


Рис. 1.4. Статистика о работе устройства

По умолчанию после запуска процесс имитации будет происходить до тех пор, пока не будет нажата кнопка «Стоп». Однако, можно задать остановку процесса по прошествии некоторого времени. Для этого необходимо перейти на вкладку «Проекты» и выбрать Simulation (Main). Далее задать необходимое модельное время. Есть возможность выбора режима виртуального или реального времени. В режиме виртуального времени будет осуществляться прогон настолько быстро, насколько это возможно. В режиме реального времени задается связь модельного и реального времени. Задав конечное время и из выпадающего списка остановить выбрав «В заданное время», получим модель, которая проработает заданное время, после чего остановится. После этого уже имеет смысл анализировать статистику.

1.1.2. **Задание длительности обслуживания**

Как правило, длительность обслуживания представляет собой случайную величину. В этом случае для ее задания необходимо использовать одну из наиболее подходящих вероятностных функций. Наиболее часто используются следующие распределения из теории вероятностей.

1. **Равномерное распределение**. Это означает, что время обслуживания может быть с одинаковой вероятностью любой величиной из некоторого временного интервала. График случайной величины, распределенной по равномерному закону в интервале [a,b], имеет вид (*рис. 1.5*).

Например, если сказано, что длительность выполнения некоторой работы составляет 25±3 с, то это означает, что соответствующая случайная величина имеет равномерный закон распределения в интервале [22,28].

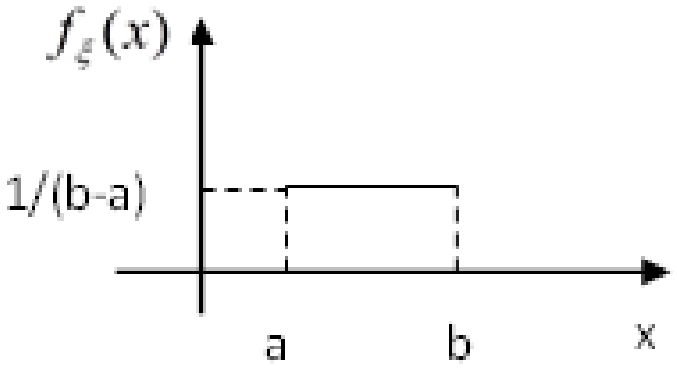


Рис. 1.5. Случайная величина, распределенная по равномерному закону

Для моделирования равномерного закона распределения используется функция ***Uniform***, имеющая два параметра: нижнюю и верхнюю границы возможных значений случайной величины.

2. **Экспоненциальное распределение**. Данное распределение является одним из самых распространенных для моделирования простейших обслуживающих систем. Например, с помощью экспоненциального закона можно смоделировать случайную величину, описывающую длительность обслуживания клиента на кассе, на автозаправочной станции и т.д. График случайной величины, распределенной по экспоненциальному закону, имеет вид (*рис.6*).

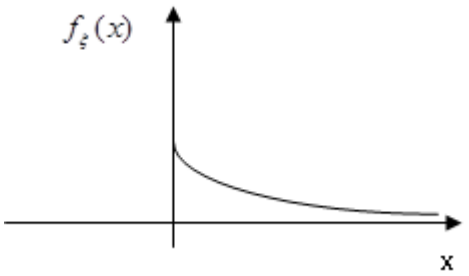


Рис. 1.6. Экспоненциальный закон распределения

Экспоненциальный закон имеет единственный параметр λ, обозначающий интенсивность обслуживания. Интенсивность и длительность обслуживания связаны между собой обратной зависимостью:

$$T_{обсл}=1/\lambda. \qquad (1.1)$$

Для моделирования случайной величины, распределенной по экспоненциальному закону, используется функция ***exponential***. Например, если известно, что длительность обслуживания распределена экспоненциально и составляет в среднем 2 мин, то необходимо воспользоваться функцией exponential(0.5).

3. **Нормальный закон распределения**. Нормальный закон распределения является самым распространенным законом в теории вероятностей. Он также достаточно часто используется для моделирования случайных величин. График нормально распределенной случайной величины имеет следующий вид (*рис.1.7*).

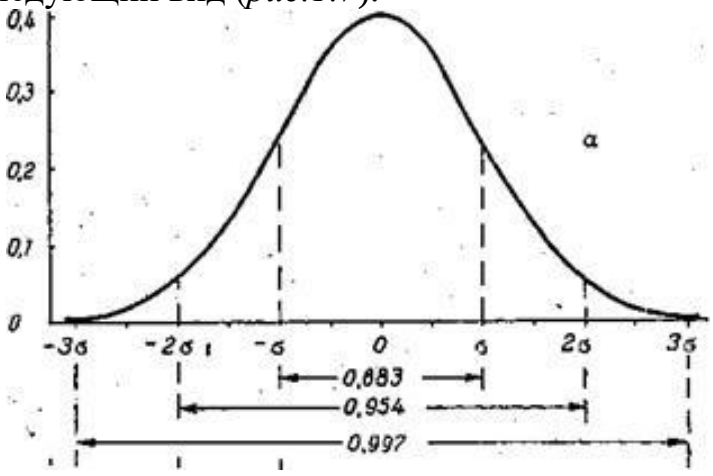


Рис. 1.7. Нормальный закон распределения

Случайная величина, распределенная по нормальному закону, задается двумя параметрами: ожидаемым значением a (математическим ожиданием) и стандартным отклонением σ (корнем из дисперсии). В Anylogic нормальное распределение задается функцией ***normal(σ, a)***, где первый параметр – стандартное отклонение; второй – ожидаемое значение.

4. **Другие законы**. В Anylogic существует возможность задания большого числа случайных величин. В частности, может потребоваться треугольное (triangle), бета (beta) и другие распределения. Для получения справочной информации обо всех поддерживаемых распределениях необходимо в справке набрать «вероятностные распределения». Выбрав нужную функцию, можно получить справку о ней и ее параметрах.

1.1.3. Формирование условий

Как правило, ни одна модель не обходится без различных «условий», в результате выполнения или невыполнения которых заявка будет двигаться по одному или другому маршруту. В Anylogic существует блок **Select Output** для формирования таких условий. У блока есть основное свойство «условие». Если оно выполнено, то заявка после этого блока попадает на порт OutT (сокращение от Out True), если не выполнено – то на порт – OutF (Out False). Работа блока Select Output может осуществляться в двух

режимах: в режиме условия и в вероятностном режиме. В режиме условия осуществляется проверка условия и, если оно выполнено, заявка будет направлена на порт OutT, если нет – на порт OutF. В вероятностном режиме заявка с заданной вероятностью будет направлена на порт OutT, с оставшейся - в блок OutF. Режим выбирается с помощью переключателей «Выход true выбирается».

Например, необходимо сформировать условие, в результате которого заявка поступит на обслуживание в том случае, если хотя бы один канал пятиканальной системы свободен и получит отказ в противном случае. В этом случае в поле «условие» блока Select Output надо написать условие:

delay.size()<5

Здесь size – это обращение к свойству блока delay, которое показывает текущее число занятых каналов. После этого порт OutT необходимо связать с устройством delay, а порт OutF – с блоком Sink, уничтожающим заявки.

Аналогичным образом можно создать систему с ограниченной очередью. Пусть, к примеру, в случае, если в очереди находится более трех заявок, поступившая заявка получает отказ. Содержимое «условия» блока Select Output будет тогда следующим:

queue.size()<3

Как было отмечено ранее, кроме условного режима, блок Select Output может работать в так называемом вероятностном режиме. В этом случае появляется поле вероятность, и с указанной вероятностью A заявка попадет на порт OutT. С вероятностью 1-A заявка будет направлена на порт OutF.

1.2. Задание к работе

Смоделировать работу устройства с заданным числом каналов в соответствии с вариантом задания в течение 8 часов модельного времени. Найти среднюю длину очереди и среднее число занятых каналов. Предполагается, что время поступления и время обслуживания распределено экспоненциально.

Таблица 1.1

Варианты заданий

№ п/п	Число каналов	Время поступления	Время обслуживания	Число мест в очереди
1	3	Через 20 с	60 с	3
2	4	Через 20 с	80 с	3
3	5	Через 20 с	100 с	4

4	3	Через 30 с	90 с	2
5	4	Через 10 с	45 с	4
6	5	Через 30 с	160 с	5
7	3	2 заявки в мин	1,5 мин	4
8	4	2 заявки в мин	2 мин	3
9	5	2 заявки в мин	2,5 мин	5
10	3	4 заявки в мин	45 с	4
11	4	4 заявки в мин	1 мин	3
12	5	4 заявки в мин	1,5 мин	5

Отчет должен содержать:
- постановку задачи;
- описание формирования модели;
- результаты моделирования.

ЛАБОРАТОРНАЯ РАБОТА № 2
СОЗДАНИЕ МОДЕЛЕЙ С ПОМОЩЬЮ БИБЛИОТЕКИ МОДЕЛИРОВАНИЯ ПРОЦЕССОВ

Цель лабораторной работы:

Освоение библиотеки моделирования процессов для разработки моделей сложных технических систем.

2.1. Краткие теоретические сведения

2.1.1. Дополнительные сведения об объекте Queue

Объект Queue моделирует очередь заявок, ожидающих приема объектами, следующими за ним в потоковой диаграмме (или моделирует хранилище заявок). При необходимости можно задать максимальное время ожидания, а также извлекать заявки из любых позиций в очереди.

Заявка может покинуть объект Queue четырьмя различными способами:

1. Обычным способом через порт Out, когда объект, следующий за объектом Queue, готов принять заявку.
2. Через порт TimeOut, если заявка проведет в очереди заданное количество времени. Для моделирования очередей с ограничением на время ожидания в свойствах объекта Queue в категории «Специфические»

необходимо включить флаг «Разрешить уход по таймауту» и задать время ожидания (*рис. 2.1*).

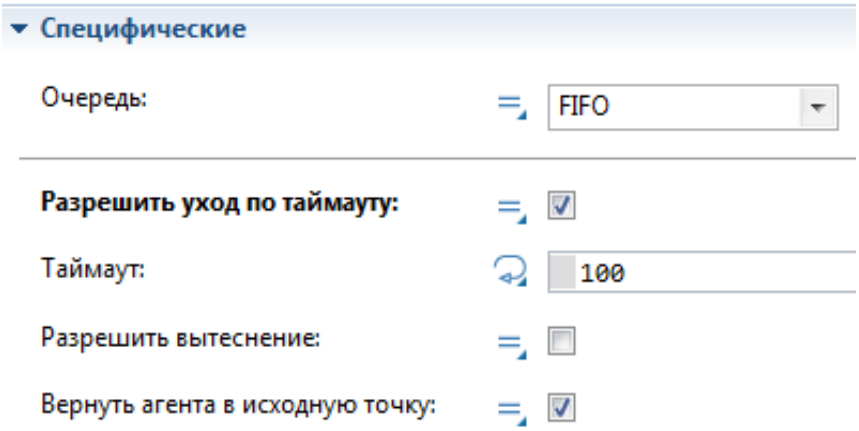


Рис. 2.1. Моделирование очереди с ограничением на время

3. Через порт OutPreempted, будучи вытесненной другой заявкой при заполненной очереди. Таким образом моделируются очереди ограниченной длины. При этом необходимо включить флаг «Разрешить вытеснение» (*рис. 2.2*).

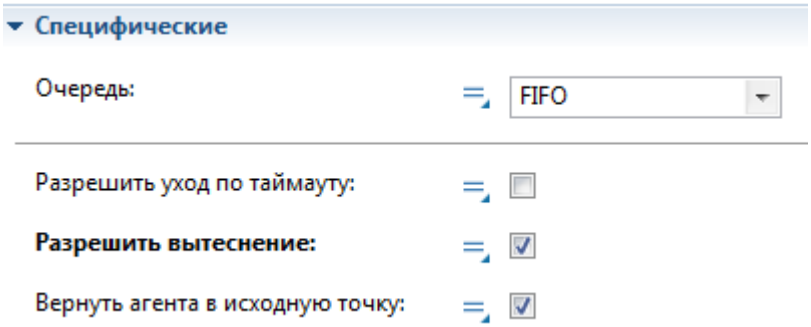


Рис.2.2 Моделирование очереди с ограничением на количество

4. «Вручную», путем написания программного кода (функции remove() или removeFirst()).

Порты объекта Queue представлены на рис. 2.3.

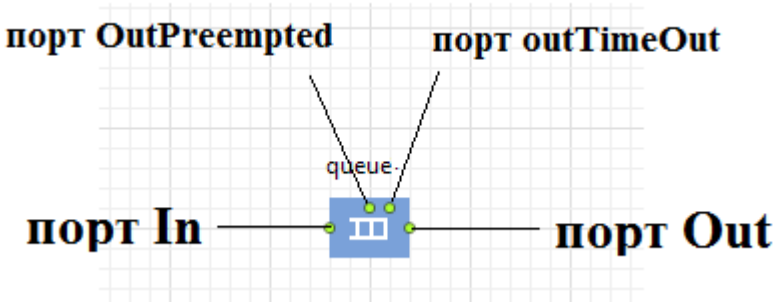


Рис.2.3. Порты объекта Queue

Существует возможность организации различных режимов ожидания. По умолчанию заявки помещаются и извлекаются из очереди согласно правилу FIFO (первый пришел – первый обслужен). Возможна также организация очередей согласно правилу LIFO (последний пришел – первый обслужен), либо по приоритету заявки. В этом случае в случае занятости очереди заявка может быть вытеснена лишь в том случае, когда приоритет новой поступившей заявки ВЫШЕ приоритета находящейся в очереди заявки.

Выбрать дисциплину ожидания можно в категории «Специфические» свойств объекта Queue (*рис. 2.4*).

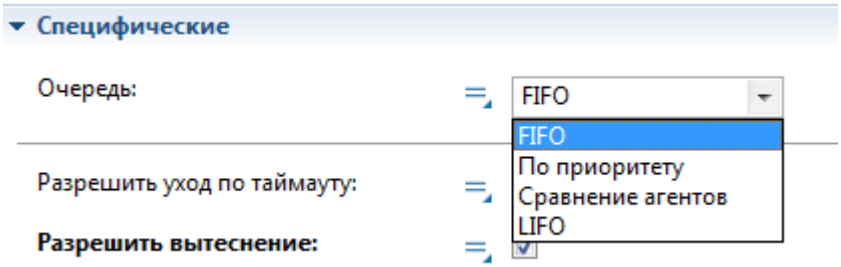


Рис.2.4. Выбор дисциплины ожидания

2.1.2. **Создание простейшей анимации**

Anylogic позволяет добавлять к модели схематическую анимацию интересующих объектов. Анимация модели рисуется в той же диаграмме, в которой задается диаграмма моделирующего процесса. На вкладке «Разметка пространства» палитры представлен ряд компонентов, позволяющих изображать элементарные процессы. В частности, для иллюстраций можно использовать как примитивы (прямоугольник, путь и т.д.), так и более специфические иллюстрации (дороги, ж/д пути и т.д.).

Чтобы связать компоненты схемы моделирования процессов с той или иной частью иллюстрации, необходимо у данного компонента выбрать соответствующий объект в раскрывающимся списке «Место агентов». Например, если одноканальное устройство будет изображено прямоугольником (node), то надо выполнить следующие действия (*рис. 2.5*).

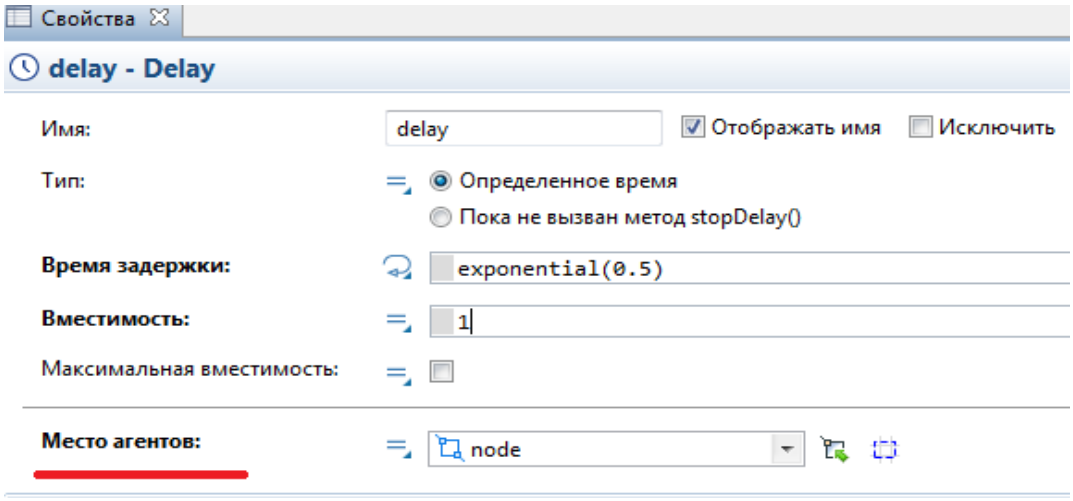


Рис.2.5. Связывание одноканального устройства с узлом

Если необходимо по ходу моделирования менять значения тех или иных свойств в зависимости от каких-либо условий, то можно ввести в поле соответствующего динамического свойства выражение, которое будет постоянно вычисляться заново при выполнении модели. Например, если необходимо, чтобы менялся цвет фигуры, в зависимости от занятости устройства (занято – красный, свободно - зеленый), то в поле «Цвет заливки» необходимо щелкнуть по стрелке (сразу после знака «=») и выбрать «Динамическое значение» (рис. 2.6).

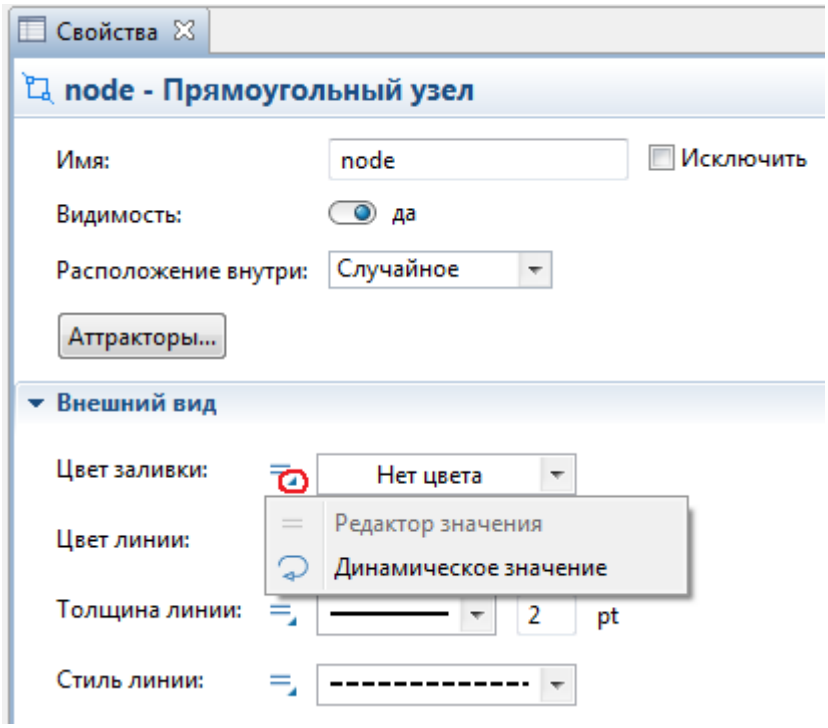


Рис.2.6. Рисунок 6 – Динамическое изменение цвета

В открывшейся строке для одноканального устройства ввести условие:

delay.size()>0?red:green

Если условие выполнено, то производится окраска цветом, описанным перед знаком «:», в противном случае – после него. delay.size() – число занятых каналов (если оно больше 0, то одноканальное устройство занято).

2.1.3. Параметры модели

Любой активный объект может иметь параметры. Параметры обычно используются для задания характеристик объекта или расчета каких-либо величин (счетчики и т.д.). Значения параметров можно при необходимости менять во время работы модели.

Элемент параметры находится на вкладке «Основная» Палитры. Перетащив его на диаграмму класса Main, можно задать его значение как сразу (в свойствах поле «Значение по умолчанию»), так и изменять его во время моделирования.

Пример. Рассмотрим возможность изменения интенсивности поступления заявок в процессе моделирования.

Создадим параметр intens и зададим ему значение 10 (*рис. 2.7*).

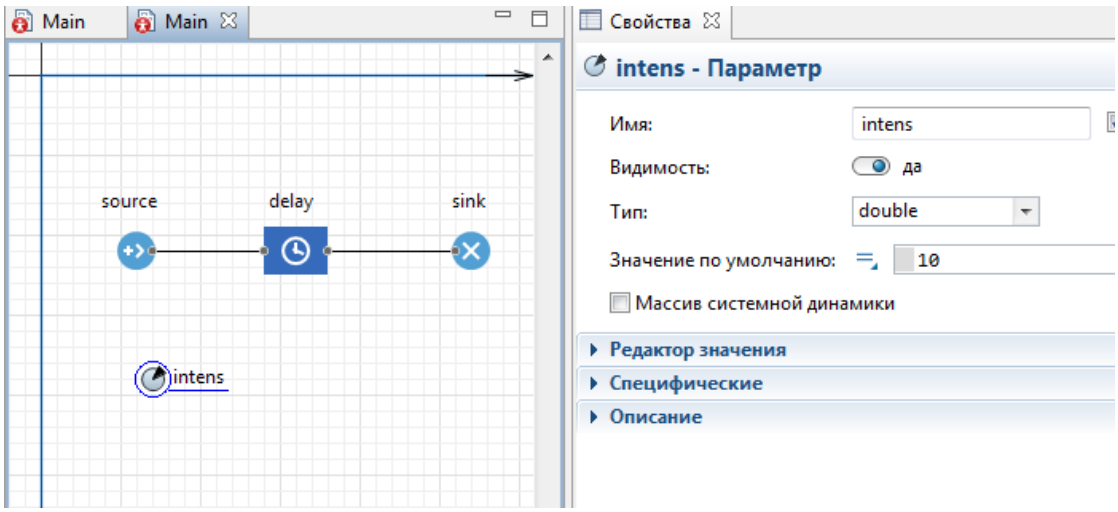


Рис.2.7. Задание свойств параметру

Объект «Delay» сделаем 6-канальным с временем обслуживания – 1 с (время распределено экспоненциально).

В свойстве «Интенсивность прибытия» объекта source впишем параметр intens.

Из вкладки Палитры «Элементы управления» перетащим в рабочую область объект «Бегунок».и свяжем его с параметром intens (*рис. 2.8*).

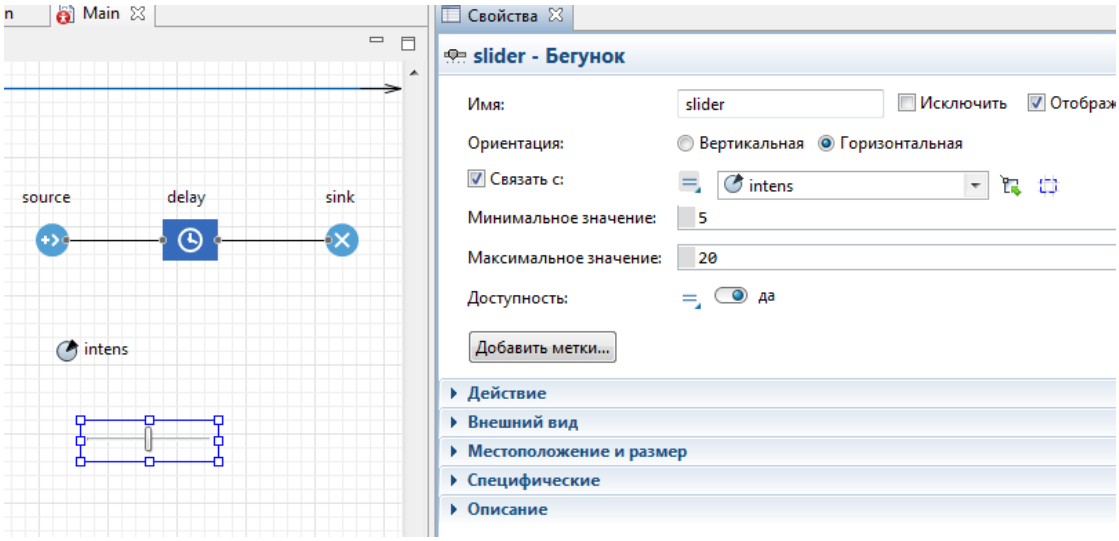


Рис.2.8. Бегунок

Добавим объект Select Output для отказа в случае занятости устройства delay. Условие будет иметь вид:

Delay.size()<6

В результате получим следующую модель (рисунок 9). Варьируя значения бегунка, будем изменять интенсивность входного потока.

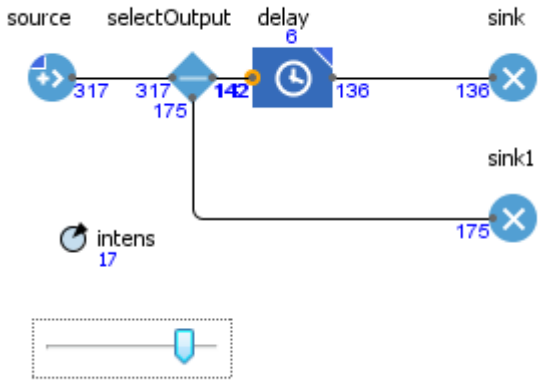


Рисунок 9 – Моделирование с переменной интенсивностью

Возможна также корректировка значений параметров в самой модели. Предположим, что если заявка получила отказ, то интенсивность потока снижается до 5 заявок в с, а в случае успешного обслуживания – увеличивается до 15 заявок в с.

В свойствах объекта Sink необходимо прописать действия, совершаемые при входе. Выбрав соответствующую строку, запишем.

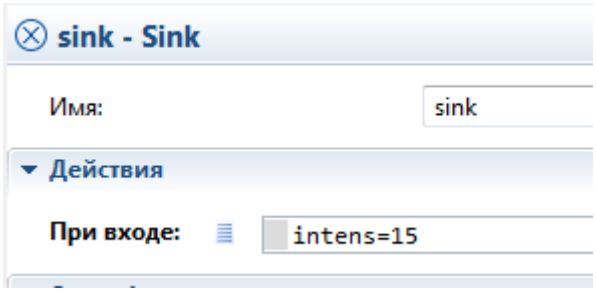


Рис.2.10. Изменение значений параметра при входе в блок Sink

Аналогичным образом для случая попадания заявки в объект Sink1 значению параметра intens присваивается 5.

2.1.4. Статистика

Чтобы автоматически получать некоторые статистические данные, можно использовать вкладку «статистика». В частности, на данной вкладке есть элемент «statistics», который может собрать статистику о любой величине. Если, например, необходимо найти среднее число занятых каналов многоканального устройства, смоделированного с помощью блока delay, то достаточно поместить элемент statistics в любую свободную область рабочего окна и связать с требуемой характеристикой следующим образом (*рис. 2.11*).

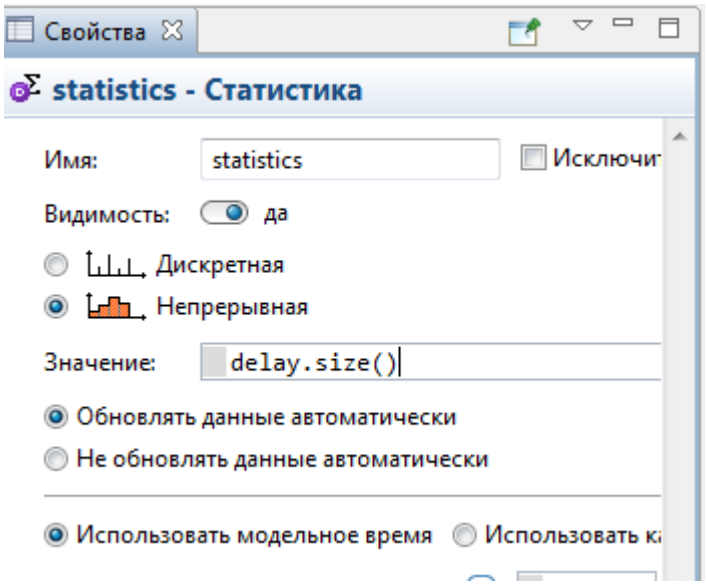


Рис. 2.11. Элемент «Статистика»

2.2. Задание к работе

На вход системы поступает поток заявок с интенсивностью k. Система состоит из двух обслуживающих устройств У1 и У2. Пришедшая заявка с вероятностью p поступает на устройство У1 и с вероятностью 1-p – на устройство У2. Устройство У1 имеет n_1 каналов обслуживания и допускает очередь, которая ограничена m местами. Устройство У2 имеет n_2 каналов обслуживания и допускает очередь, время пребывания в которой ограничено значением t. В случае поступления заявки в очередь 1 при ее занятости или пребывании в очереди 2 более t единиц времени, заявка получает отказ. Время обслуживания в устройствах У1 и У2 распределено экспоненциально со средними значениями t_1 и t_2.

Смоделировать работу системы. Определить вероятность отказа для первого и второго устройства, а также среднюю длину каждой очереди и среднее число занятых каналов обслуживания для каждого устройства.

* Добавить в модель два семафора, сигнализирующих об отказе обслуживания на первом и втором устройстве (в обычном режиме зеленый цвет, в случае отказа загорается красный цвет).

** Для снижения уровня отказов ввести пороговые вероятности p_1 и p_2 для первого и второго устройств соответственно. При достижении порогового значения время обслуживания сокращается до t_3 и t_4 соответственно.

Все числовые значения выбрать самостоятельно.

ЛАБОРАТОРНАЯ РАБОТА № 3

СБОР СТАТИСТИКИ И ВИЗУАЛИЗАЦИЯ РЕЗУЛЬТАТОВ МОДЕЛИРОВАНИЯ

Цель работы: изучение подходов для сбора статистики и анализа результатов моделирования

3.1. Краткие теоретические сведения

AnyLogic предоставляет пользователю удобные средства для сбора статистики по работе любых блоков. Объекты Библиотеки моделирования процессов самостоятельно производят сбор основной статистики. Для этого просто необходимо включить сбор статистики для интересующего объекта.

Столбиковая диаграмма

Столбиковая диаграмма отображает несколько элементов данных в виде столбцов, «растущих» в заданном направлении от базовой линии. Размеры столбцов пропорциональны значениям соответствующих элементов данных. Для добавления столбиковой диаграммы необходимо

перетащить соответствующий элемент из палитры **Статистика** в то место графического редактора, где необходимо нарисовать диаграмму. Далее необходимо связать диаграмму с исследуемым объектом. Для этого необходимо перейти в секцию **Данные** панели **Свойства** и щелкнуть мышью по кнопке  **Добавить элемент данных**. При этом откроется диалоговое окно (*рис. 3.1*).

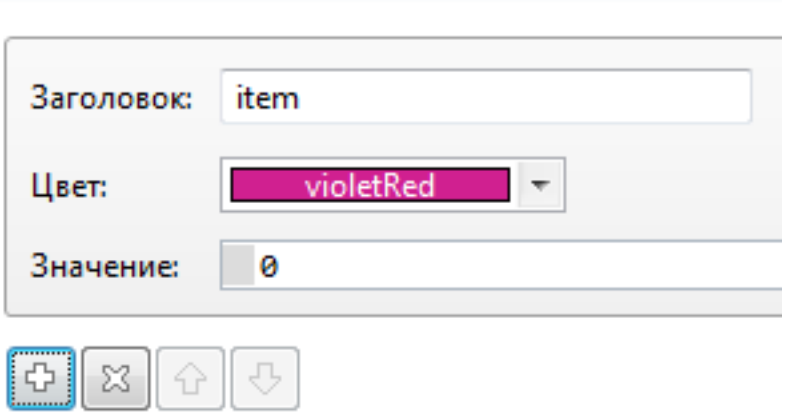


Рис. 3.1. Данные для добавления диаграммы

Поле «Заголовок» будет отображать название, закрепленное за диаграммой. В поле «Значение» необходимо ввести функцию, по которой будет собираться статистика. Например, для компонента Delay есть встроенный набор данных **statsUtilization**, предназначенный для сбора статистики использования этого объекта. Функция mean() возвращает среднее из всех измеренных этим набором данных значений (можно использовать и другие методы сбора статистики, такие, как min() или max() и др.). Например для того, чтобы получить данные о среднем числе занятых каналов обслуживания (средней загрузке устройства), необходимо в поле «Значение» вызвать метод Delay.statsUtilization.mean() (где Delay – наименование компонента, для которого производится сбор статистики).

При необходимости можно аналогичным образом добавить в диаграмму другие столбцы.

В секции «Внешний вид» можно выбрать вид диаграммы; в секции «Легенда» можно изменить расположение легенды относительно диаграммы.

Сбор статистики по времени обслуживания

В некоторых случаях возникает необходимость детализации статистики о распределении каких-либо временных характеристик и отображении собранной статистики в виде гистограмм.

Сбор временных данных можно выполнить как с помощью объектов anylogic, так и программным способом. В первом случае необходимо

включить в модель два объекта: «Time Mesure Start» в точке начала временного отсчета и «Time Mesure End» в точке окончания сбора временной статистики. Для программного сбора статистики необходимо, чтобы поступающие в систему заявки содержали два параметра: «начало отсчета» и «конец отсчета». Смысл этих параметров идентичен объектам «Time Mesure Start» и «Time Mesure End». Разница между этими параметрами даст необходимое время. В свойстве «Объекты TimeMesureStart» для объекта TimeMesureEnd необходимо указать объект TimeMesureStart.

Статистика собирается в два объекта. Один из них предназначен для построения гистограмм (distribution). Второй объект необходим для построения временных графиков (dataset).

Пример.

Пусть для обычной системы обслуживания с очередью необходимо определить гистограмму случайной величины, описывающей время ее пребывания в системе и временной график пребывания. Модель системы будет иметь следующий вид (*рис. 3.2*).

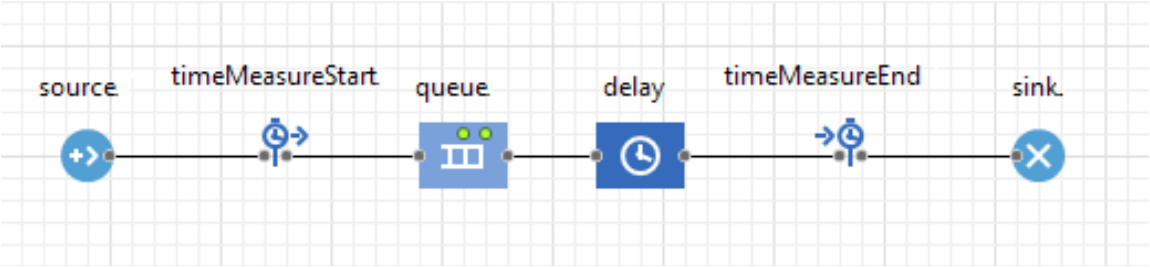


Рис. 3.2. Модель исследуемой системы

Для визуализации собранной статистики в anylogic есть элемент «Гистограмма». Основное ее свойство «Данные» содержит объект, который использовался для сбора данных. Это может быть, например, объект «Time Mesure End». Поместим два объекта из палитры «Статистика»: гистограмму и временной график. В свойстве Данные гистограммы укажем: timeMesureEnd.distribution (*рис.3.3*).

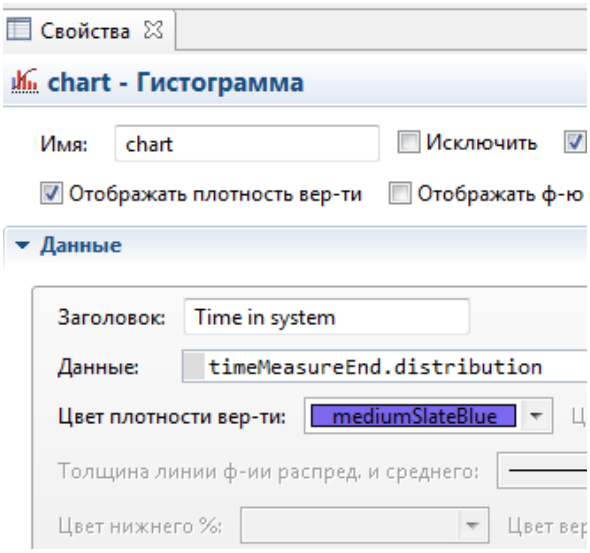


Рис. 3.3. Гистограмма

В аналогичном свойстве временного графика укажем timeMesureEnd.dataset (*рис.3.4*).

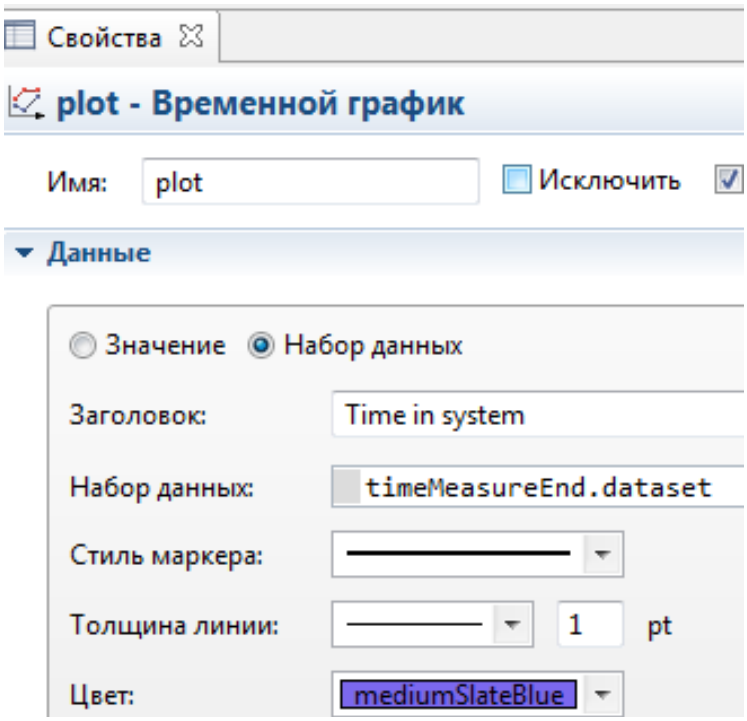


Рис. 3.4. Временной график

Получим следующие результаты (*рис. 3.5*).

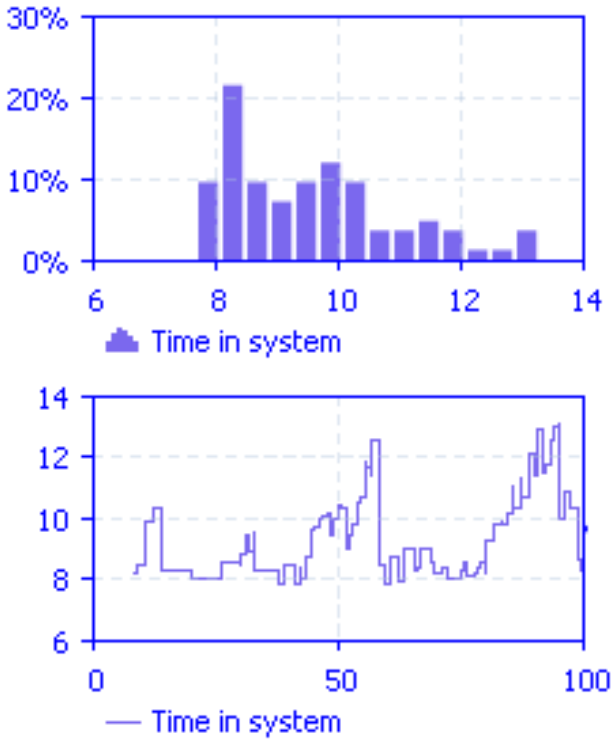


Рис. 3.5. Графики

Другой вариант набора данных для гистограммы – это объект «Данные гистограммы». Рассмотрим другой пример сбора данных об ожидании перед одноканальным устройством. Схема модели представлена на рис. 3.6.

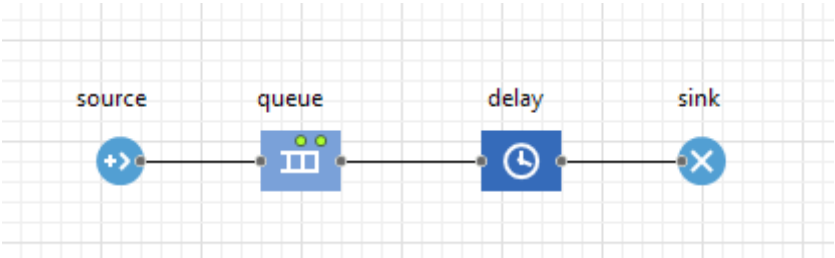


Рис.3.6. Схема модели обслуживания с очередью

Добавим элемент «Данные гистограммы», который назовем TimeWaitDistr.

Чтобы собрать статистику, создадим новый тип агента «Clients», у которого будет свойство StartWaiting типа «время». При входе в очередь будем фиксировать это время, а при выходе получим время ожидания как разницу между текущим на тот момент временем и временем StartWaiting. Это временную разницу будем добавлять в элемент «TimeWaitDistr».

Поскольку параметр StartWaiting является параметром агента Clients, то обратиться к параметру необходимо с указанием агента (*рис. 3.7*).

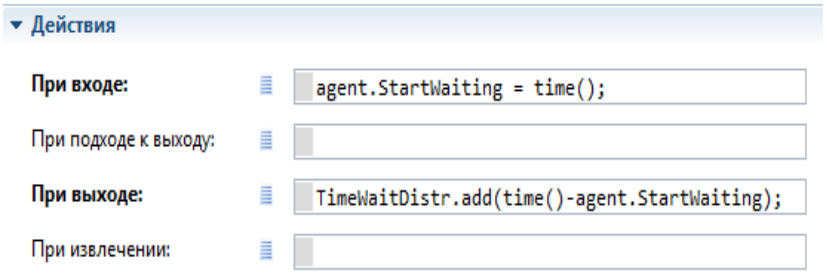


Рис.3.7. Действия при входе и выходе из очереди

При этом у объекта source необходимо в поле «Новый агент» выбрать «Clients», поскольку необходимо, чтобы в модель поступали заявки, имеющие свойство StartWaiting, что есть именно у агентов созданного типа Clients.

Простейшая анимация

AnyLogic позволяет отобразить происходящие в системе процессы с необходимой детализацией. В простейшем случае можно показать место возникновения и обслуживания заявок. Для этого из разметки пространства необходимо поместить в рабочую область соответствующие элементы (например, узлы) и в каждом блоке (Source, Delay, Queue) описать место возникновения/обслуживания/ ожидания (*рис. 3.8*).

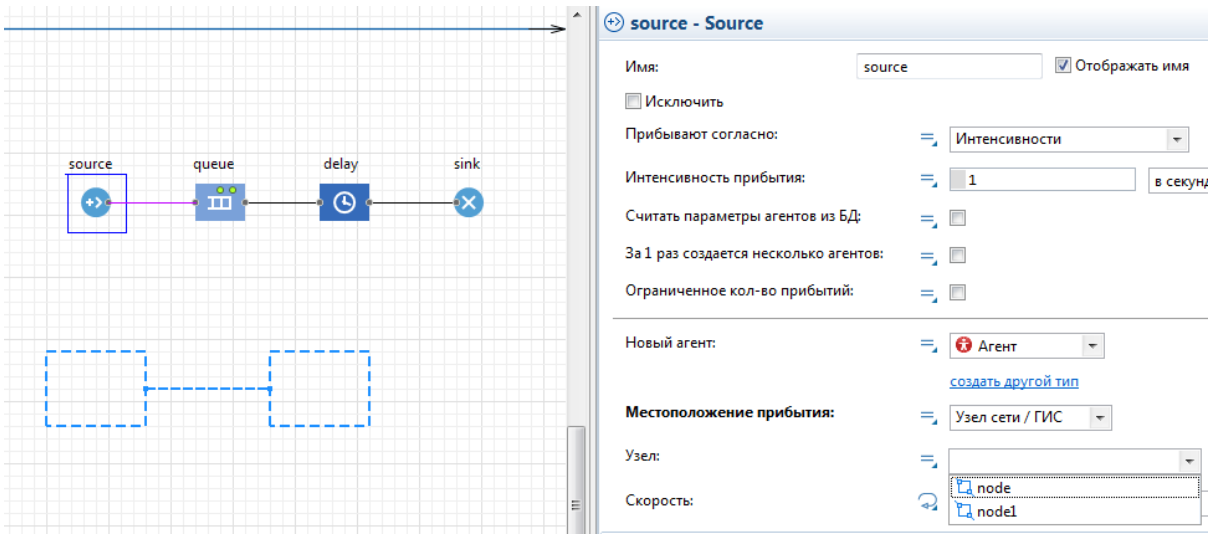


Рис. 3.8. Реализация простейшей анимации

Создадим картинку для объекта. Для этого поместим объект «агент» из палитры Агент в рабочую область и выберем «Просто создать тип агента». Назовем каким-либо образом данный тип (*рис. 3.9*).

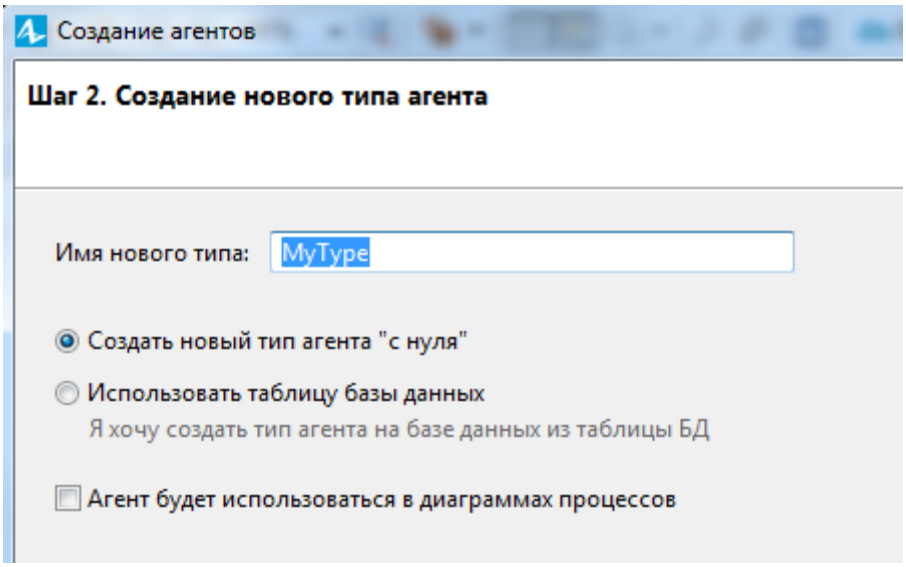


Рис.3.9. Настройка типа агента

Выберем картинку для заявки. Пусть в данном случае это будут коробки (*рис. 3.10*).

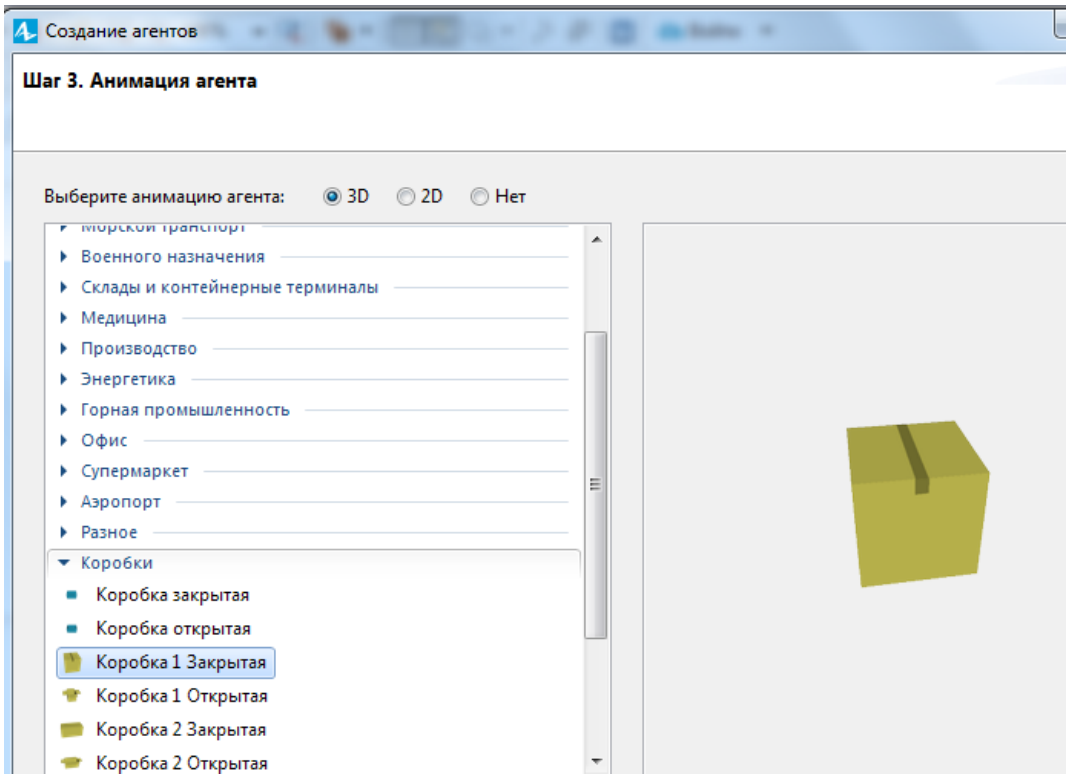


Рис. 3.10. Выбор изображения для данного типа заявок

В свойствах Source выберем свойство «Новый агент» и из распахивающегося списка выберем созданный тип (*рис. 3.11*).

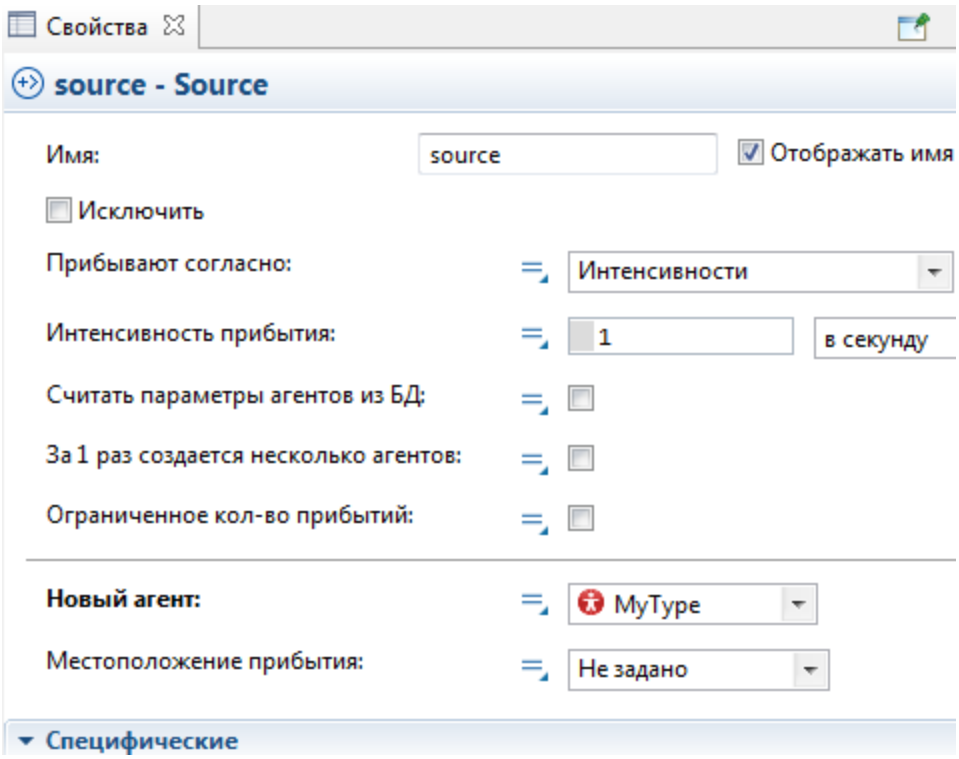


Рис. 3.11. Настройка свойств Source

В результате получим следующую анимацию (*рис. 3.12*).

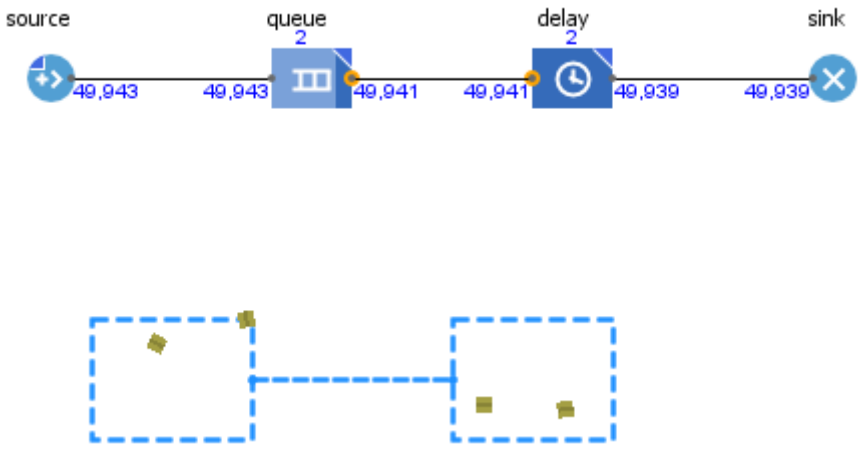


Рис. 3.12. Результат работы программы

Задание работы модели «на время»

Для того, чтобы задать выполнение модели в течение заданного времени, необходимо:

1) Запустить модель

2) Выбрать «Запустить на момент времени», распахнув список «Запустить»:

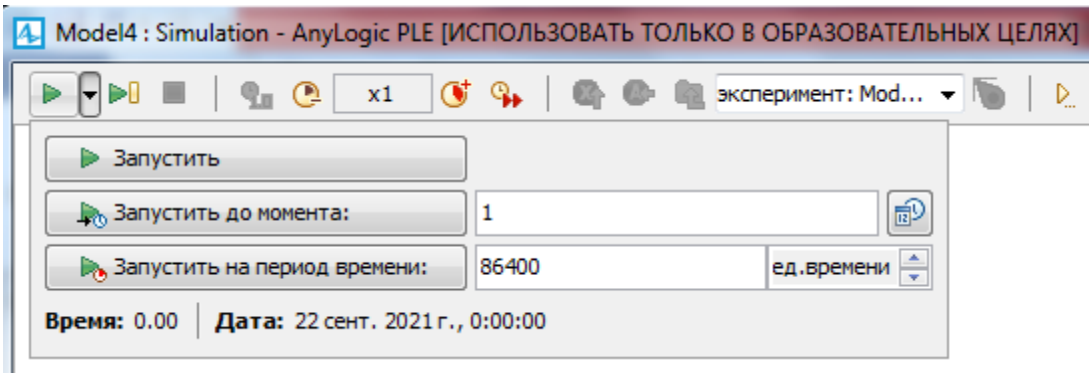


Рис. 3.13. Запуск модели на время

3) Выбрать необходимое время. Например, если модельное время изменяется в секундах, а требуется создать работу модели в течение суток, запуск будет выглядеть, как показано на *рис. 3.13*.

3.2.Задание к работе

Создать имитационную модель согласно варианту задания, отображая перемещение заявок по всем фрагментам (появление, ожидание, обслуживание и т.д.). Рассчитать заданные вероятностно-временные характеристики и отобразить необходимые графики и гистограммы.

Отчет должен содержать:

- постановку задачи;
- реализованную модель с описанием особенностей ее построения;
- результаты моделирования, включая все требуемые характеристики и графики.

3.3. Варианты заданий

1. В цех на участок обработки поступают партии деталей по три детали в каждой. Интервалы между приходами партий – случайные величины, равномерно распределенные в интервале 50±10 мин. Первичная обработка деталей происходит на одном из станков двух типов. Деталь поступает на обработку на станок с меньшей очередью. Станок первого типа обрабатывает деталь за 40±10 мин и допускает 10% брака; второго типа – соответственно за 60±15 мин и 5% брака. Все бракованные детали возвращаются на повторную обработку на свой станок.

После первичной обработки детали поступают на вторичную обработку, которую проводят два параллельно работающих станка за

время 25 мин (время распределено экспоненциально). В случае занятости станков детали ожидают в очереди. В обычном режиме вторичная обработка осуществляется первым станком, а второй подключается в случае наличия в очереди более трех деталей.

Смоделировать работу производственного участка в течение суток. Определить распределение времени нахождения деталей на производственном участке, а также загрузку всех станков. Результаты представить в виде гистограмм.

- Создать простейшую анимацию, отобразив поступление деталей, процесс их обслуживания и повторную обработку.

2. В центр тестирования поступают заявки в среднем через 80 с (время распределено экспоненциально). Одна треть из них поступает на в подсистему У1, состоящую из двух устройств, каждой из которых выполняет тестирование в целом за 200 с (время экспоненциальное), а две трети – в подсистему У2, состоящую из одного устройства, осуществляющего тестирование в среднем за 70 с (время экспоненциальное). Первая подсистема отбраковывает ориентировочно 8% заявок, а вторая – 5 %. Все бракованные заявки проходят доработку в специальном устройстве У3 в среднем за 150 с (время экспоненциально), после чего направляются в подсистему У1 на повторную проверку.

Смоделировать процесс тестирования. Определить загрузку У1, У2 и У3. Определить распределение времени пребывания заявки в системе и время ожидания в очереди перед устройством У3. Результаты представить в виде гистограмм.

- Создать простейшую анимацию, отобразив поступление заявок и процесс тестирования.

ЛАБОРАТОРНАЯ РАБОТА 4
РАЗРАБОТКА МОДЕЛЕЙ С ИСПОЛЬЗОВАНИЕМ ДИАГРАММ СОСТОЯНИЙ

Цель работы: получение практических навыков разработки моделей с использованием диаграмм состояний

4.1. Краткие теоретические сведения

Для того чтобы построить «стейтчарт», следует использовать элементы из палитры Диаграмма состояний (*рис. 4.1*).

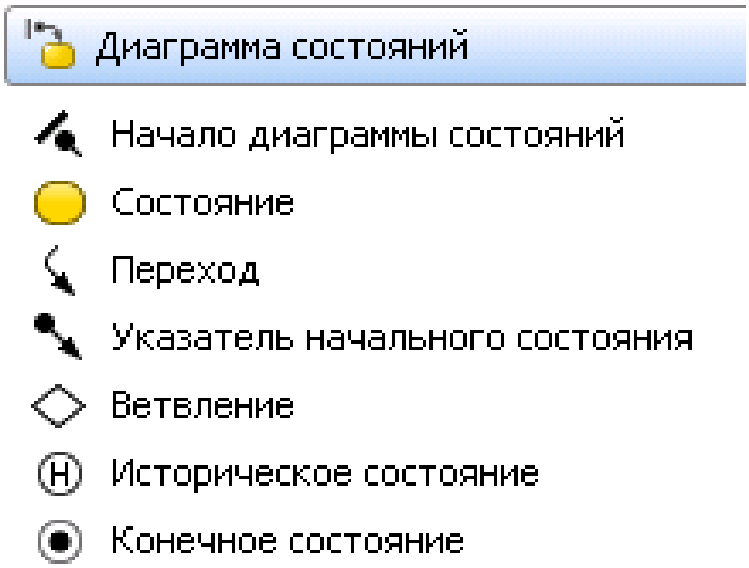


Рис. 4.1. Элементы палитры Диаграммы состояний

1. определяет начальное состояние всего «стейтчарта»;
2. с помощью кнопки рисуются состояния (как простые, так и гиперсостояния);
3. используется для рисования переходов между состояниями;
4. определяет начальное состояние внутри сложного состояния;
5. используется для рисования состояния, являющегося "финальным" в поведении активного объекта.

Диаграмма может содержать так называемые сложные (составные) состояния. Это означает, что такое состояние будет включать в себя множество простых состояний, связанных некоторыми переходами.

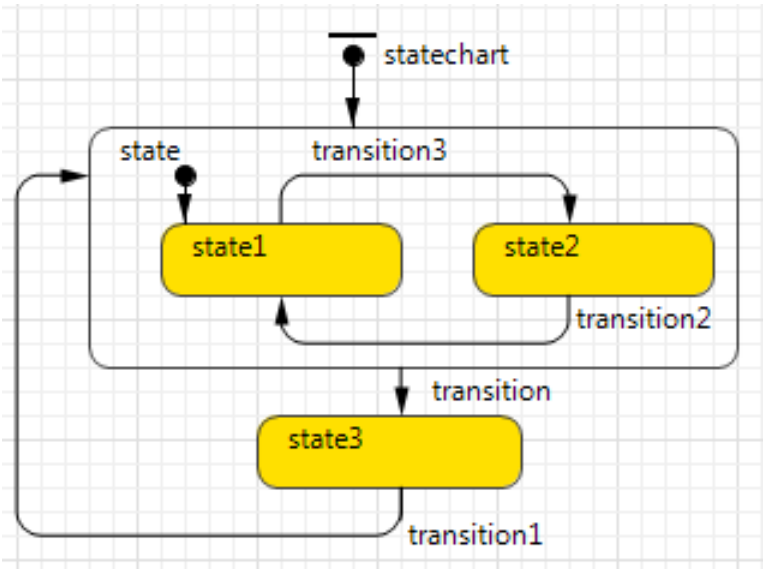


Рис.4.2. Пример диаграммы состояний

Чтобы нарисовать переходы между такими состояниями, удобно использовать режим рисования. Для этого необходимо сделать двойной щелчок мышью по элементу Переход в палитре (при этом его значок должен поменяться на этот:). Теперь можно рисовать переход, последовательными щелчками мыши добавляя в нужных местах диаграммы начальную точку перехода, затем точки его изгиба и, наконец, двойным щелчком - конечную точку перехода.

Внутри сложного состояния обязательно должно быть начальное состояние, отмеченное стрелкой .

4.2. Задание к работе

Реализовать в среде AnyLogic модель светофора, для которого определены следующие состояния: зеленый, мигающий зеленый, желтый, красный, красный и желтый.

ХОД ВЫПОЛНЕНИЯ РАБОТЫ

Сначала необходимо разработать диаграмму, соответствующую указанным в задании состояниям. Каждое из таких состояний будет соответствовать определенному цвету (или комбинации цветов) для модели светофора. Можно определить следующее соответствие между необходимыми в задании цветами и состояниями (таблица 1).

Таблица 4.1

Соответствие состояний цветам светофора

Состояние	Цвет
Движение	Зеленый
Внимание	Мигающий зеленый
Замедление	Желтый
Остановка	Красный
приготовиться	Желтый+красный

Все состояния кроме мигающего зеленого являются простыми. Мигающий зеленый – это сложное состояние, включающее два подсостояния:

- зеленый свет горит;
- зеленый свет не горит.

Исходя из этого, спроектируем диаграмму состояний. Особенностью данной диаграммы является то, что она не будет иметь конечного

состояния (поскольку из состояния «приготовиться» система опять должна попасть в состояние «движение»).

Следующим этапом является прорисовка светофора. Для этого из палитры «Презентация» переместим на форму элемент «прямоугольник», в который поместим три элемента «овал» таким образом, чтобы это визуально напоминало светофор. Зададим серый цвет заливки прямоугольника. Цвета трех овалов должны динамически меняться (в этом и заключается цель данной модели).

Далее необходимо связать диаграмму и картинку светофора и настроить его цвета. Для этого воспользуемся тремя параметрами, каждый из которых будет соответствовать определенному цвету и иметь тип boolean (включен или выключен). В зависимости от значения каждого из параметров соответствующий овал должен быть включен (определенного цвета) или серый. Например, для красного цвета в заливке соответствующего овала будет условие:

cl_red? red: gray

Здесь cl_red – параметр, соответствующий красному цвету.

Последним этапом осталось определить значения параметров в зависимости от наступления состояний. Рассмотрим состояние «движение». У каждого состояния можно определить действия при входе и при выходе. Очевидно, что движение начинается при зеленом цвете светофора и заканчивается (переходит в состояние «внимание»), когда зеленый начинает мигать. Поэтому действия для данного состояния будут следующими (*рис. 4.3*).

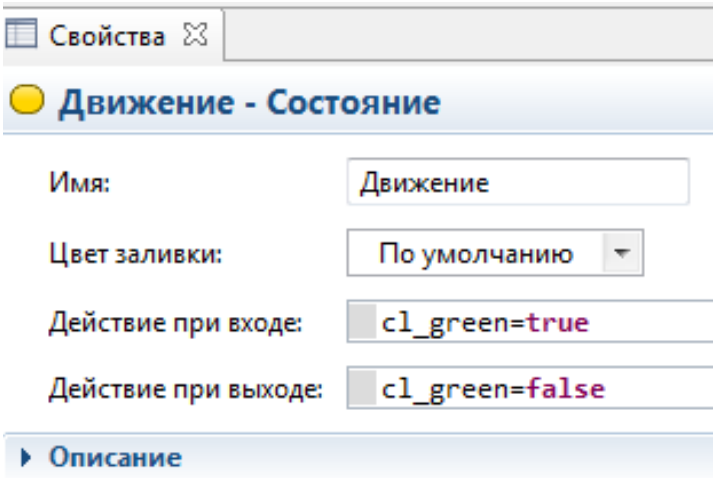


Рис. 4.3. Свойства состояния «Движение»

Здесь cl_green – параметр, соответствующий зеленому цвету.

Настроив аналогичным образом остальные параметры, получим работающую модель.

По умолчанию время перехода их одного состояния в другое равен 1с. Для того, чтобы изменить это время и сохранить пропорцию между

нахождением системы в каждом из состояний (например, желтый и красный с желтым горят значительно меньше, чем красный и зеленый) необходимо настроить таймауты для каждого из переходов (например переход из состояния «приготовиться» в состояние «движение» длится 4с, переход из состояние «движение» в состояние «внимание» - 10 с и т.д.).

Смоделированный светофор должен иметь вид, представленный на *рис. 4.4* - .

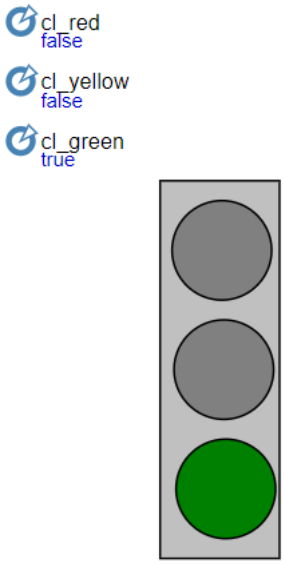


Рис. 4.4. Модель светофора с горящим зеленым светом

На следующем рисунке приведен пример желтого мигающего света.

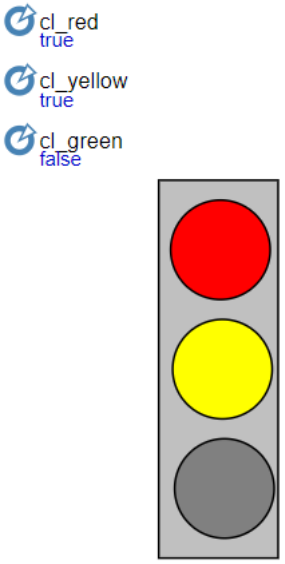


Рис. 4.5. Модель светофора с представленным состоянием «приготовиться»

Отчет по лабораторной работе должен содержать:

- постановку задачи;
- описание процесса построения имитационной модели;
- скриншоты с результатами;
- вывод о проделанной лабораторной работе.

ЛАБОРАТОРНАЯ РАБОТА 5

РАЗРАБОТКА МОДЕЛЕЙ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕК ПЕШЕХОДНОГО И ДОРОЖНОГО ДВИЖЕНИЯ

Цель работы: изучение возможности визуализации моделей и сбора статистики с использованием пешеходной и дорожной библиотек.

5.1. Краткие теоретические сведения

Пешеходная библиотека

В условиях растущего населения крупных городов и увеличения темпов строительства зданий все большую актуальность приобретает моделирование пешеходных потоков. В связи с этим, в среде Anylogic предусмотрены блоки, которые обеспечат возможность моделирования и наглядной визуализации движения пешеходов.

Целесообразность построения переходных моделей продиктована целым рядом причин. Во-первых, такая модель весьма эффективна на стадии проектирования нового проекта. Ее наличие поможет осуществить поиск наилучших решений, сравнение различных вариантов и быстрой оценки вносимых изменений. Во-вторых, она целесообразна на стадии предварительной оценки проекта. В этом случае имитационное моделирование поможет оценить планируемую нагрузку проекта и, при необходимости, внести корректировки на ранних стадиях. Пешеходные модели могут использоваться и на работающих объектах для достижения следующих целей:

- поиск наилучших точек для размещения рекламы, торговых палаток и т.д.;
- оптимизация работы торговых точек (количество персонала, часы работы и т.д.);
- оценка пропускной способности исследуемого объекта при увеличении нагрузки;
- решение различных вопросов по безопасности (планы эвакуации и т.д.);

- поиск наиболее оптимальных временных маршрутов при ремонтных работах;

и т.д.

Исходя из данных целей, при моделировании движений пешеходов решаются следующие задачи:

- оценка пропускной способности зданий и объектов внутри них;
- оптимизация бизнес-процессов в пунктах обслуживания;
- оценка плотности потока посетителей торговых зон;
- нахождение «узких мест» пешеходных потоков;
- создание и обоснование планов эвакуации при ЧС;
- оценка доступности парковок, дорожной сети и общественного транспорта.

Пешеходная библиотека имеет вид, представленный на *рис. 5.1*.

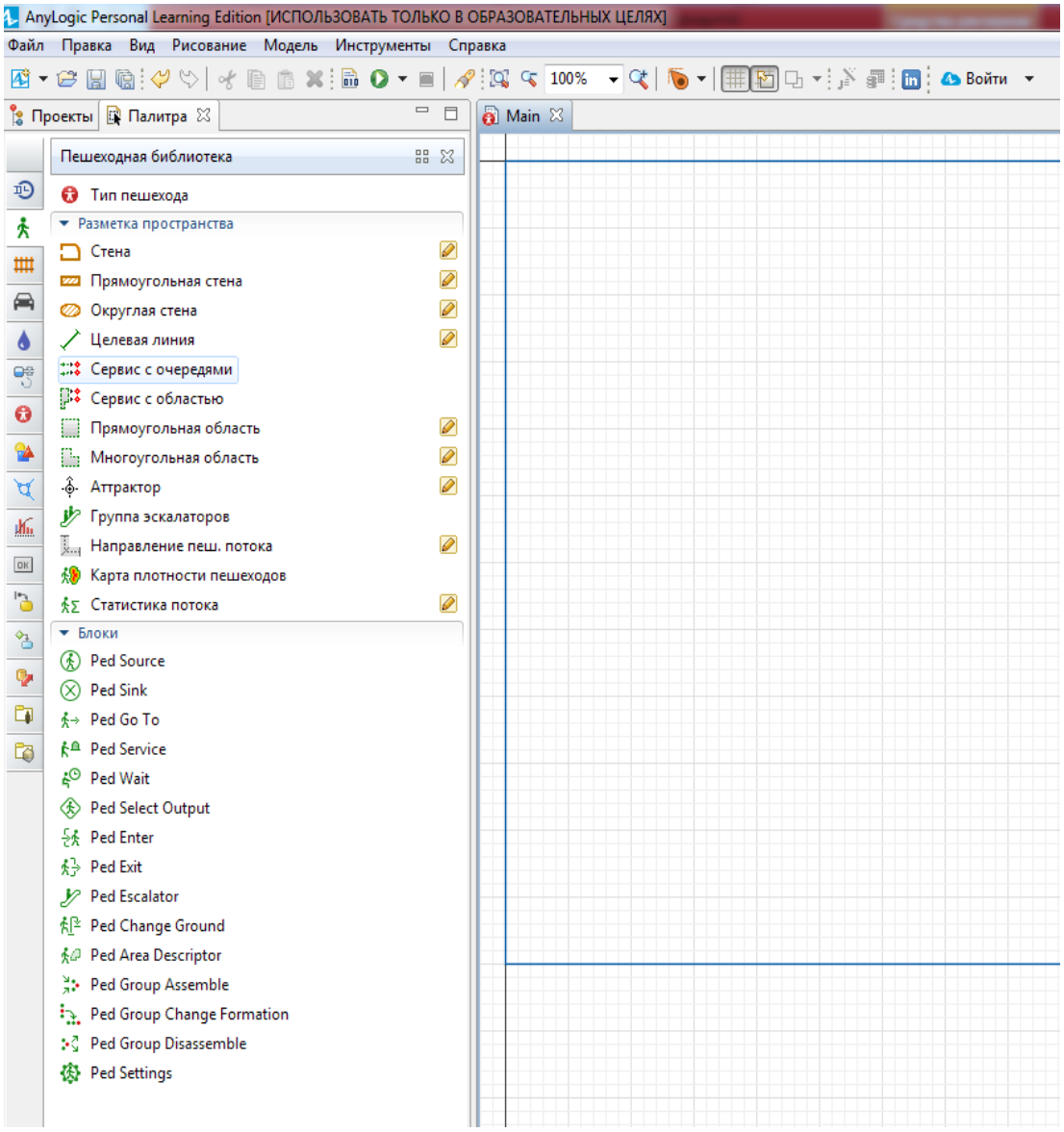


Рис. 5.1. Пешеходная библиотека

Моделирование процесса пешеходного движения осуществляется в три этапа. На первом этапе на рабочую область наносится чертеж моделируемого объекта. С точки зрения Anylogic он будет выступать в качестве рисунка.

На втором этапе осуществляется так называемая разметка пространства. Она заключается в нанесении поверх данного чертежа пешеходных маршрутов, направлений пешеходных потоков, стен, торговых точек и т.д. Этот этап осуществляется с помощью библиотека «Разметка пространства», однако все основные блоки разметки пространства, необходимые для решения данной задачи, вынесены также в пешеходную библиотеку. На последнем этапе производится непосредственно моделирование. На основании статистических данных определяются источники потоков и их интенсивности, направления потоков и т.д.

Рассмотрим основные блоки пешеходной библиотеки, доступные для решения описанной задачи моделирования.

PedSource. Данный блок предназначен для создания в модели пешеходов. Поток пешеходов задается:

- местом своего появления;
- интенсивностью или временем появления;
- скоростью.

Свойства пешеходов представлены на *рис. 5.2.*

Если возникает необходимость в задании более качественной анимации, можно в свойствах «Новый пешеход» выбрать «Создать другой тип». При этом откроется диалоговое окно, в котором можно выбрать иллюстрацию для пешеходов (например, рабочие).

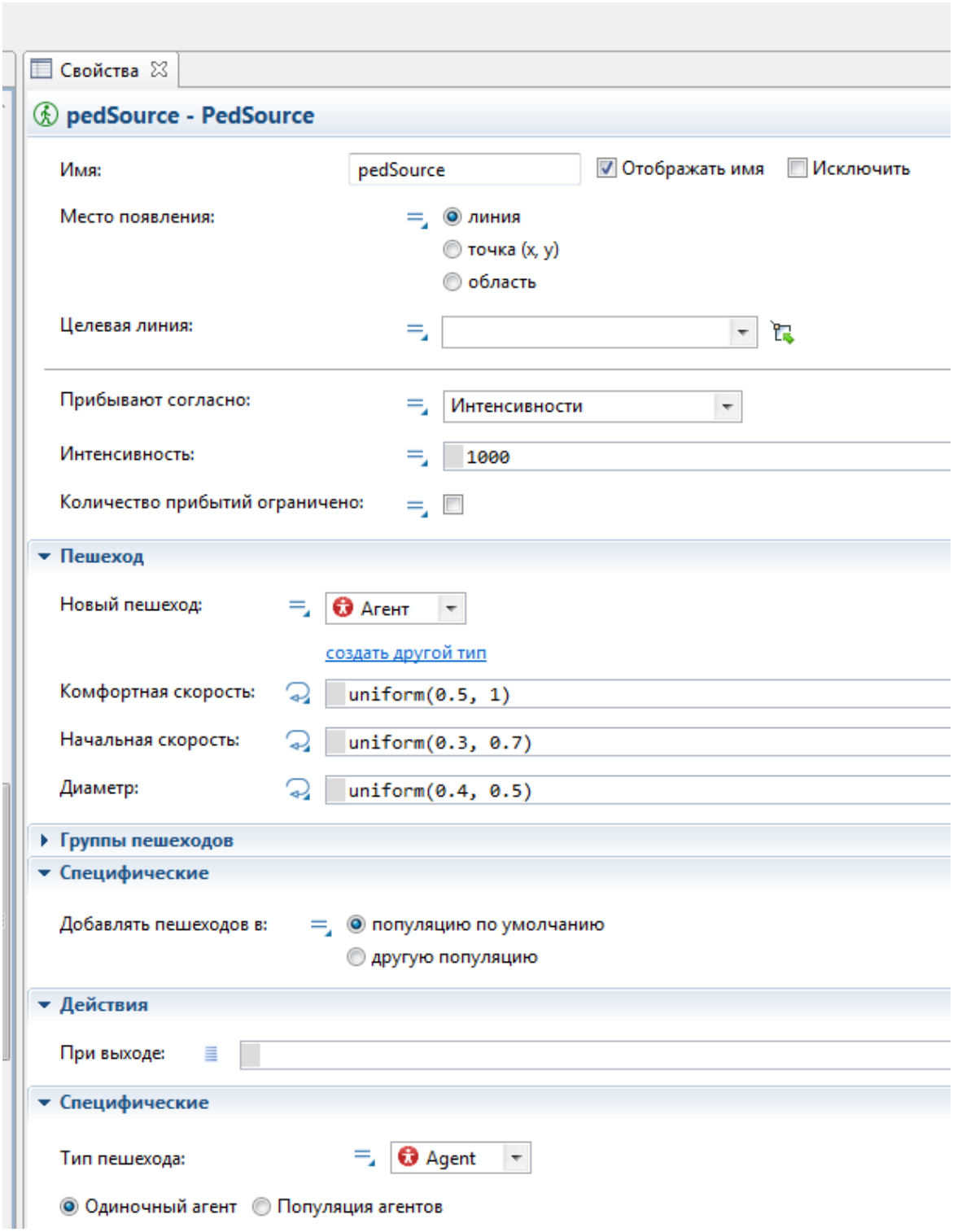


Рис. 5.2. Свойства объекта – генератора пешеходов

PedGoTo. Заставляет пешеходов перейти в заданное место моделируемого пространства, которое может быть задано линией или точкой. Свойства блока представлены на *рис. 5.3*.

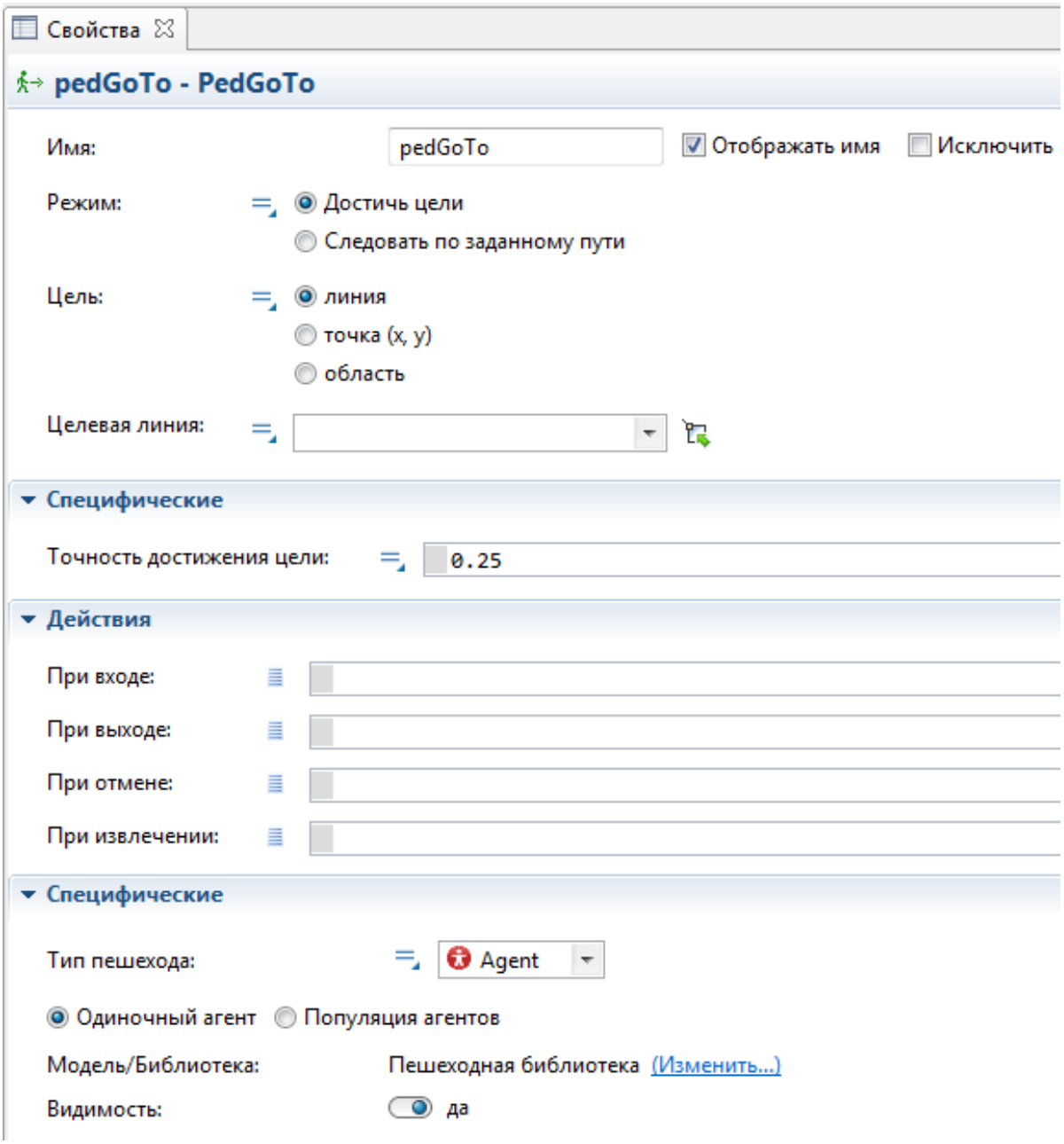


Рис. 5.3. Свойства PedGoTo

Как следует из данного рисунка, пешеход может осуществлять движение одним из двух способов:

- идти к заданной цели;
- двигаться по заданному маршруту.

Первый случай представлен на рис. 4. Цель может быть задана линией, точкой или областью. Маршрут задается блоком «Направление пеш. Потока». При помещении на рабочую область этот блок фактически представляет собой дорогу, которую можно нарисовать с необходимой для пользователя точностью (в виде ломаной, с поворотами и т.д.).

PedSink. Удаляет поступивших в объект пешеходов из моделируемой среды. Обычно объект используется в качестве конечной

точки блок-схемы, формализующей поток пешеходов. PedSink автоматически ведет подсчет пешеходов.

PedWait. Заставляет пешеходов перейти в заданное место и ожидать там в течение определенного периода времени. Место может быть выбрано случайно внутри заданной области (с аттракторами или без них) или указано вручную. Аттракторы - это места внутри области, которые притягивают пешеходов во время их ожидания (например, реклама, информационные стенды и т.д.).

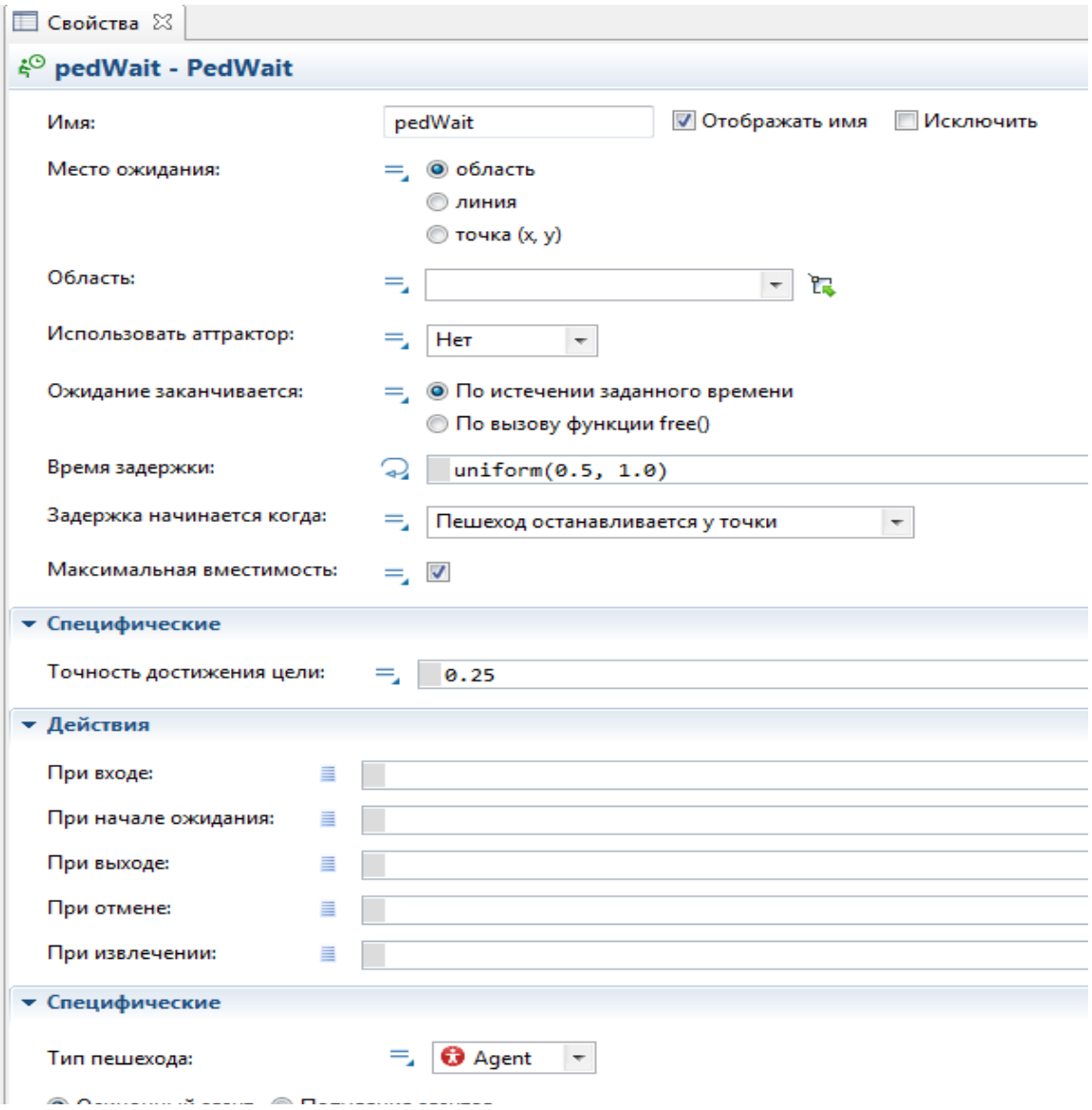


Рис. 5.4. Свойства блока PedWait

Блок PedWait предусматривает два вида ожиданий:
- ожидание в течение заданного времени;
- ожидание до тех пор, пока не произойдет некоторое событие.

Эта особенность регулируется свойством «ожидание заканчивается». В первом случае появляется поле, в которое необходимо ввести время, по истечении которого ожидание будет завершено. Во втором случае, в каком-либо из блоков необходимо вызвать функцию:

pedWait.freeAll();

PedService. Направляет поток пешеходов на обслуживание в некоторый сервис и задерживает там пешехода в течение заданного времени. Данный сервис создается с помощью одного из двух блоков разметки пространства:

- сервис с очередями (используется для того, чтобы задавать сервисы, в которых пешеходы ждут в очереди, пока сервис не будет доступен);
- сервис с областью – используется для того, чтобы задавать сервисы с электронной очередью. В таком случае пешеходы не стоят в очереди, а ждут в расположенной рядом области.

Моделирование дорожного движения

Современный этап развития общества характеризуется интенсивным увеличением количества автомобилей. В связи с этим, важной задачей является повышение эффективности транспортной работы автомобильных дорог в проекте организации движения. Ее решение возможно только при широком внедрении в практику вариантного проектирования автомобильных дорог и имитационного моделирования, которое практически невозможно без широкой автоматизации проектирования на основе ЭВМ.

В настоящее время в ведущих дорожных проектных организациях объем автоматизированного проектирования и моделирования составляет от 85% общего объема работ проектирования.

Высокие требования, которые предъявляет современный автомобильный транспорт к качеству автомобильных дорог, могут быть реализованы лишь при системном подходе, как к самому процессу проектирования, так и к последующим этапам реализации результатов этого проектирования путем моделирования: строительству и эксплуатации [].

Среда Anylogic располагает специальной библиотекой дорожного движения, которая позволяет реализовать следующий функционал:

- моделирование улиц, дорог, перекрестков и транспортных развязок;
- работа общественного транспорта;
- действия автомобилей на парковках;
- анализ плотности траффика в дорожной сети;
- реакция транспорта на светофоры и правила дорожного движения.

Данная библиотека имеет следующий вид (*рис. 5.5*).

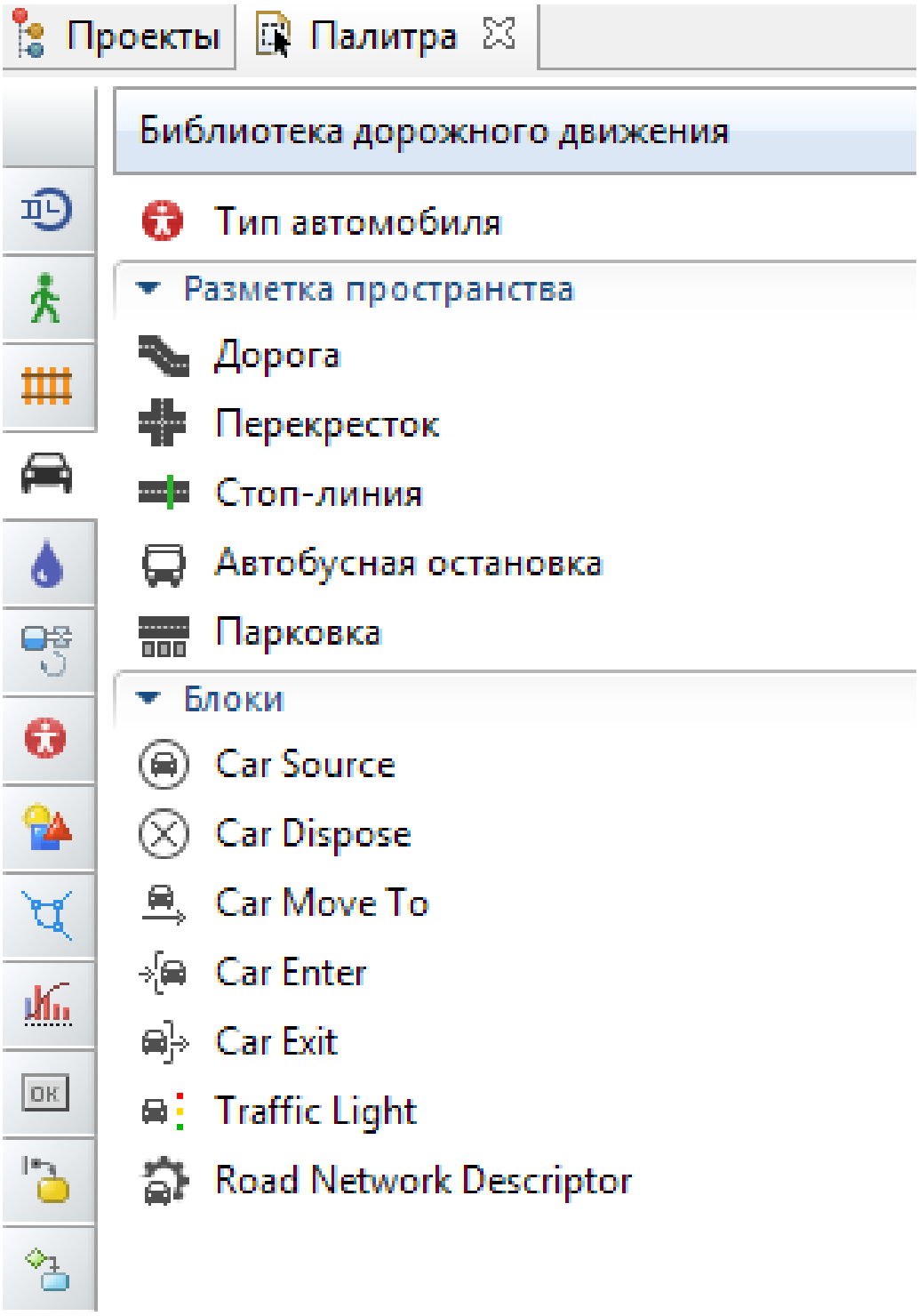


Рис. 5.5. Библиотека дорожного движения

Как и в случае пешеходных потоков, моделирование дорожного движения включает в себя три основные стадии:

- нанесение на рабочую область карты с исследуемыми дорогами (которая будет восприниматься просто как изображение, необходимое пользователю для повышения эффективности визуального восприятия результатов моделирования);

- нанесение поверх данной карты разметки пространства (дороги, стоп-линии, светофоры и т.д.), элементы которой необходимы для работы основных блоков библиотеки;

- описание непосредственно модели с помощью блоков библиотеки дорожного движения.

Рассмотрим основные элементы разметки пространства и библиотеки дорожного движения, необходимые для создания дорожной модели.

Элементы разметки пространства

1. **Дорога** – элементарный компонент разметки пространства, позволяющий создавать дорожную сеть. Дорога может содержать произвольное количество полос. Дорога может быть как односторонняя, так и двусторонняя. Одна дорога содержит неизменное количество полос на всем своем протяжении. Можно создавать участки слияния дорог, путем соединения двух дорог с разным количеством полос.

2. **Перекресток** – необходим для слияния двух и больше дорог. Перекресток автоматически получается при соединении дорог. При формировании перекрестка автоматически появляются белые точечные линии – соединители полос, которые задают разрешенные направления движения транспорта на перекрестке.

3. **Стоп-линия** – ия является графическим элементом разметки пространства, задающим точку на дороге, у которой транспорт должен останавливаться. Стоп-линия может быть использована блоком Traffic Light (Светофор) для регулирования движения на сложно контролируемых участках дороги.

Блоки дорожного движения

- CarSource – Создает автомобили и пытается поместить их в указанное место дорожной сети. Автомобиль можно поместить на указанную дорогу или парковку. Основные свойства:
  - Интенсивность появления;
  - Место появления;
  - Скорость

И т.д.

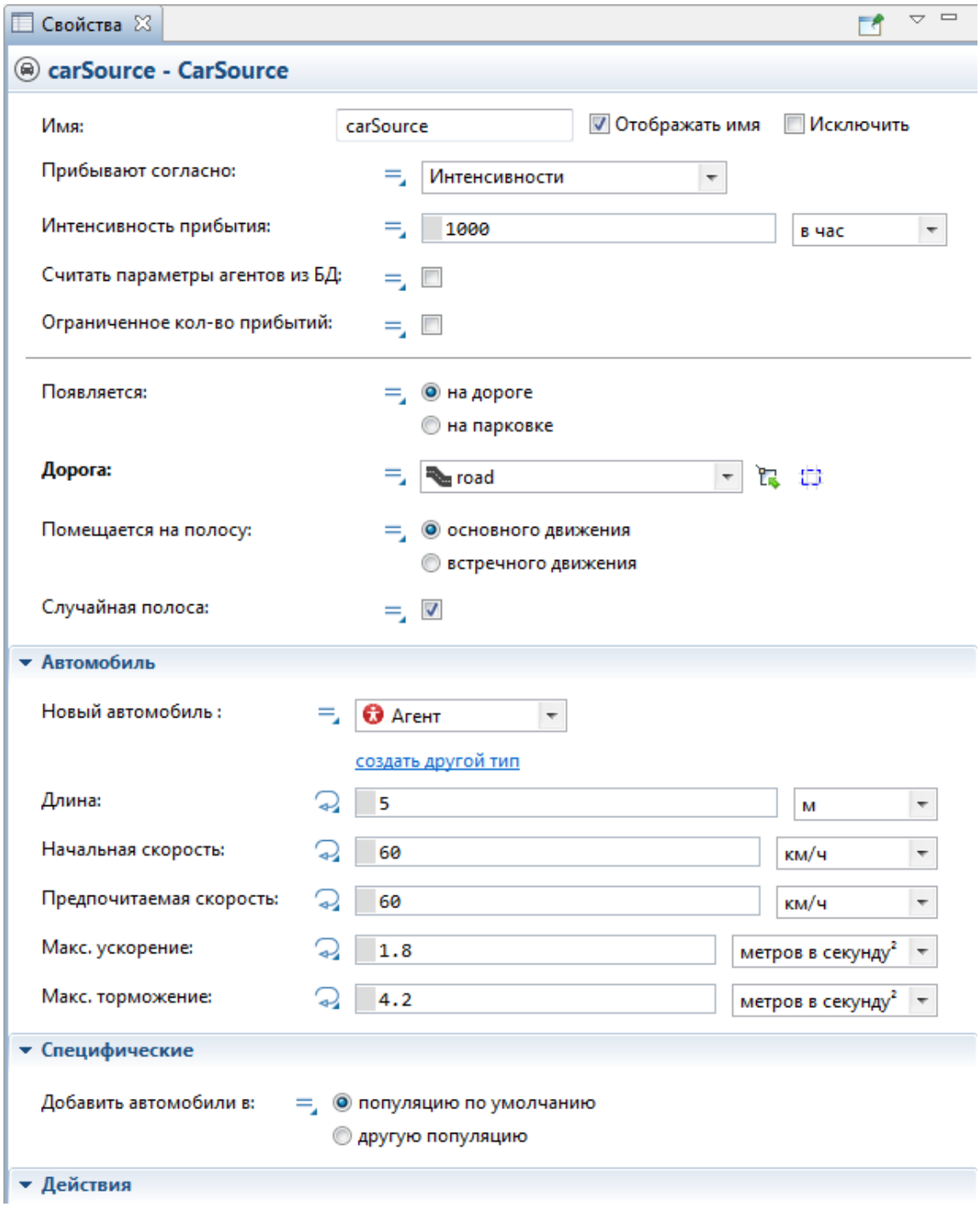


Рис. 5.6. Свойства компонента CarSource

- CarMoveTo – Управление движением машин. 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. Одним из важнейших свойств элемента является цель движения. 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.

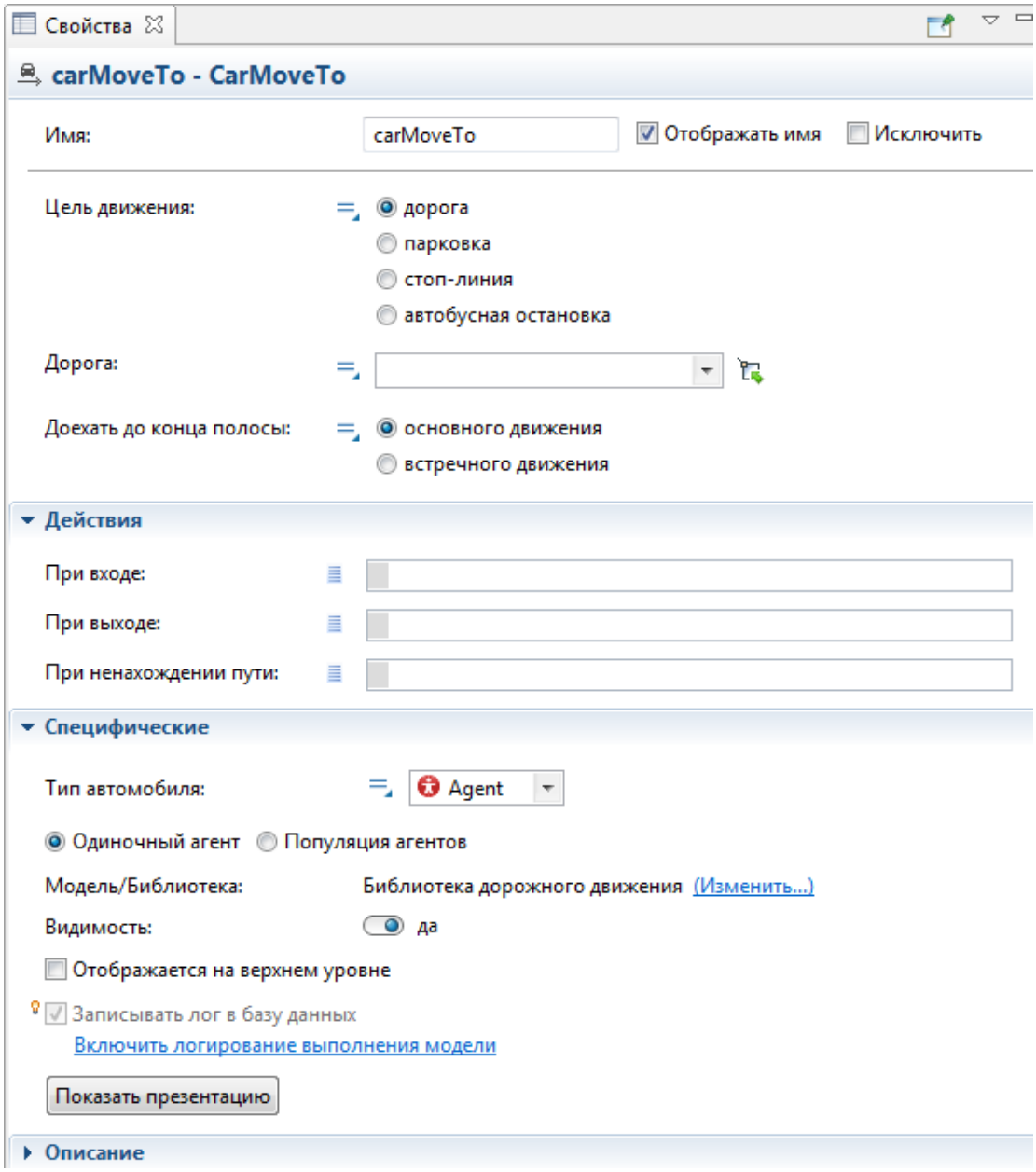


Рис. 5.7. Свойства блока CarMoveTo

Удаление машины из модели осуществляется элементом Car Dispose.

TrafficLight – моделирует светофор (оптическое устройство, предназначенное для регулирования движения на автомобильных, пешеходных перекрестках, а также других сложно контролируемых участках дорожного движения). В свойствах необходимо указать, где

данный светофор будет задавать режим работы. Существуют следующие варианты:

- режим работы для стоп-линий перекрестка (подразумевается, что в модели есть перекресток и далее происходит настройка его стоп-линий);
- соединителей полос перекрестка;
- заданных стоп-линий.

Выбрав необходимый вариант, далее производится настройка фаз светофора (выбираются цвета фаз для каждой стоп-линии и указывается их длительность).

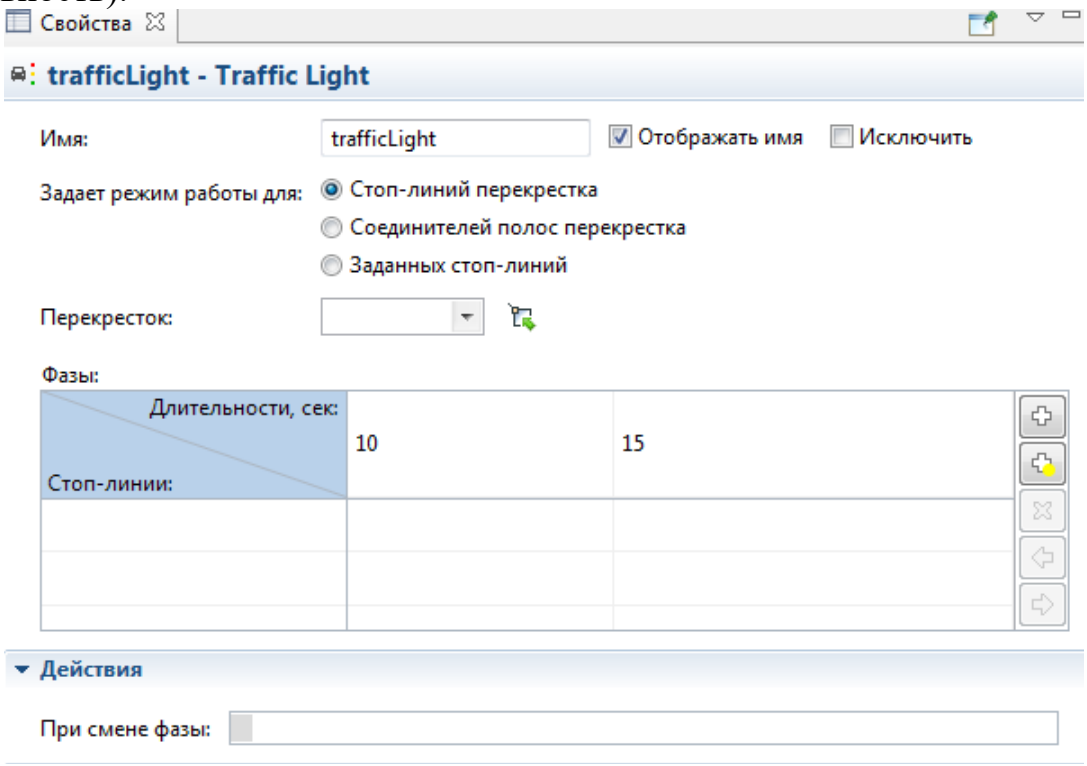


Рис. 5.8. Свойства TrafficLight

Предположим, что, к примеру, представленному на рис. 7, необходимо добавить дорогу и светофор, который работал бы синхронно со светофором, описанным с помощью диаграммы состояний. Поместим на рабочую поверхность дорогу и две стоп-линии, как это показано на рис. 9.

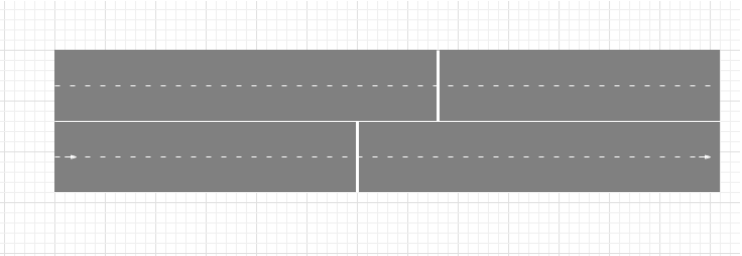


Рис. 5.9. Дорога со стоп-линиями

Стоп-линии должны располагаться на достаточном расстоянии от начала (и, соответственно, от конца) дороги, иначе машины не успеют затормозить, и светофор, заданный объектом TrafficLight, работать не будет.

Поместим данный объект на рабочую область. В свойстве «Задает режим работы для» выберем «заданных стоп-линий» (*рис. 5.10*).

После этого имеется возможность добавлять стоп-линии (метка (1) на *рис. 5.10*) и фазы светофора (метка(2) на *рис. 5.10*).

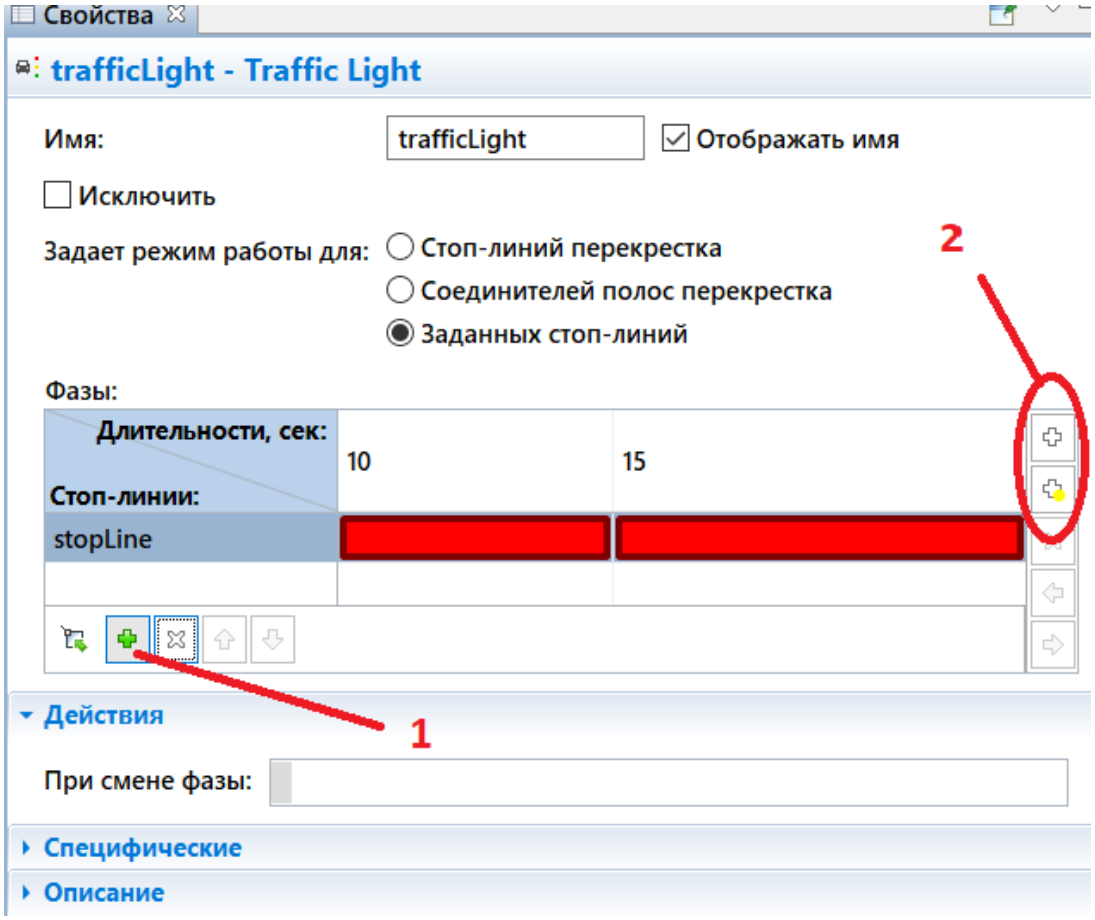


Рис. 5.10. Настройка свойств TrafficLight

Чтобы синхронизировать работу светофора со светофором, смоделированным ранее, перейдем в диаграмму состояний, и в действиях при входе в очередное состояние вызовем метод SwitchToNextPhase() для объекта trafficLight (*рис. 5.11*). При этом длительности фаз, которые задаются в свойствах TrafficLight, должны быть установлены в бесконечность (infinity).

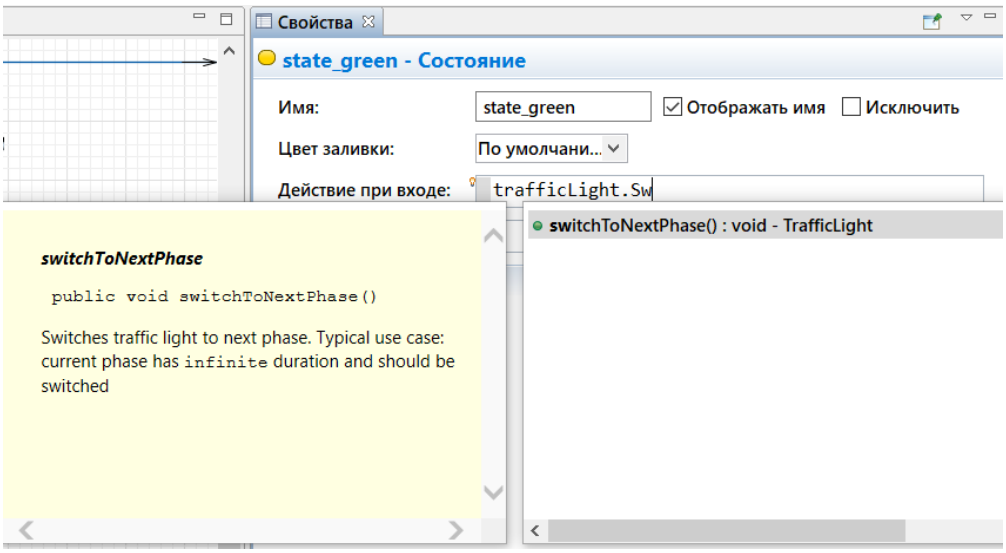


Рис. 5.11. Синхронизация объекта TrafficLight со светофором, смоделированным ранее

Для того, чтобы смоделировать перекресток, необходимо соединить два (или более) объекта типа «дорога». В этом случае автоматически будут указаны все маршруты, по которым возможно передвижение (рис. 5.12).

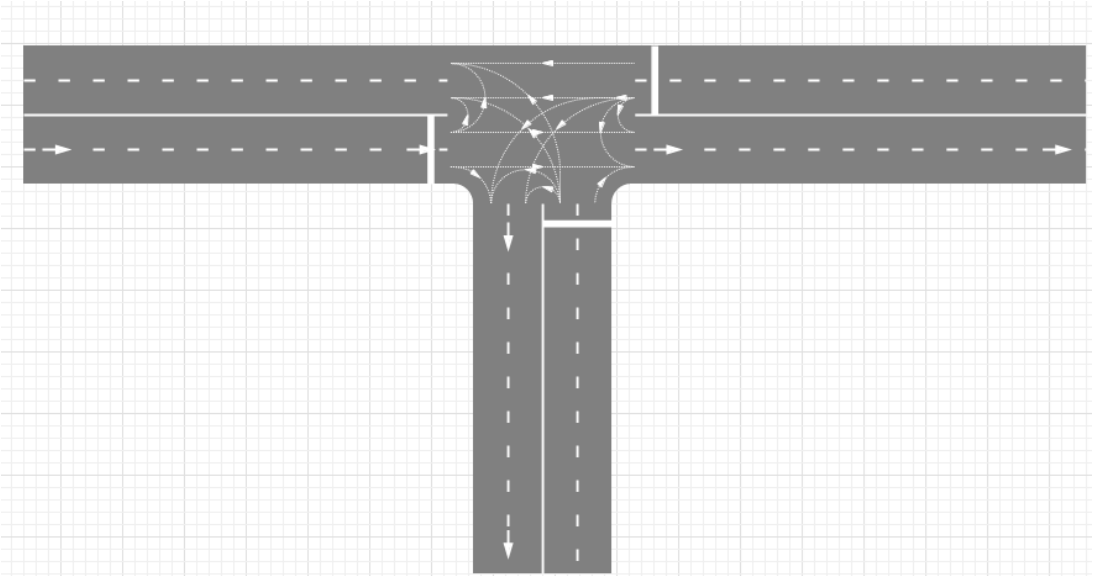


Рис. 5.12. Перекресток

В данном случае может возникнуть ошибка, если машина попытается выполнить поворот в том случае, если отсутствует соответствующая стрелка. В частности, у блока PedSource есть свойство «случайная полоса», которое отображает, на какой полосе возникает поток машин. Если задано свойство «случайная полоса», то перед попыткой поворота необходимо

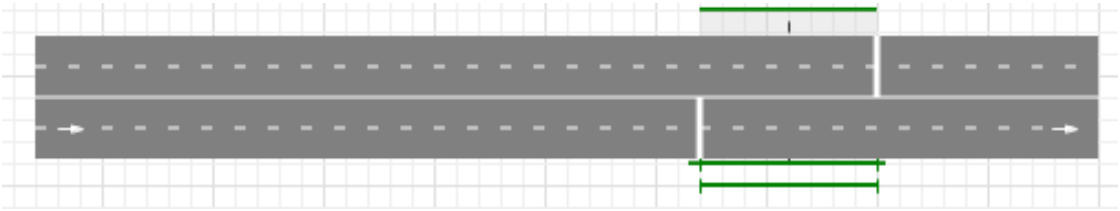
проверять, с какой полосы данный поворот осуществляется и только после этого разрешать сам поворот. Второй вариант отслеживания «правильных» поворотов – создание нескольких потоков машин, движущихся из данной точки. Поток, который может выполнить поворот, должен иметь строго заданную полосу.

5.2. Задание к работе

Задание к работе

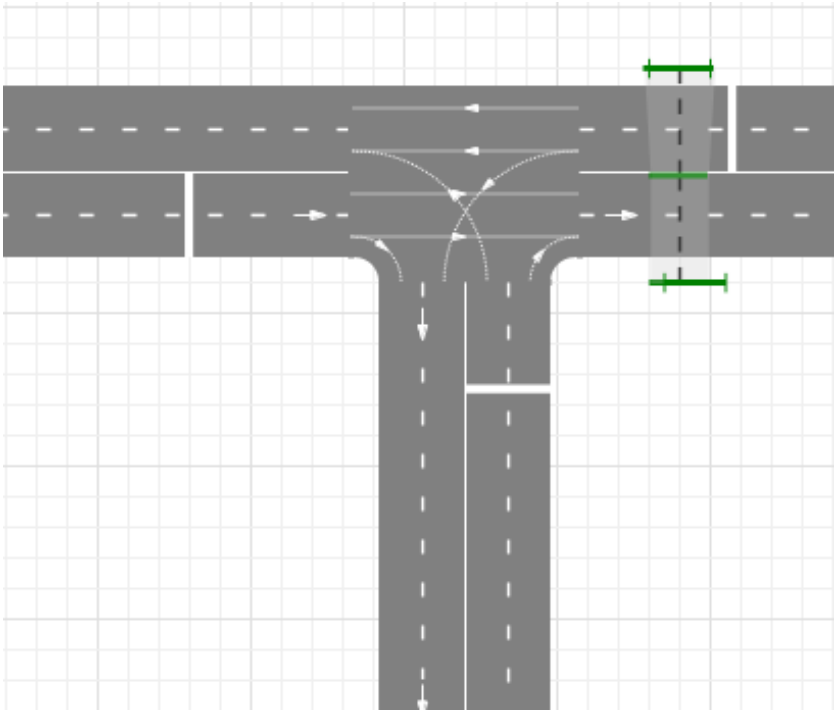
Смоделировать автомобильную дорогу с пешеходным переходом, воспользовавшись компонентами: дорога, две стоп-линии, светофор (для дороги); направление пешеходного потока, место ожидания (линию или область ожидания перед дорогой).

А) на оценку удовлетворительно и хорошо (рис. 5.13).



5.13. Модель дороги на оценку «хорошо»

Б) на оценку отлично



5.14. Модель дороги на оценку «хорошо»

Для данного варианта специфику светофора подобрать самостоятельно

Взять компонент Traffic Light и синхронизировать его работу с работой светофора из предыдущей лабораторной работы. Цвета «красный с желтым» и мигающий зеленый можно рассмотреть как желтый и зеленый соответственно. (*Учесть, что если расстояние от начала автомобильной дороги до стоп-линии будет достаточно коротким, то Traffic Light может не сработать*).

Создать поток транспорта и проверить работоспособность данной части модели.

Разработать алгоритм поведения пешеходов в соответствии с работой светофора (пешеходы должны двигаться на свой зеленый (или на красный свет для машин) и стоять перед дорогой на свой красный свет). Дополнительный светофор для пешеходов можно не делать. Создать модель движения пешеходов. Достаточно будет движения пешеходов в одну сторону.

Отчет по лабораторной работе должен содержать:

– постановку задачи;
– описание процесса построения имитационной модели;
– скриншоты с результатами;
– вывод о проделанной лабораторной работе.

Примечание. Как было отмечено ранее (после рис. 5.13), для того, чтобы поток машин автоматически останавливался автоматически по красному цвету светофора, необходимо стоп-линии поместить на достаточно удаленное расстояние от начала/окончания дороги.

ЛАБОРАТОРНАЯ РАБОТА 6
МОДЕЛИРОВАНИЕ СИСТЕМ, ИСПОЛЬЗУЮЩИХ РЕСУРСЫ

Цель работы: получение навыков моделирования сложных систем, особенностями которых является использование разного рода ресурсов.

6.1. Краткие теоретические сведения

Большинство обслуживающих и производственных систем использует разнообразные ресурсы. К ним можно, в частности, отнести, специалистов, выполняющих работу, необходимое оборудование, материалы и т.д. В Anylogic ресурсы бывают трех типов:

- статические (привязаны к определенному местоположению и не могут перемещаться ни самостоятельно ни принудительно);
- двигающиеся (могут перемещаться самостоятельно);
- перемещаемые (могут перемещаться агентами или движущимися ресурсами).

Каждый набор ресурсов задается блоком **ResourcePool (***рис. 6.1***).**

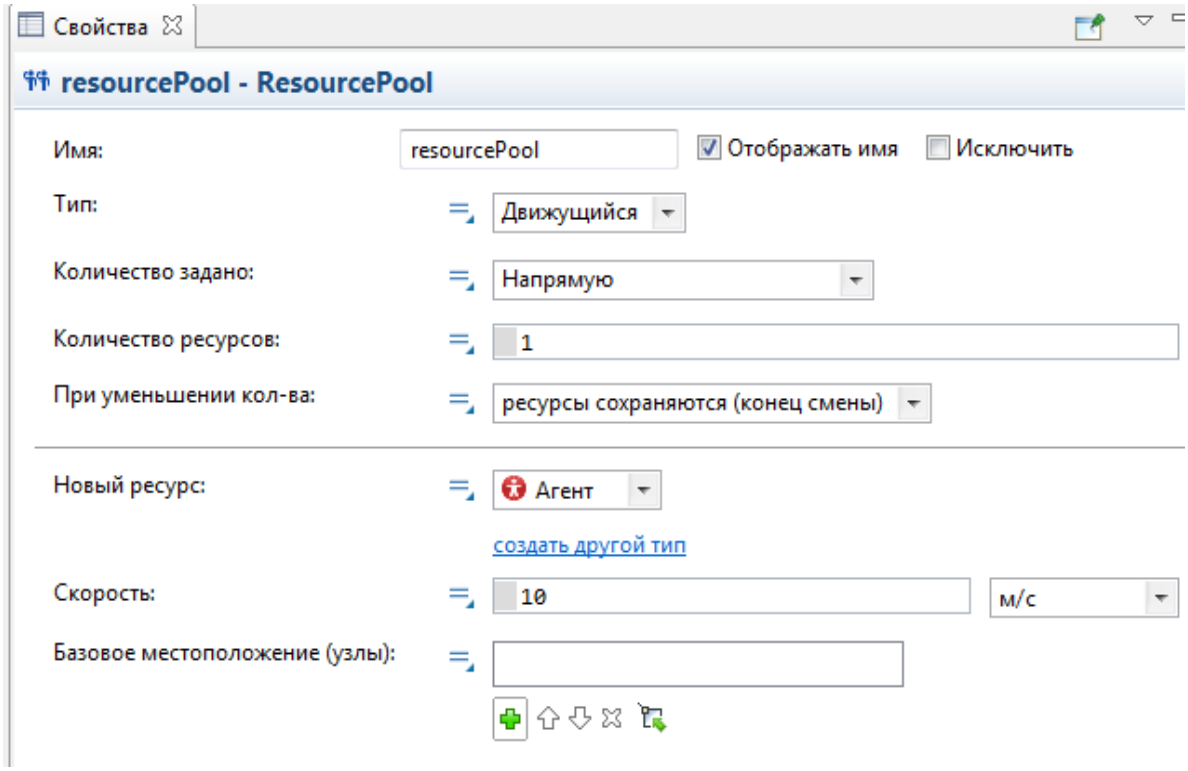


Рис. 6.1. Задание ресурсов с помощью ResourcePool

Ресурсы могут быть заданы:
- напрямую;
- базовым местоположением;
- расписанием;
- расписанием доступности;
- сменами: расписаниями групп;
- планом смен.

Если ресурс задан напрямую, то ниже необходимо определить количество ресурсов и как будет вести себя ресурс в течение работы с ним. Ресурсы могут принадлежать к одному из двух видов:
- ресурсы, безвозвратно расходующиеся на стадии обслуживания;
- ресурсы, сохраняющиеся при завершении обслуживания.

К первому типу относятся, например, материалы, а ко второму – инструменты, оборудование. Кроме того, в свойствах можно задать принадлежность ресурса к некоторому объекту.

Задание ресурсов базовым местоположением целесообразно для статических ресурсов, которые не могут перемещаться.

Кроме того, в anylogic есть объекты типа «Расписание», на основании которых можно также задать ресурсы.

Для ресурса можно использовать стандартный агент или создать нового агента (например, если требуется сформировать изображение).

Базовое положение формируется с помощью разметки пространства (точки, прямоугольные узлы и т.д.).

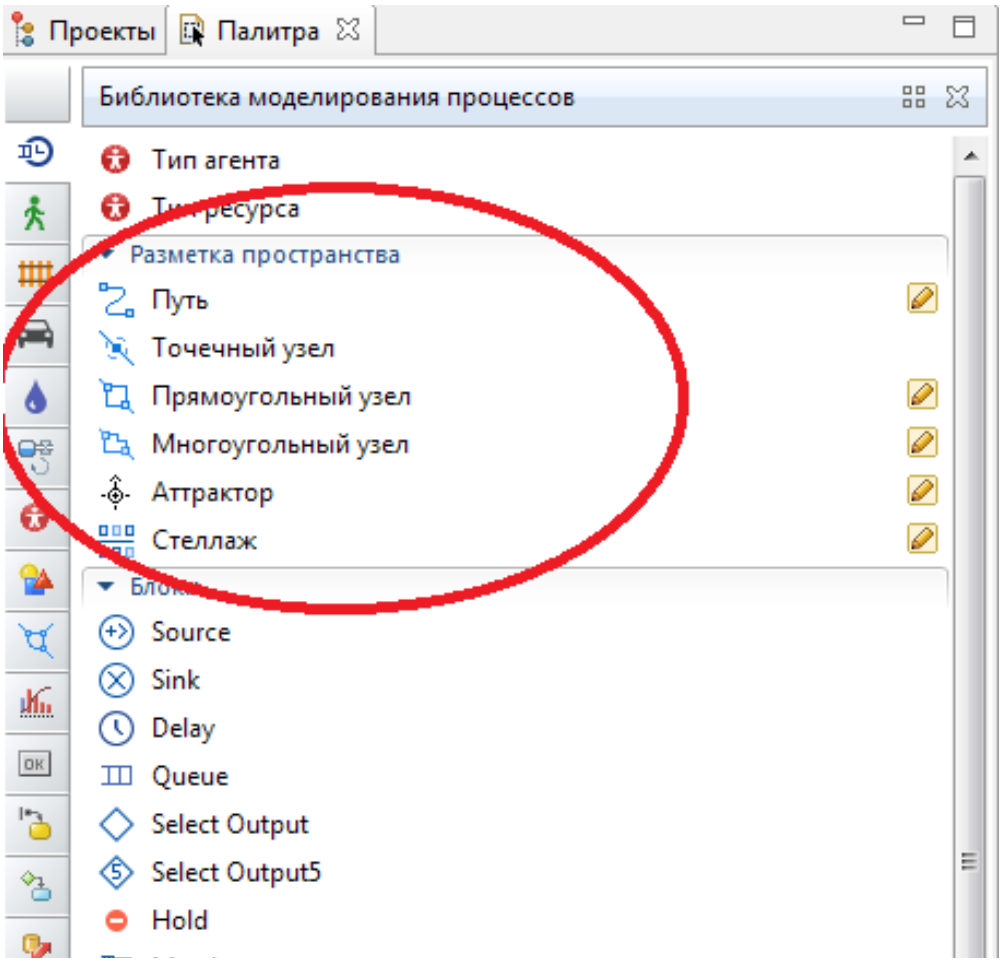


Рис.6.2. Базовое положение ресурсов

Если в системе используются несколько разных видов ресурсов, то для каждого из них необходимо создать элемент «ResourcePool».

Seize

Seize Захватывает заданное количество ресурсов для выполнения некоторой операции.

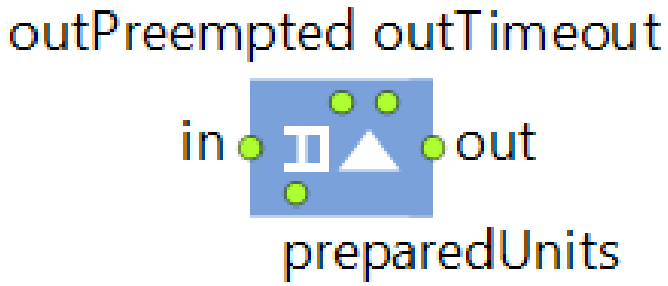


Рис.6.3. Порты объекта Seize

Заявка может покинуть объект Seize различными способами:

- Обычным способом через порт Out, когда объект, следующий за объектом Queue, готов принять заявку.
- Через порт TimeOut, если заявка проведет в объекте заданное количество времени. В этом случае в свойствах объекта необходимо выбрать «Разрешить уход по таймауту» и в появившейся строке задать время;
- Через порт OutPreempted, будучи вытесненной другой заявкой. В этом случае в свойствах объекта необходимо выбрать «Разрешить вытеснение».

Захватить можно ресурсы одного типа (в этом случае выбирается тип ресурсов и указывается количество) или альтернативный набор ресурсов (в этом случае требуется последовательно добавлять тип захватываемых ресурсов и их количество).

Если движущиеся ресурсы в данный момент располагаются в другом месте, их можно переместить к заявке (или к некоторому узлу), выбрав «Пересылать захваченные ресурсы». В этом случае появляется новое поле «Место назначения», которое может принимать следующие значения:

- Агент (который захватил данный ресурс);
- узел сети;
- аттрактор;
- другой захваченный ресурс (появляется дополнительное поле, в котором можно выбрать ресурс);
- базовый узел захваченного ресурса;
- точка.

При необходимости можно присоединить захваченные ресурсы.

Release – освобождает заданное количество ресурсов после выполнения операции. Можно освобождать:

1. Все захваченные ресурсы любого типа. В этом случае, все ресурсы всех типов, которые были заняты, становятся свободными.
2. Все ресурсы, захваченные данным блоком seize. В этом случае необходимо выбрать блок seize, которые захватил ресурсы, и все захваченные ресурсы в количестве, заданном в блоке seize, станут свободными.
3. Все захваченные ресурсы данного типа. Появляется строка «Объекты ResourcePool», в которой необходимо указать тип ресурсов. После этого все ресурсы данного типа станут свободными.
4. Заданные ресурсы (список типов). Указывается тип освобождаемых ресурсов и количество по каждому типу.
5. Указанное количество ресурсов (выбирается конкретный объект типа ResourcePool и количество освобождаемых ресурсов этого типа).

При освобождении можно указать, что делать с движущимися ресурсами (они возвращаются в базовую точку или остаются на месте).

Пример. Имеется производственный участок, в котором работают 3 человека. Процесс изготовления изделия состоит из следующих этапов:

- сборки изделия (выполняется одним (любым) рабочим в течение 30±5 мин);
- его обработки на станке в течение 15±2 мин.

Изделия поступают в среднем через 50±5 мин. Смоделировать процесс обслуживания.

Решение. Создадим два элемента типа «ResourcePool»: один для рабочих(ResourcePool_W); другой – для станка (ResourcePool_M). Для данной постановки задачи пока не принципиально, к какому типу ресурсов будут относиться рабочие и станки. Тем не менее, укажем в типе ResourcePool_W – движущийся; в типе ResourcePool_M – статический.

После поступления ресурсов в модель, сначала необходимо занять ресурсы типа ResourcePool_W (в количестве одной единицы); после чего имитировать обслуживание с помощью блока «Delay». Поскольку после этого этот же рабочий будет обрабатывать изделие на станке, то, не освобождая ресурс ResourcePool_W, необходимо занять ресурс ResourcePool_M. Далее имитируем процесс обработки изделия на станке (блок Delay), после чего объектом Release освобождаем сначала машину, потом - рабочего.

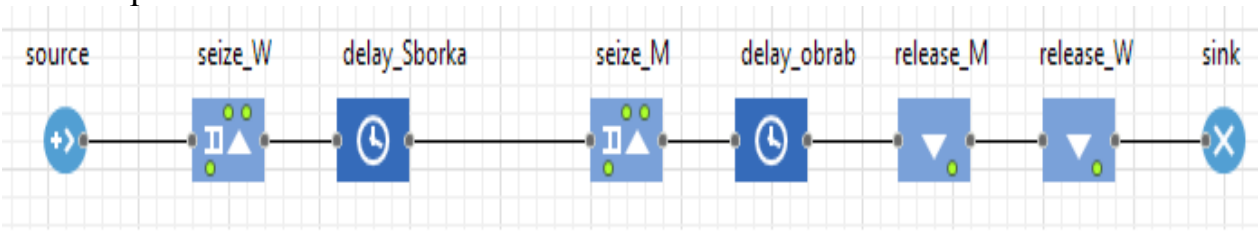


Рис. 6.4. Модель системы

resourceTaskStart и resourceTaskEnd

При работе с ресурсами часто возникает необходимость определить процесс подготовки к операции с данными ресурсами, а также завершение работы после того, как обслуживание будет закончено. Для этого используются блоки resourceTaskStart и resourceTaskEnd соответственно.

Service

Захватывает для агента заданное количество ресурсов, задерживает их, а затем освобождает захваченные им ресурсы. Эквивалентен последовательности объектов **Seize**, **Delay**, **Release** (и сам реализован именно таким способом) и должен использоваться в тех случаях, когда все,

что требуется - это задержать захваченные ресурсы на заданное время, а затем их отпустить. Большинство параметров этих вложенных объектов вынесены в интерфейс объекта Service.

ResourceSendTo - Посылает (перемещает) сетевые ресурсы из их текущего местоположения в заданный узел сети. Могут перемещаться только движущиеся ресурсы или переносные ресурсы в сопровождении движущихся. Ресурсы, пересылаемые этим объектом, могут находиться в различных местах, но в случае перемещения переносных ресурсов нужно, чтобы в месте нахождения каждого пересылаемого переносного ресурса находился и движущийся объект, который смог бы его переместить (такой движущийся ресурс должен быть также указан в списке перемещаемых объектом ресурсов). Агент покинет этот объект после прибытия последнего перемещаемого ресурса в заданный узел назначения. Каждая группа ресурсов, пересылаемых вместе, движется со скоростью самого медленного из этих ресурсов. Ресурс может быть перемещен:

- в заданный узел сети;
- в заданный аттрактор;
- в местоположение заданного агента;
- к захваченному ресурсу;
- к базовому узлу захваченного ресурса.

Ресурсы будут отображаться на анимации движущимися по кратчайшему из возможных путей от исходной точки до узла назначения. Агент при этом будет отображаться в ее текущем местоположении (в случайном месте внутри текущего узла сети).

MoveTo

Если есть необходимость перемещения агента, то можно использовать объект MoveTo. Если к агенту присоединены какие-то ресурсы, то они перемещаются вместе с агентом. Существуют следующие возможные точки перемещения агента:

- узел сети/ ГИС (при выборе появляется дополнительное поле с возможностью выбора узла);
- аттрактор (появляется поле с возможностью выбора аттрактора);
- захваченный ресурс;
- базовый узел захваченного агента;
- агент/ресурс (появляется поле с возможностью ввода агента);
- точка;
- узел +точка;
- широта, долгота;
- географическое место (появляется строка с возможностью ввода названия места);
- агент, который содержит меня;
- другой ресурс, захваченный моим агентом;
- базовый узел другого ресурса.

Хранение
Разметка пространства – стеллаж

Для хранения ресурсов или агентов в anylogic используются стеллажи. Стеллаж представляет собой элемент разметки пространства. Существует возможность выбора конфигурации стеллажа.

При работе со стеллажом используются два объекта, позволяющие загружать элементы в стеллаж и извлекать их.

Rack Store

Данный объект помещает агента в ячейку заданного стеллажа. Если к агенту присоединены какие-то ресурсы, то они перемещаются вместе с агентом. Если агент перемещается с помощью ресурсов, то RackStore захватывает их, перемещает в местоположение агента, присоединяет к агенту, перемещает агента к ячейке, а затем освобождает ресурсы.

Rack Pick

Извлекает агента из ячейки стеллажа и перемещает его в заданную точку пространства. При этом для перемещения агента могут использоваться движущие ресурсы. Если агент перемещается с помощью ресурсов, то Rack Pick захватывает их, переносит к ячейке, в которой хранится агент, присоединяет ресурсы к агенту, перемещает агента в заданный узел сети, а затем освобождает ресурсы.

Во всех динамических параметрах ресурс доступен как локальная переменная ***unit.***

6.2. Задание к работе

Смоделировать системы согласно варианту задания. Осуществить разметку пространства, поместив произвольным образом станки, роботов/автопогрузчиков и другие пункты модели, упомянутые в варианте задания. Отчет должен содержать:

- постановку задачи;
- разработанную модель с описанием ее проектирования;
- результат моделирования;
- выводы.

6.3. Варианты заданий

Вариант 1

Роботизированная производственная система имеет два станка, три робота, пункт прибытия и склад обработанных деталей. Детали прибывают на склад в среднем через 40±10 минут. Далее они захватываются одним из свободных роботов и перемещаются к первому станку, после чего робот освобождается. После завершения обработки на первом станке деталь захватывается роботом и перемещается на второй станок. После обработки

на втором станке деталь перемещается на склад обработанных деталей. Скорости передвижения роботов составляет 20 м/мин. Время обработки на первом станке распределено по нормальному закону со средним значением 50 мин и отклонением 10 мин. Время обработки на втором станке подчинено экспоненциальному закону со средним значением 40 мин.

Смоделировать работу цеха при условии, что каждый робот может использоваться на каждом из путей перемещения деталей. Определить загрузку роботов и станков.

Вариант 2

Роботизированная производственная система имеет два станка, три робота, пункт прибытия и склад обработанных деталей. Детали прибывают на склад в среднем через 40±10 минут. Далее они захватываются одним из свободных роботов и перемещаются к первому станку, после чего робот освобождается. После завершения обработки на первом станке деталь захватывается роботом и перемещается на второй станок. После обработки на втором станке деталь перемещается на склад обработанных деталей. Скорости передвижения роботов составляет 20 м/мин. Время обработки на первом станке распределено по нормальному закону со средним значением 50 мин и отклонением 10 мин. Время обработки на втором станке подчинено экспоненциальному закону со средним значением 40 мин.

Смоделировать работу цеха при условии, что каждый робот закреплен на определенном пути перемещения деталей (один робот: пункт прибытия – первый станок; второй: первый станок – второй станок; третий: второй станок - склад). Определить загрузку роботов и станков.

Вариант 3

Каждые три часа на завод приезжает грузовик с заготовками для деталей. В каждом грузовике помещается по 10 ящиков с заготовками. Эти ящики с помощью автопогрузчика помещаются в подготовительную зону хранения. Далее эти коробки по одной доставляются автопогрузчиками к станку с ЧПУ, на котором осуществляется производство конечных изделий. Время, затрачиваемое на производство изделий из одной коробки, составляет 20±5 мин.

Готовые изделия перевозятся автопогрузчиком в зону отгрузки.

Смоделировать работу завода. Определить загрузку роботов и станков.

Вариант 4

Производственный участок состоит из основного цеха, зоны прибытия и пункта контроля. В зону прибытия через 5±1 мин поступают детали, которые можно отнести к одному из двух типов. С вероятностью 0,4 деталь относится к типу 1 с вероятностью 0,6 – к типу 2. Из зоны

прибытия детали с помощью одного из двух автопогрузчиков доставляются в цех, в котором расположены два станка А и В и пункт контроля. Детали первого типа обрабатываются станком А за 10±3 мин, после перемещаются в пункт контроля. С вероятностью 0,1 деталь не отвечает требованиям и возвращается на повторную обработку на станок А, в противном случае с помощью она перемещается к зоне готовых деталей (покидает систему).

Детали второго типа обрабатываются станком В в среднем за 8±4 мин, после перемещаются в пункт контроля. С вероятностью 0,1 деталь не отвечает требованиям и возвращается на повторную обработку на станок В, в противном случае с помощью она перемещается к зоне готовых деталей (покидает систему).

Проверка в пункте контроля для деталей любых типов длится 3±1 мин. Смоделировать работу участка. Определить загрузку роботов и станков.

ЛАБОРАТОРНАЯ РАБОТА 7

АГЕНТНОЕ МОДЕЛИРОВАНИЕ

Цель работы: приобретение навыков разработки моделей на основе агентного подхода.

7.1. Краткие теоретические сведения

Под агентом понимается элемент модели, который может иметь поведение, память, историю, контакты и т.д. В качестве примеров агентов могут выступать люди, компании, товары, ресурсы и т.д.

Создание агента обычно начинается с определения его интерфейса для связи с внешним миром и/или с другими агентами.

Начальное состояние и поведение агента может быть реализовано различными способами. Состояние (накопленная история) может быть представлено с помощью переменных или состояния диаграммы состояний.

Агент может вести себя пассивно или активно. В случае пассивного поведения агент может только реагировать на прибытие сообщений или вызов методов. При этом он не имеет собственных событий. Если агент ведет себя активно, то у него существует внутренняя динамика, заданная в виде некоторой последовательности событий. В этом случае для такого агента необходимо разработать событие и/или диаграмму состояний.

Создание агентов

Для создания агентов необходимо перетащить в основное окно соответствующий элемент из библиотеки Агент. При этом можно создать:

- множество однотипных агентов;
- единственного агента;
- просто создать тип агента (*рис.7.1*).

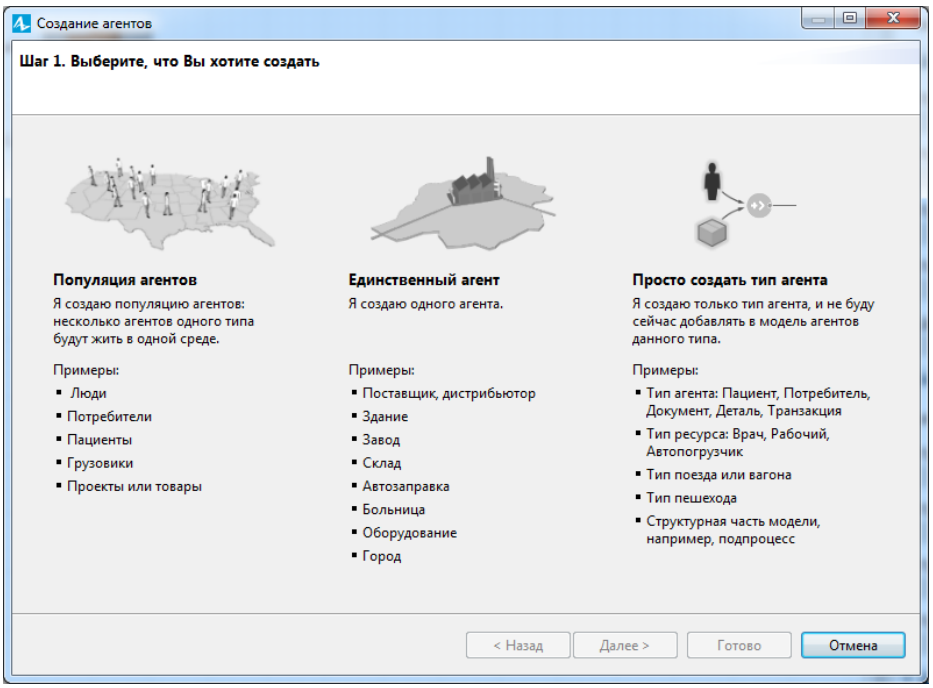


Рис. 7.1. Создание агентов

После этого агент будет виден на основной форме. Типичная архитектура агентной модели в Anylogic – агент Main, на диаграмме которого заданы все остальные агенты. В этом случае Main играет роль среды обитания для всех остальных агентов. Среда задает:

- тип пространства, в котором живут агенты;
- расположение агентов в пространстве;
- сеть контактов агентов, которая может использоваться при их общении друг с другом.

Остановимся более подробно на параметрах агента.

Изменить свойства агента в любой момент времени можно с помощью оператора присваивания вида:

<агент>.<параметр> = значение

В Anylogic доступна локальная переменная Agent, которая ссылается на того агента, который в настоящий момент активен (активизировал данное действие).

Пример.

На вход устройства поступают два потока заявок. Заявки первого типа поступают в среднем через 20с и обслуживаются 10±2 с. Заявки второго типа поступают в среднем через 15с и обслуживаются 8±2 с. Перед устройством имеется неограниченная очередь. Смоделировать работу системы.

Решение. Поскольку длительность обслуживания является индивидуальной для каждой заявки, целесообразно эту длительность отражать в свойствах заявки. Для этого требуется создать специальный тип агентов, имеющий одно дополнительное свойство «длительность». Это будет популяция агентов (поскольку это будет источник заявок).

Создадим модель согласно *рис. 7.2*.

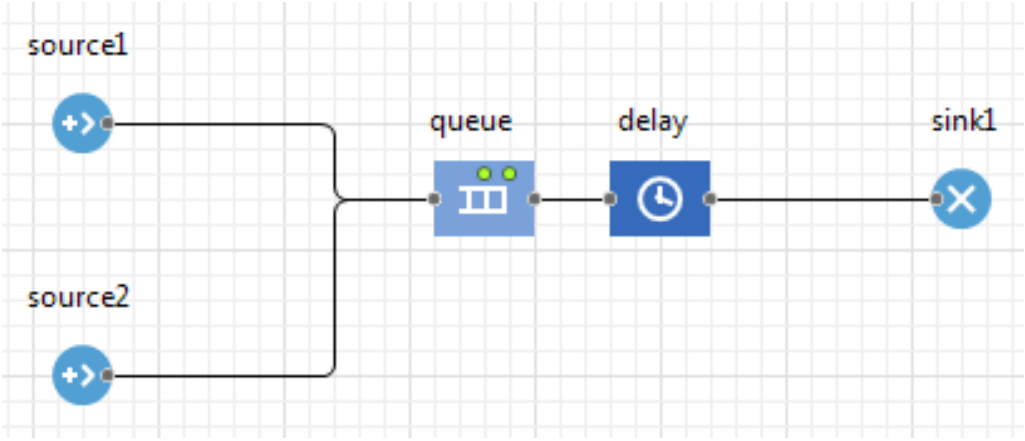


Рис. 7.2. Схема модели

Зайдем в свойства элемента source1 и в действиях выберем «действие при выходе» и напишем код:

Agent.длительность = 10;

Аналогичным образом опишем «действие при выходе» для элемента source2:

Agent.длительность = 8;

Далее необходимо эту длительность задать в блоке delay. Зададим следующее время задержки:

uniform(agent.длительность, 2)

Модель будет работать следующим образом. Источник source1 будет создавать агентов, и, как только они будут покидать данный блок, в свойство агентов «длительность» будет присваиваться значение 10. Аналогичным образом, все агенты, которые будут созданы источником source2, будут иметь в поле «длительность» значение 8. Блок delay будет анализировать свойство «длительность» поступившей заявки. Более точно, данный блок будет задерживать заявку на время, которое записано в ее свойстве «длительность».

Специфика функционирования многоагентной системы заключается в том, что агенты, функционируя, постоянно взаимодействуют между собой и, в зависимости от полученной информации, принимают те или иные решения. Исходя из этого, важнейшим аспектом реализации многоагентной системы является обеспечение их взаимодействия.

Основным способом организации взаимодействия агентов в anylogic являются сообщения. Агенты взаимодействуют посредством передачи

друг другу сообщений. Получив некоторое сообщение, агент выполняет определенное действие, тем самым реагируя на это сообщение. Таким образом, для взаимодействия агентов необходимо обеспечить возможность посылать сообщения и выполнять действия при получении сообщения.

Пересылка сообщений осуществляется посредством функции send. Для нее необходимо указать два параметра:

- сообщение;
- кому это сообщение предназначается.

Сообщение может иметь разные типы. В простейшем случае оно может иметь тип String. Можно использовать или создавать более сложные типы сообщений (например, тип агента, в котором будут содержаться необходимые параметры).

Для формирования реакции на сообщение в anylogic разработан объект «Connections». содержит связи с контактами этого агента и задает настройки взаимодействия. Данный элемент располагается над осью X в графическом редакторе агента (*рис.7.3*).

Для задания реакции на получения сообщения необходимо:

1. Открыть диаграмму агента-получателя сообщения.
2. Щелкнуть мышью по его элементу connections.
3. Открыть секцию свойств «Взаимодействие»
4. Ввести соответствующий код в свойстве «Действие при получении сообщения».
5. Реакцию на получение сообщения можно организовать в виде диаграммы состояний или в виде цепочки действий дискретно-событийного моделирования. В первом случае необходимо, чтобы был выбран выбрать пункт «Перенаправлять сообщение в диаграммы состояний» и далее в таблице будут отображены все диаграммы состояний данного агента. Во втором случае действия можно начать с создания источника заявок source или вставить агента в уже существующую диаграмму. В первом случае в свойстве «Действие при получении сообщения» необходимо ввести код «source.inject(1);», а в свойствах объекта source задать прибытие заявок согласно функции inject. В случае вставки агента в свойстве «Действие при получении сообщения» необходимо ввести код enter.take(msg), где msg – данное сообщение.

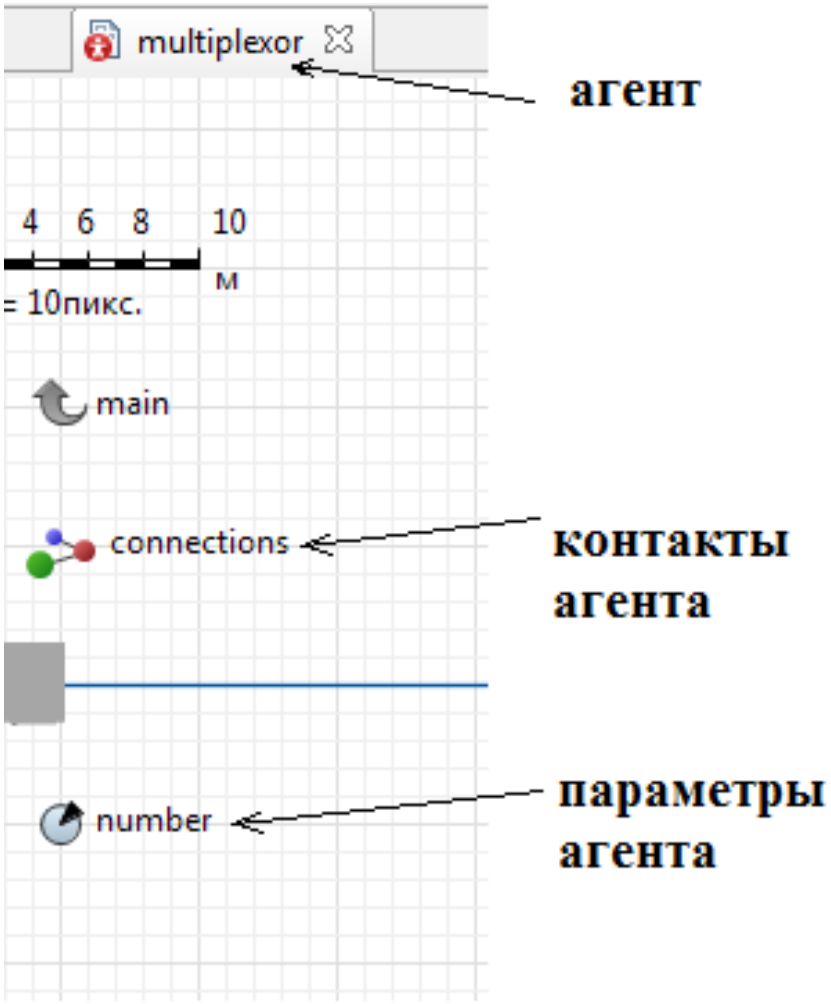


Рис. 7.3. Окно агента

Области видимости

Поскольку агенты имеет некоторую принадлежность к определенному местоположению, для обращения к ним необходимо соблюдать правила, продиктованные областью видимости тех или иных элементов. Рассмотрим пример области видимости, который приведен в справочной системе Anylogic.

Поясним данный рисунок. Если необходимо обратиться к некоторому объекту, находясь в агенте Company, который был создан из агента main, то функция revenue, располагающаяся в этом же агенте, вызывается напрямую, без дополнительных указаний (A). Аналогичным образом было определено обращение к размеру очереди queue, которая принадлежит агенту Company. Если необходимо обратиться к функции агента, который был непосредственно создан в данном агенте (на *рис. 7.4* это агент Employee), то обращение производится через указание конкретного агента.

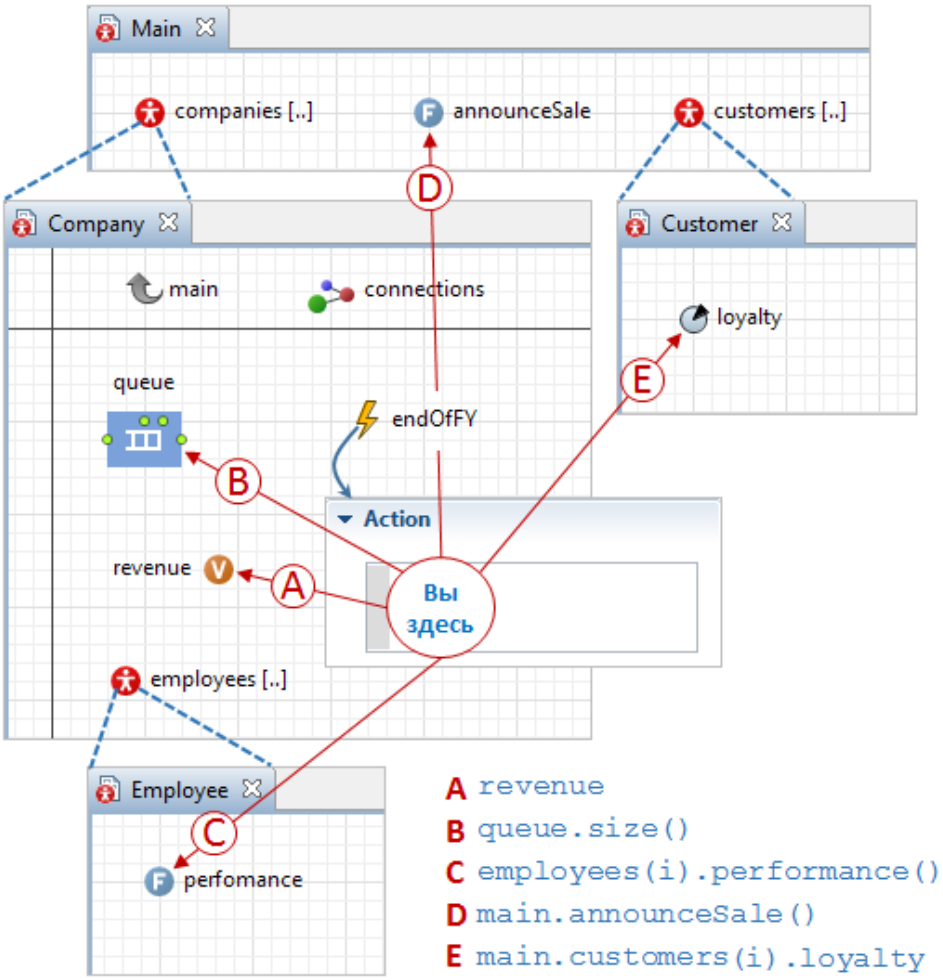


Рис. 7.4. Области видимости

Поскольку была создана популяция агентов (что обозначено символами […]), то требуется указать конкретного агента, а затем – функцию: employees(i).perfomance(). Если же требуется обратиться к функции, которая принадлежит главному агенту main, то необходимо сначала обратиться к агенту main, а затем указать эту функцию: main.announceSale(). Аналогичным образом осуществляется обращение в случае, если требуется обратиться к другому агенту, располагающемуся на аналогичном уровне относительно агента main. В этом случае сначала указывается имя главного агента (main), затем – конкретный агент из созданной популяции агентов, а далее - его параметр: main.customers(i).loyalty.

7.2. Задание к работе

Разработать модель системы, согласно варианту. Определить необходимые характеристики

Отчет должен содержать:

- постановку задачи;

- модель с подробным ее описанием;
- результаты моделирования;
- выводы.

7.3. Варианты заданий

Лабораторная работа по теме «Агентное моделирование»

Вариант 1.

В мини-лаборатории осуществляется анализ и обработка данных. Данные поступают в лабораторию через время t_1. Обработка данных осуществляется на ЭВМ1 или ЭВМ2 (в соответствии с вариантом задания) за время t_2. После этого данные попадают на предобработку в ЭВМ3, которая длится t_3 для заданий, пришедших с ЭВМ1 и t_4 для заданий, обрабатываемых на ЭВМ2. Анализ данных осуществляется на ЭВМ4 или ЭВМ5, причём данные, пришедшие с ЭВМ1 анализируются на ЭВМ4, с ЭВМ2- на ЭВМ5. Время анализа для ЭВМ4 и ЭВМ5 – t_5. В случае занятости каждой ЭВМ задания образую очередь, однако суммарное число заданий, присутствующих в системе с учётом ожидающих заданий, не должно превышать К. Задание, пришедшее в момент, когда длина очереди больше или равна К, получает отказ.

Смоделировать работу системы. Найти вероятность отказа и загрузку каждой ЭВМ.

Варианты	Параметры						
	T_1	Обработка	T_2	T_3	T_4	T_5	K
1.1	40	B1	80	35	45	70	10
1.2	25	B2	50	30	25	45	7
1.3	30	B3	60	28	32	55	5
1.4	20	B4	40	25	18	40	10

Варианты обработки.

B1 – сначала на ЭВМ2, а в случае её занятости – на ЭВМ1;

B2 – с вероятностью 0.55 – на ЭВМ2, с вероятностью 0.45 – на ЭВМ1;

B3 – равновероятно на любую ЭВМ;

B4 – сначала на ЭВМ1, а в случае её занятости – на ЭВМ2.

Вариант 2.

Участок цеха состоит из сортировочного и сборочного отделов. Детали поступают в среднем через 20±5 мин и с вероятностью 0.4 относятся к деталям первого типа, с вероятностью 0.6 – второго. Сортировка деталей и перенаправление их к станкам 1 и 2, которые обрабатывают детали первого и второго типов соответственно, проводится в течение 5±1 мин. Первый станок обслуживает детали в среднем за 30±5 мин; второй – за 40±5 мин. Первый станок допускает 5% брака; второй – 8%. Все бракованные детали возвращаются на обработку на первый станок. Детали, отбракованные второй раз, считаются браком и покидают систему.

После первичной обработки детали поступают на станок 3 на детальную обработку, которая длится 20±2 мин для деталей первого типа; 25±3 – для деталей второго типа.

Смоделировать работу системы. Определить загрузку каждого станка, вероятностно-временные характеристики очередей перед ними и вероятность отказа.

Вариант 3

Система передачи данных обеспечивает передачу пакетов данных из пункта *A* в пункт *C* через транзитный пункт *B*. В пункт *A* пакеты поступают через 10±5 мс. Здесь они буферизуются в накопителе емкостью 20 пакетов и передаются по любой из двух линий *AB1*-за время 20 мс или *AB2* - за время 20±5 мс. В пункте *B* они снова буферизуются в накопителе емкостью 25 пакетов и далее передаются по линиях *BC1* (за 25±3 мс) и *BC2* (за 25 мс). Причем пакеты из *AB1* поступают в *BCl*, а из *AB2* - в *BC2*. Чтобы не было переполнения накопителя, в пункте *B* вводится пороговое значение его емкости - 20 пакетов. При достижении очередью порогового значения происходит подключение резервной аппаратуры и время передачи снижается для линий *BC1* и *BC2* до 15 мс.

Смоделировать прохождение через систему передачи данных 500 пакетов. Определить вероятность подключения резервной аппаратуры и характеристики очереди пакетов в пункте *B*. В случае возможности его переполнения определить необходимое для нормальной работы пороговое значение емкости накопителя.

Вариант 4

В системе передачи цифровой информации передается речь в цифровом виде. Речевые пакеты передаются через два транзитных канала, буферируясь в накопителях перед каждым каналом. Время передачи пакета по каналу составляет 5 мс. Пакеты поступают через 6±3 мс. Пакеты, передававшиеся более 10 мс, на выходе системы уничтожаются, так как их появление в декодере значительно снизит качество передаваемой речи. Уничтожение более 30% пакетов недопустимо. При достижении такого

уровня система за счет ресурсов ускоряет передачу до 4 мс на канал. При снижении уровня до приемлемого происходит отключение ресурсов.

Смоделировать 10 с работы системы. Определить частоту уничтожения пакетов и частоту подключения ресурса.

Вариант 5

В узел коммутации сообщений, состоящий из входного буфера, процессора, двух исходящих буферов и двух выходных линий, поступают сообщения с двух направлений. Сообщения с одного направления поступают во входной буфер, обрабатываются в процессоре, буферизуются в выходном буфере первой линии и передаются по выходной линии. Сообщения со второго направления обрабатываются аналогично, но передаются по второй выходной линии. Применяемый метод контроля потоков требует одновременного присутствия в системе не более трех сообщений на каждом направлении. Сообщения поступают через интервалы 15±7 мс. Время обработки в процессоре равно 7 мс на сообщение, время передачи по выходной линии равно 15±5 мс. Если сообщение поступает при наличии трех сообщений в направлении, то оно получает отказ.

Смоделировать работу узла коммутации в течение 10 с. Определить загрузки устройств и вероятность отказа в обслуживании из-за переполнения буфера направления. Определить изменения в функции распределения времени передачи при снятии ограничений, вносимых методом контроля потоков.

Вариант 6

Пусть имеется отдел реализации и отдел снабжения. Отдел реализации занимается изготовлением некоторых деталей. Каждая деталь требует комплектующих, которые расходуются в среднем за 2-3 часа в количестве 1 единицы. Как только количество комплектующих сокращается до трех единиц, оформляется заказ отделу снабжения, и через 10-15 часов комплектующие в количестве 20 единиц будут доставлены. Если количество комплектующих в отделе реализации равно нулю, возникает простой.

Смоделировать процесс функционирования и взаимодействия данных отделов. На основном окне реализовать визуализацию доставки деталей из пункта А (где находятся детали) в пункт В (где отдел реализации) с помощью фуры. Найти вероятность простоя.

ЛАБОРАТОРНАЯ РАБОТА 8

СИСТЕМНАЯ ДИНАМИКА

Цель работы: освоение практических навыков решения задач дискретной оптимизации метаэвристисескими методами (генетическим методом и методом муравьиных колоний).

Освоение теоретических навыков разработки эвристических и метаэвристических алгоритмов, а также получение практических навыков программной реализации данных алгоритмов.

8.1. Краткие теоретические сведения

Дифференциальное уравнение в системной динамике описывается при помощи накопителя, потока, а также динамической переменной или параметра. Накопитель описывает непосредственно производную, динамическая переменная или параметр – переменные, составляющие правую часть уравнения.

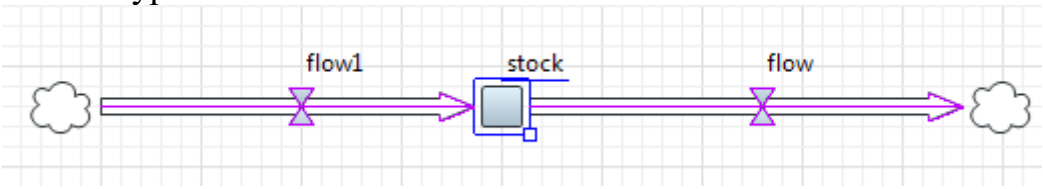


Рис. 8.1. Накопитель и потоки

В свойствах накопителя необходимо описать его начальное значение (по умолчанию это 0) и уравнение, отражающее влияние потоков на изменение состояния накопителя.

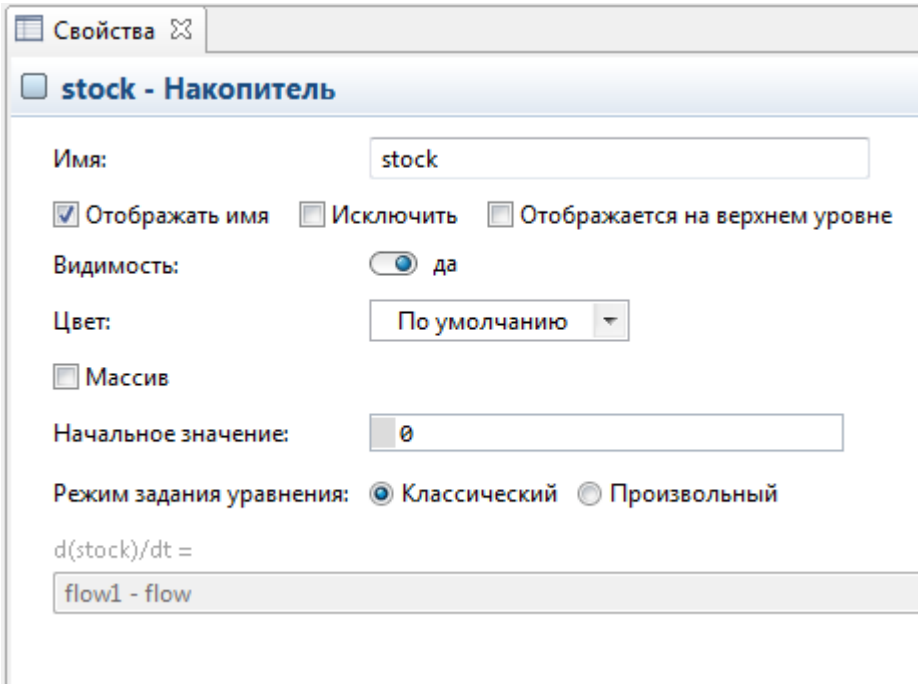


Рис. 8.2. Свойства накопителя

В классическом варианте изменение (т.е. производная по времени) накопителя зависит со знаком плюс от входящего потока и со знаком минус от исходящего. Если зависимость более сложная (например,

значение потока необходимо умножить на параметр и т.п.), в свойствах «Режим задания уравнения» следует выбрать «произвольный» и описать зависимость.

Значение потока в общем случае зависит от множества параметров и/или динамических переменных. В этом случае необходимо обеспечить входящую связь от каждого такого параметра (переменной) к потоку (рис. 3).

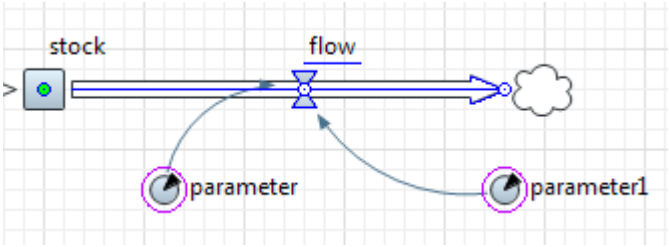


Рис. 8.3. Зависимость потока от параметров и переменных

После этого, в свойствах потока требуется описать уравнение, включающее ВСЕ связанные параметры/переменные (*рис. 8.4*).

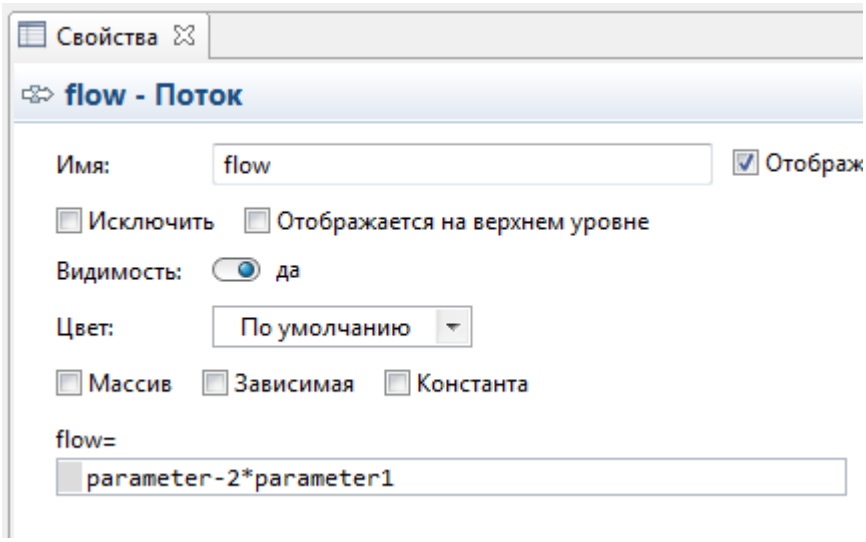


Рис. 8.4. Свойства потока

Выбор динамической между динамической переменной и параметром зависит от того, можем или не можем управлять этим элементом. Если управление возможно, то необходимо использовать параметр, если нет, то это динамическая переменная.

Например, темп роста населения ни от чего не зависит, следовательно, он не является управляемой переменной.

Эксперимент Варьирование параметров

С помощью данного эксперимента можно выполнить несколько прогонов модели, последовательно выбирая для каждого прогона свою комбинацию параметров.

Необходимо задать для каждого варьируемого параметра его минимальное и максимальное значение, а также величину шага.

Для запуска эксперимента необходимо перейти на вкладку «Проекты», выбрать «Main» и, открыв контекстное меню, выбрать команду «Создать»/ «Эксперимент».

Из предложенных вариантов необходимо выбрать «варьирование параметров».

Далее необходимо настроить параметры для эксперимента. По умолчанию они имеют фиксированный тип. При необходимости изменения значений выбирается тип «Диапазон», после чего задается наибольшее и наименьшее значение.

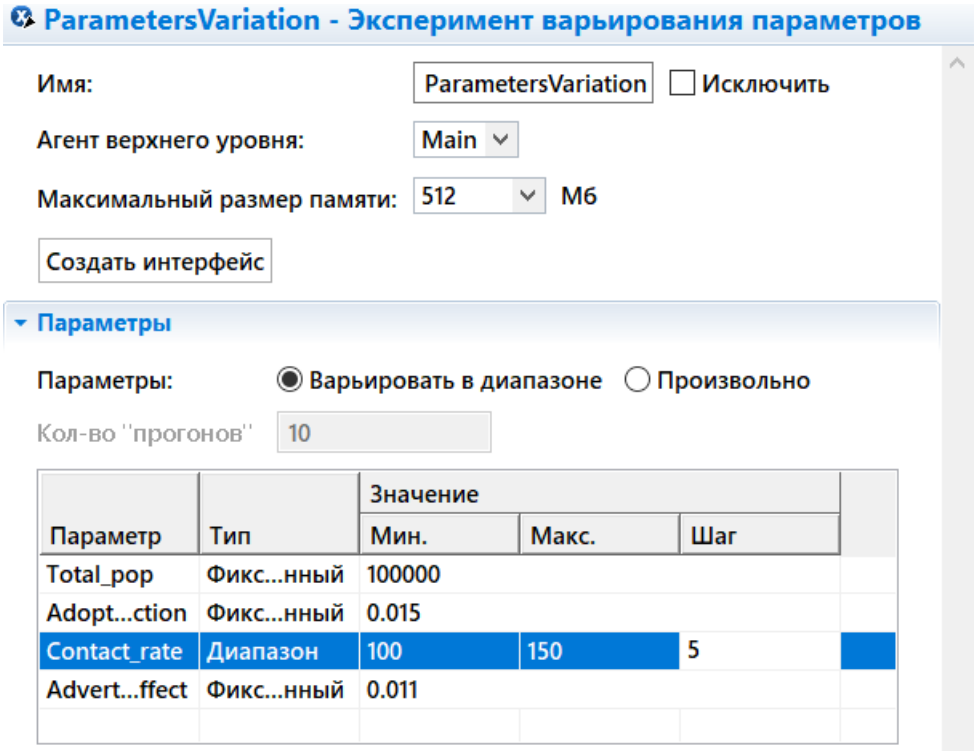


Рис. 8.5. Настройка параметров

Щелчок по кнопке «Создать интерфейс» позволит вывести все результаты эксперимента.

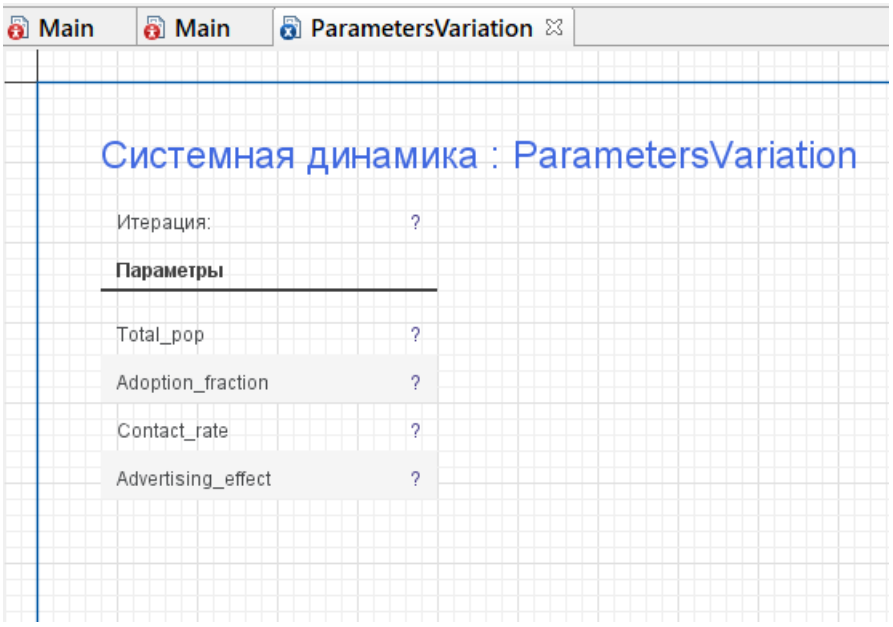


Рис. 8.6. Окно «Варьирование параметров»

Для запуска эксперимента необходимо, запуская модель, выбрать сохраненный эксперимент.

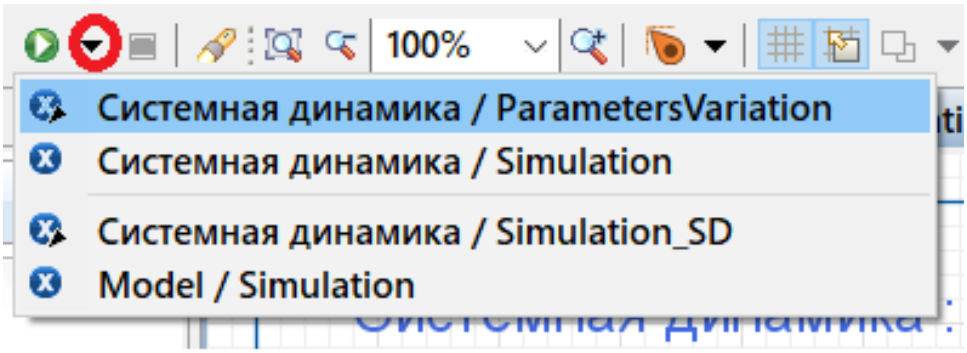


Рис. 8.7. Вызов эксперимента

Возможно получить графики изменения некоторой величины для каждого из значений параметров. Для этого поместим график в рабочее окно варьирования параметров (*рис. 8.8*).

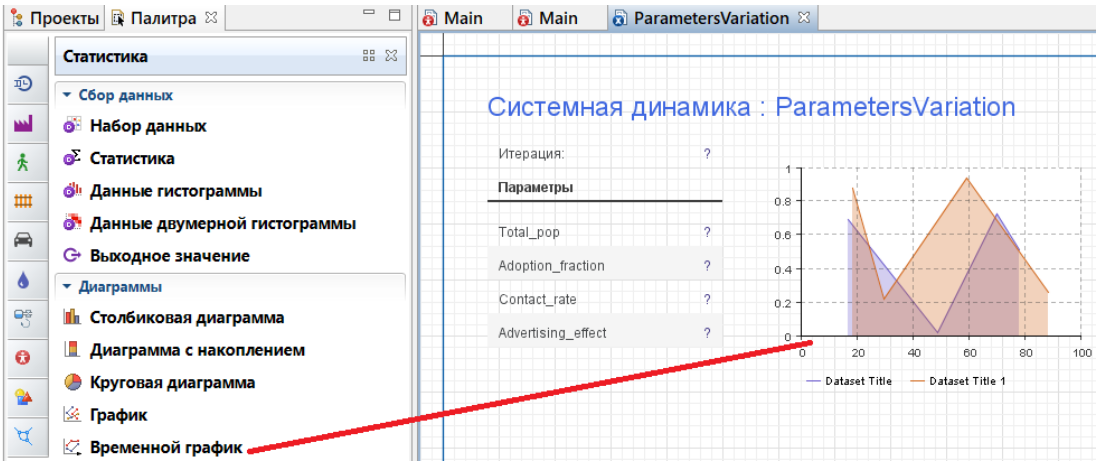


Рис.8.8. График для эксперимента

Теперь необходимо добавить данные в график plot. Это выполняется с помощью свойства AddDataSet. У свойства addDataSet есть нескольтко нотаций. В данном случае будет использоваться нотация addDataSet(DataSet ds, String title), где в качестве первого аргумента используется набор.

Чтобы автоматически получить такой набор данных, необходимо вызвать контекстное меню у интересующего состояния и выбрать команду «Создать набор данных» (*рис. 8.9*).

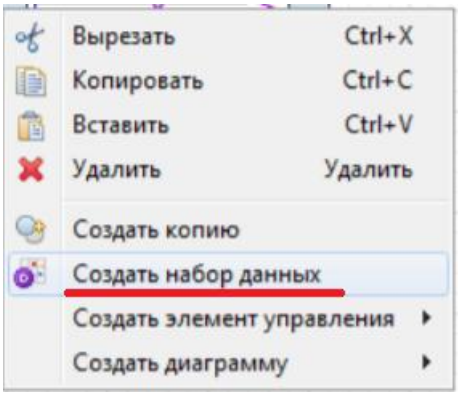


Рис. 8.9. Создание объекта DataSet

После этого в свойствах эксперимента во вкладке «Действия Java» опишем следующее действие после прогона модели (*рис. 8.10*).

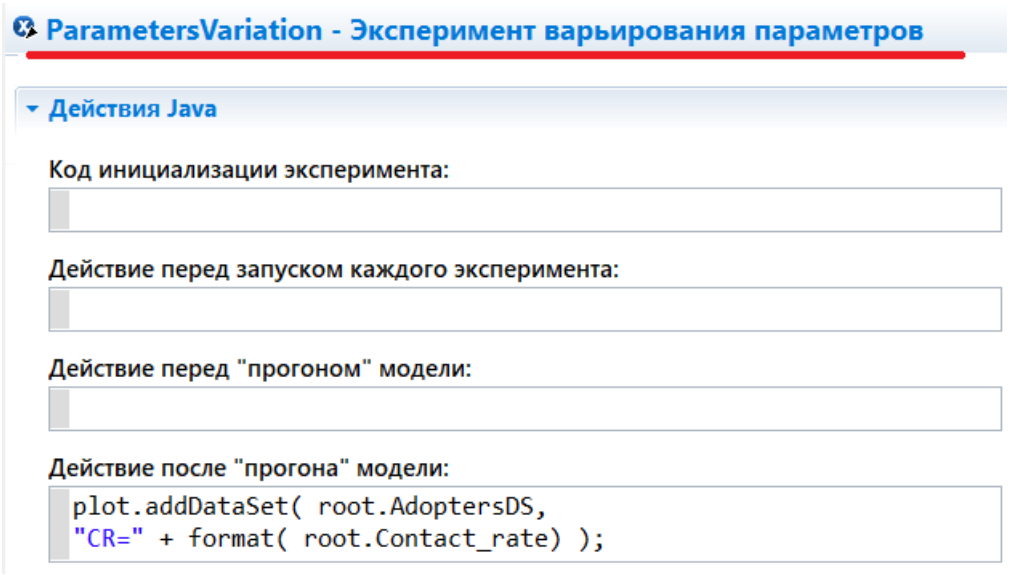


Рис. 8.10. Настройка графика

После этого при запуске Варьирования параметров будут показаны графики для каждого из изменяющихся параметров.

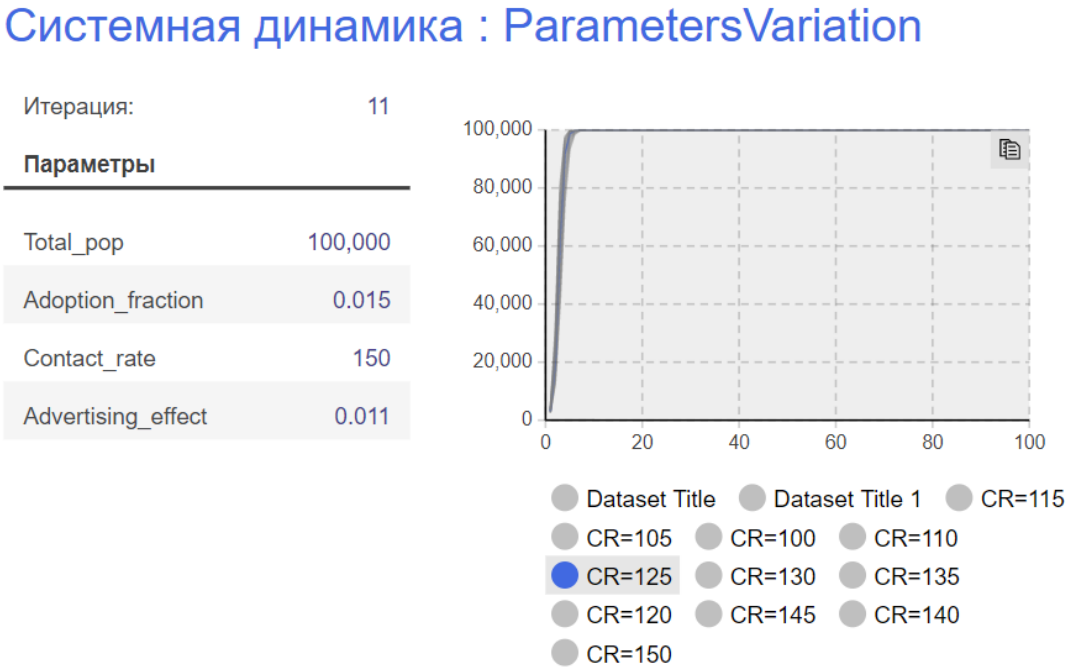


Рис. 8.11. Результаты моделирования

По данным графикам можно сделать вывод о несущественной зависимости исследуемого состояния (для которого строился график) от изменяемого параметра CR.

Рассмотрим другой пример результирующих графиков. По данным изображениям видна бОльшая зависимость исследуемого накопителя от изменяемого параметра (*рис. 8.12*).

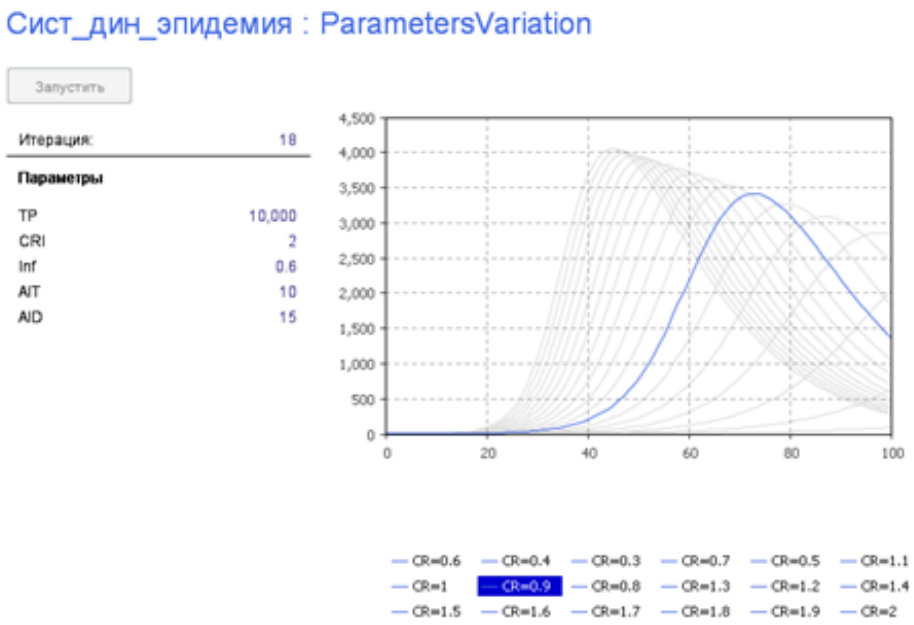


Рис. 8.12. Графики, иллюстрирующие зависимость накопителя от изменяемого параметра

8.2. Задание к работе

Построить модель согласно варианту, используя библиотеку системной динамики. Проанализировать зависимость заданного параметра модели от указанной характеристики с помощью варьирования параметров. Сделать выводы о функционировании исследуемой системы. Отчет должен содержать:

- постановку задачи;
- разработанную модель с детальным описанием ее отдельных блоков;
- результаты моделирования;
- выводы.

8.3. Варианты заданий

Вариант 1 Модель Солоу

Смоделировать систему, которая описывается следующими уравнениями.

$$\frac{dP}{dt} = \alpha_p \cdot P;$$

$$\frac{dK}{dt} = i \cdot V - \mu \cdot K;$$

$$V = A \cdot P^{\alpha_1} \cdot K^{\alpha_k};$$

$$c = \frac{(1-i)\cdot V}{P}$$

Здесь используются следующие обозначения:

P – численность населения (начальное значение 10000);

K – объем капитала (начальное значение 10000);

V – объем производства;

α_p – темп роста населения (0.002);

i- Норма накопителя (0.5);

μ - коэффициент амортизации капитала (0.1);

A – коэффициент НТП (1);

α_1 – коэффициент эластичности производства по труду (0.7);

α_2 – коэффициент эластичности производства по капиталу (0.3);

C –потребление на душу населения.

Для того, чтобы возвести один аргумент (arg1) в степень, равную другому аргументу (arg2), можно воспользоваться функцией

Pow (arg1, arg2)

Отобразить графически изменение Потребления на душу населения «C» от времени.

Поместить на форму бегунок и связать его с параметром i. Изменять значение данного параметра (от 0 до 1) и проанализировать специфику изменения C от i.

Провести эксперимент: проанализировать изменение потребления C во времени в зависимости от коэффициента i.

Вариант 2 Модель распространения эпидемии

Построить тривиальную модель, описывающую распространение инфекционного заболевания среди населения.

Модель строится при следующих условиях:

1. Рассматривается численность населения TotalPopulation (**TP**)=10000
2. Во время болезни один человек контактирует в среднем с другими людьми с интенсивностью ContactRateInfection (**CRI**), равной 1.25 человека в день.
3. Если заразившийся человек контактирует с восприимчивыми к болезни, то вероятность передачи инфекции Infectivity (**Inf**) равна 0.6
4. После того, как человек заражается, инкубационный период AverageIncubationTime (**AIT**) длится 10 дней.
5. Сдерняя длитлеьность болезни после инкубационного периода AverageIllnessDuration (т.е. длительность периода, когда человек может заражать других) составляет 15 дней.

6. Выздоровившие люди получают иммунитет к болезни и заболеть снова не могут.

При построении математической модели были выделены четыре категории людей:

- Susceptible (**S**) – восприимчивые к заражению люди (которые еще не были заражены вирусом);

- Exposed **(Ex)**– люди, находящиеся в латентной фазе заражения (они уже заражены, но еще не могут заражать других);

- Infectious (**I**)– люди в активной фазе заражения (они могут заражать других людей);

- Recovered (**R**) - выздоровевшие люди (они приобрели иммунитет к данному заболеванию)

Начальные значения:
I(0)=1
S(0)=TP-1 (численность минус 1 инфицированный).

Поместить на форму бегунок и связать его с параметром CRI. Изменять значение данного параметра (от 0.3 до 2) и проанализировать специфику изменения графиков от CRI.

Провести эксперимент. Менять CRI от 0.3 до 2 с шагом 0.1. Проанализировать сценарий распространения инфекции в зависимости от значения CRI.

Вариант 2.1.

$$\frac{dS}{dt} = -\frac{I \cdot CRI \cdot Inf \cdot S}{TP}$$

$$\frac{dEx}{dt} = \frac{I \cdot CRI \cdot Inf \cdot S}{TP} - \frac{Ex}{AIT}$$

$$\frac{dI}{dt} = \frac{Ex}{AIT} - \frac{I}{AID}$$

$$\frac{dR}{dt} = \frac{I}{AID}$$

Вариант 2.2

$$\frac{dS(t)}{dt} = -aS(t)\frac{I(t)}{N}; \qquad (8.1)$$

$$\frac{dE(t)}{dt} = aS(t)\frac{I}{N(t)} - bE(t); \quad (8.2)$$

$$\frac{dI(t)}{dt} = bE(t) - cI(t); \quad (8.3)$$

$$\frac{dR(t)}{dt} = cI(t). \quad (8.4)$$

В формулах (1)-(4) используются следующие обозначения.

- α - коэффициент, который можно интерпретировать как вероятность получения болезни в случае контакта восприимчивого индивидуума с инфицированным:

$$a = \frac{1}{T_{inf}}, \quad (8.5)$$

где T_{inf} – период, в течение которого больной является заразным для окружающих.

- b- скорость перехода от латентной фазы в фазу инфицирования, который определяется формулой:

$$b = \frac{1}{T_{inc}}, \quad (8.6)$$

где T_{inc} –среднее время инкубационного периода инфицированного;

- c – скорость выздоровления:

$$c = \frac{1}{T_{ill}}, \quad (8.7)$$

где T_{ill} –среднее время течения болезни.

Таблица 8.1 Значения параметров модели

T_{ill}	T_{inc}	T_{inf}
12 (дней)	5.10 (дней)	2.79 (дней)

Поместить на форму бегунок и связать его с параметром T_{inf} . Изменять значение данного параметра (от 2 до 10) и проанализировать специфику изменения графиков от CRI.

Провести эксперимент. Менять T_{inf} от 2 до 10 с шагом 0.25. Проанализировать сценарий распространения инфекции в зависимости от значения T_{inf}.

Вариант 2.3

$$\frac{dS(t)}{dt} = -aS(t)\frac{I(t)}{N}; \quad (8.8)$$

$$\frac{dE(t)}{dt} = aS(t)\frac{I}{N(t)} - bE(t); \quad (8.9)$$

$$\frac{dI(t)}{dt} = bE(t) - (c+\mu)I(t); \quad (8.10)$$

$$\frac{dR(t)}{dt} = cI(t); \quad (8.11)$$

$$\frac{dD(t)}{dt} = \mu I(t); \quad (8.12)$$

В формулах (1)-(4) используются следующие обозначения.

- α - коэффициент, который можно интерпретировать как вероятность получения болезни в случае контакта восприимчивого индивидуума с инфицированным:

$$a = \frac{1}{T_{inf}}, \quad (8.13)$$

где T_{inf} – период, в течение которого больной является заразным для окружающих.

- *b*- скорость перехода от латентной фазы в фазу инфицирования, который определяется формулой:

$$b = \frac{1}{T_{inc}}, \quad (8.14)$$

где T_{inc} –среднее время инкубационного периода инфицированного;

- *c* – скорость выздоровления:

$$c = \frac{1}{T_{ill}}, \quad (8.15)$$

где T_{ill} –среднее время течения болезни.

μ - коэффициент смертности.

Таблица 8.2 Значения параметров модели

T_{ill}	T_{inc}	T_{inf}	μ
12 (дней)	5.10 (дней)	2.79 (дней)	0.33

Поместить на форму бегунок и связать его с параметром T_{inf} . Изменять значение данного параметра (от 2 до 10) и проанализировать специфику изменения графиков от CRI.

Провести эксперимент. Менять T_{inf} от 2 до 10 с шагом 0.25. Проанализировать сценарий распространения инфекции в зависимости от значения T_{inf}.

Вариант 2.4.

$$\frac{dS}{dt} = -\frac{R_0 S(t) I(t)}{T_{inf}}; \quad (8.16)$$

$$\frac{dE(t)}{dt} = \frac{R(t)}{T_{inf}} I(t) S(t) - T_{inc}^{-1} E(t); \quad (8.17)$$

$$\frac{dI(t)}{dt} = \frac{E(t)}{T_{inc}} - \frac{I(t)}{T_{inf}}; \quad (8.18)$$

$$\frac{dH(t)}{dt} = (1 - p_a)\frac{I(t)}{T_{inf}} + \left(1 - p_f\right)\frac{C(t)}{T_{crt}} - \frac{H(t)}{T_{hsp}}; \quad (8.19)$$

$$\frac{dC(t)}{dt} = \frac{p_c H(t)}{T_{hsp}} - \frac{C(t)}{T_{crt}}; \quad (8.20)$$

$$\frac{dR(t)}{dt} = \frac{p_a(t) I(t)}{T_{inf}} + (1 - p_c)\frac{H(t)}{T_{hsp}}; \quad (8.21)$$

$$\frac{dD(t)}{dt} = \frac{p_f C(t)}{T_{crt}}. \quad (8.22)$$

Таблица 8.3 Значения параметров модели SEIR-HCD

R_0^b	R_0^q	T_{inc}	T_{inf}	T_{hsp}	T_{crt}	p_a	p_c	p_f
3.56	0.74	5.10 (дней)	2.79 (дней)	5.14 (дней)	14.06 (дней	0.79	0.12	0.33

Поместить на форму бегунок и связать его с параметром T_{inf} . Изменять значение данного параметра (от 2 до 10) и проанализировать специфику изменения графиков от CRI.

Провести эксперимент. Менять T_{inf} от 2 до 10 с шагом 0.25. Проанализировать сценарий распространения инфекции в зависимости от значения T_{inf}.

ЗАКЛЮЧЕНИЕ

Данный лабораторный практикум предназначен для выполнения лабораторных работ по курсу «Моделирование вычислительных систем». Данные лабораторные работы охватывают весь спектр особенностей, которые позволяет учесть среда AnyLogic. К ним относится агентное моделирование, системная динамика, проведение простых экспериментов. Большое внимание уделено визуализации моделируемых систем.

Следует отметить, что ряд вопросов, связанных со спецификой исследования моделей с помощью аналитического аппарата, не вошли в данный практикум в связи с ограниченным числом лабораторных работ в данном курсе. Так, в работе почти отсутствует материал, посвященный аналитическому аппарату, предназначенному для определения числовых характеристик . Это обусловлено направленностью лабораторных работ на численное решение задач. Аналитические подходы к нахождению характеристик исследуемых систем полной мере отражены в источниках [5, 7, 8].

Практикум может быть полезен всем лицам, заинтересованным в изучении основ математического и имитационного моделирования, а также освоения среды AnyLogic как современного средства их построения.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Акопов, А.С. Имитационное моделирование. учебник и практикум для академического бакалавриата / А.С. Акопов. - Люберцы: Юрайт, 2016. - 389 с.
2. Боев, Д. В. Компьютерное моделирование: Пособие для практических занятий, курсового и дипломного проектирования в AnyLogic / Д.В, Боев - СПб: ВАС, 2014 – 432 с.
3. Голубева, Н.В. Математическое моделирование систем и процессов: Учебное пособие/ Н.В. Голубева. – СПб.: Лань, 2013. – 192 с.
4. Горлач, Б. А. Математическое моделирование. Построение моделей и численная реализация / Б.А. Горлач, В.Г. Шахов. - М.: Лань, 2016. - 292 с.
5. Григорьев И. AnyLogic за три дня [Электронный ресурс] // AnyLogic.ru. Режим доступа: https://www.anylogic.ru/resources/books/free-simulation-book-and-modeling-tutorials/
6. Девятков, В.В. Имитационное моделирование: Учебное пособие / Н.Б. Кобелев, В.А. Половников, В.В. Девятков. - М.: КУРС, НИЦ ИНФРА-М, 2013. - 368 с.
7. Звонарев, С.В. Основы математического моделирования: учебное пособие / С.В. Звонарев. — Екатеринбург : Изд-во Урал. ун-та, 2019. — 112 с.
8. Зубарев, Ю.М. Математическое моделирование многоагентных систем конкуренции и кооперации (Теория игр для всех): Учебное пособие/ Ю.М. Зубарев, С.В. Косаревский. – СПб.: Лань П, 2016. – 624 с.
9. Каталевский, Д.Ю. Основы имитационного моделирования и системного анализа в управлении: учебное пособие; 2-е изд., перераб. и доп. / Д.Ю. Каталевский. — М.: Издательский дом «Дело» РАНХиГС, 2015. — 496 с., ил.
10. Карпов, Ю.Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5 / Ю.Г.Карпов. – СПб.: БХВ-Петербург, 57 2005.
11. Олейникова, С.А. Математическое моделирование и системы массового обслуживания: Учебное пособие/ С.А. Олейникова. – Воронеж, ФГБОУ ВО «Воронежский государственный технический университет», 2021. – 91 с.
12. Плотников, С.А. Математическое моделирование систем управления: учебное пособие/ С.А. Плотников, Д.М.Семенов, А.Л.Фрадков. - Санкт-Петербург: Университет ИТМО, 2021. - 193 с.
13. Рейзлин, В.И. Математияческое моделирование: Учебное пособие для магистратуры/ В.И. Рейзлин. – Люберцы, 2016. – 126 с.

14. Самарский, А.А. Компьютеры, модели, вычислитель- ный эксперимент / А.А. Самарский.— Москва : Наука, 1988.— 354 с.

15. Советов, Б.Я. Моделирование систем: Учеб. Для ВУЗов – 3-е изд., перераб. и доп. / Б.Я. Советов, С.А. Яковлев. – М.: Высш. шк., 2001. – 343 с.

16. Строгалев, В.П. Имитационное моделирование : учеб. пособие для вузов / В.П. Строгалев, И.О. Толкачева.— Москва : Изд-во МГТУ им. Н.Э. Баумана, 2008.— 276 с.

17. AnyLogic [Электронный ресурс]. Режим доступа: http://www.anylogic.ru/

ОГЛАВЛЕНИЕ

Учебное издание

ОЛЕЙНИКОВА Светлана Александровна

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Практикум

Издание публикуется в авторской редакции

Дизайн обложки С.А. Кравец

Подписано в печать 21.09.2022. Формат 60х84 1/16
Усл. печ. л. 5,0. Заказ 000. Тираж 500 экз.

ООО Издательство «Научная книга»
394077, Россия, г. Воронеж, ул. 60-й Армии, 25-120
http://www.sbook.ru/

Отпечатано с готового оригинал-макета
в ООО «Цифровая полиграфия»
394036, Россия, г. Воронеж, ул. Куколкина, 6
Тел. (473) 261-03-61