Уральский федеральный университет
имени первого Президента России Б. Н. Ельцина
Институт фундаментального образования

О. В. ЛИМАНОВСКАЯ

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ В ANYLOGIC 7

Часть 2

Лабораторный практикум

Министерство образования и науки Российской Федерации
Уральский федеральный университет
имени первого Президента России Б. Н. Ельцина

О. В. Лимановская

Имитационное моделирование в AnyLogic 7

В 2 частях

Часть 2

Рекомендовано методическим советом
Уральского федерального университета
в качестве лабораторного практикума для студентов вуза,
обучающихся по направлению
09.03.04 — Программная инженерия

Екатеринбург
Издательство Уральского университета
2017

УДК 004.94(076.5)
ББК 32.972.1я
Л58

Рецензенты:
ст. науч. сотр., канд. физ.-мат. наук О. Р. Рахманова (лаборатория электродных процессов ФГБУН «Институт высокотемпературной электрохимии УрО РАН);
Г. В. Лимановский (начальник группы поддержки сети Института высокотемпературной электрохимии УрО РАН)

Научный редактор доц., канд. техн. наук И. Н. Обабков

Лимановская, О. В.
Л58 Имитационное моделирование в AnyLogic 7. В 2 ч., ч. 2 : лабораторный практикум / О. В. Лимановская. — Екатеринбург : Изд-во Урал. ун-та, 2017. — 104 с.

ISBN 978-5-7996-1996-1 (ч. 2)
ISBN 978-5-7996-1995-4

Работа состоит из двух частей: часть 1 — учебное пособие, часть 2 — лабораторный практикум. В первой части излагаются основы имитационного моделирования в среде AnyLogic 7. Во второй части приведены четыре лабораторных работы для бакалавров, изучающих модуль «Математическое и компьютерное моделирование». Издание предназначено для студентов всех форм обучения и аспирантов, обучающихся по техническим специальностям.

Рис. 160.

УДК 004.94(076.5)
ББК 32.972.1я

ISBN 978-5-7996-1996-1 (ч. 2)
ISBN 978-5-7996-1995-4

Введение

Лабораторный практикум посвящен изучению работы в среде имитационного моделирования AnyLogic 7, содержит подробные пошаговые инструкции по созданию моделей в среде AnyLogic 7. Рассмотрено создание дискретно-событийных, агентных и пешеходных моделей. В ходе работы по созданию моделей могут возникать вопросы по теоретической части среды AnyLogic. Теоретические вопросы и справочные материалы по среде AnyLogic 7 изложены в учебном пособии «Имитационное моделирование в AnyLogic 7» (часть 1). Поэтому для работы со студентами в ходе выполнения ими лабораторных работ рекомендуется использование обоих изданий.

Лабораторная работа № 1
Дискретно-событийное моделирование. Моделирование систем массового обслуживания

Задача

Промоделировать работу билетных касс. В кассы есть единая очередь, которую обслуживают две основные кассы. Если основные кассы не справляются с потоком покупателей, то открывается третья касса. Поток покупателей меняется в зависимости от времени суток и становится больше в выходные дни. Расписание потока покупателей приведено ниже.

Рабочие дни:
8:00–13:00 — десять человек в час;
13:00–16:00 — пятнадцать человек в час;
16:00–22:00 — двадцать человек в час.
Выходные дни:
9:00–12:00 — двадцать человек в час;
12:00–21:00 — сорок человек в час.

Покупатели, время ожидания покупки у которых превысило час, уходят из касс, не купив билета. Время обслуживания одного покупателя в кассах меняется случайным образом от 2 до 15 минут и в среднем составляет 5 минут. Предусмотреть в модели учет купивших и некупивших билеты.

Решение

Этап 1. Задание логики работы модели

Шаг 1. Создание новой модели

Создайте новую модель (рис. 1.1) в среде AnyLogic, нажав на кнопку Создать и выбрав Модель из выпадающего меню (или используйте клавиши быстрого доступа <Ctrl+N>).

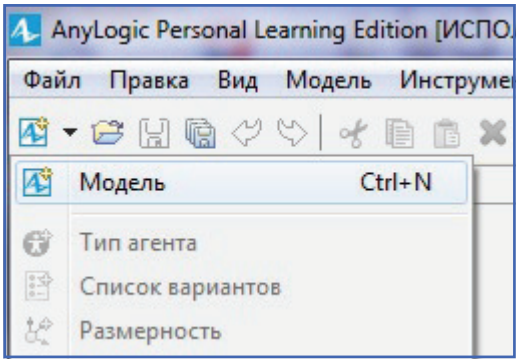

Рис. 1.1. Создание новой модели

Введите имя модели ticket в открывшемся диалоговом окне и задайте единицы модельного времени — минуты (рис. 1.2). Нажмите кнопку Готово.

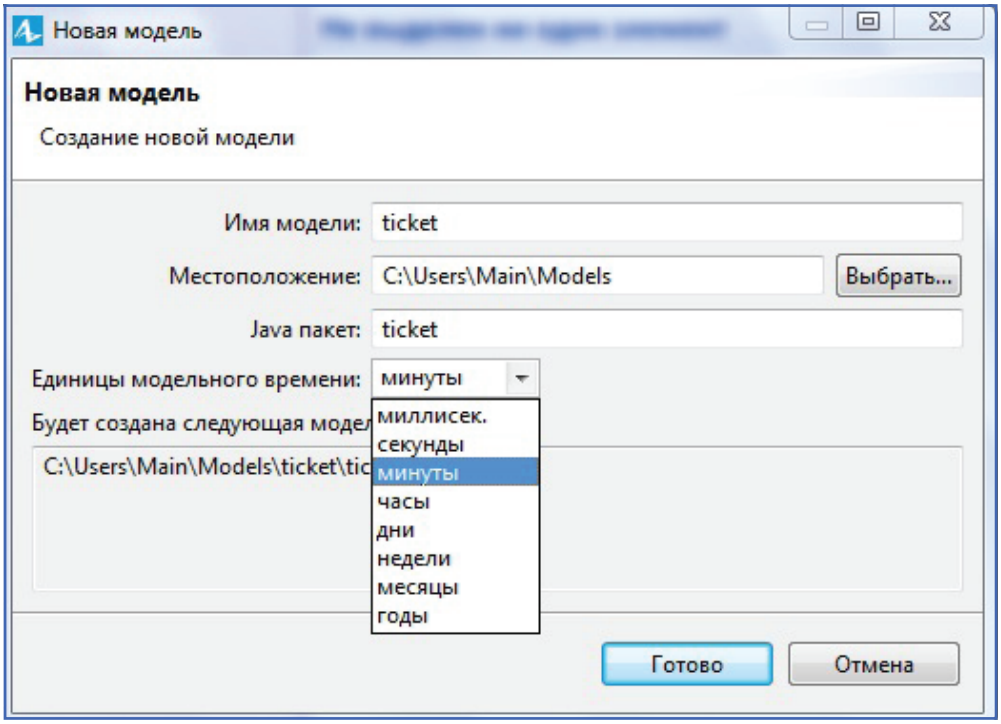

Рис. 1.2. Задание параметров новой модели

Шаг 2. Моделирование прихода покупателей

В открывшемся окне модели перейдите на вкладку Палитра и откройте первую библиотеку из списка — Библиотеку моделирования процессов. Из нее перетащите на рабочее поле блок Source (рис. 1.3). Как известно, именно этот блок моделирует появление заявок в модели. В нашем случае покупатели — это заявки.

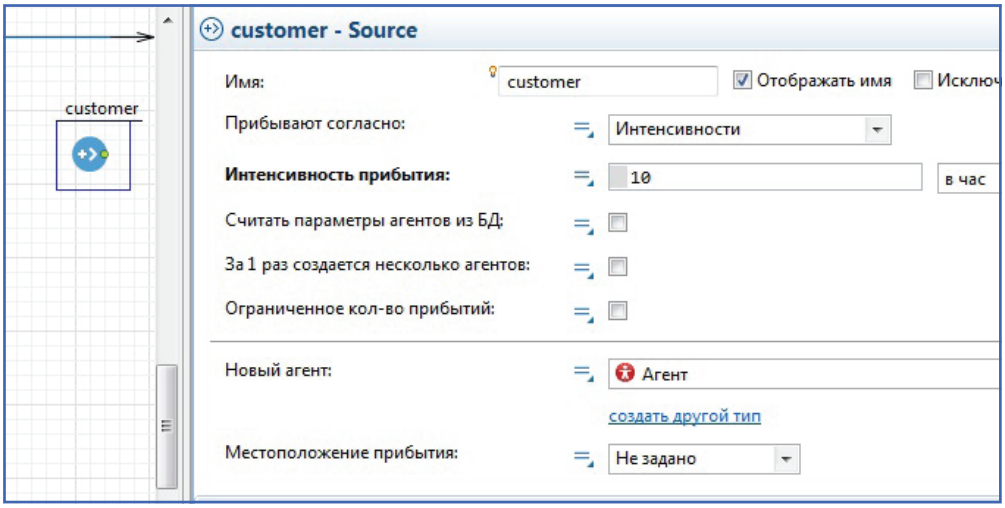

Рис. 1.3. Моделирование прихода покупателей

В начале моделирования мы не будем учитывать изменение интенсивности потока покупателей в течение дня и недели. Просто зададим наименьшую границу потока. Для этого выберем в пункте Прибывают согласно режим Интенсивности и зададим значение интенсивности 10. Это значит, что в среднем 10 раз в час будут приходить покупатели, то есть примерно каждые 6 минут.

Шаг 3. Моделирование очереди покупателей

Перетащите из библиотеки моделирования процессов блок Queue. Будьте внимательны и постарайтесь сделать так, чтобы блок Queue связался с блоком Source. Это позволит избежать лишних хлопот по дорисовыванию связей между блоками. Если не удастся связать блоки, то нужно дважды щелкнуть на выходной порт блока Source и потянуть связь до блока Queue. Заявка проходит в модели только по связям. Если

блоки не связаны, то заявка не сможет перейти в следующий блок. Блок Queue имитирует накопление заявок в модели, фактически моделирует очередь. Оставим все ее параметры по умолчанию. Это значит, что ее логика будет соответствовать логике очереди «первый пришел, первый ушел» и не будет задано никакой привязки на местности. Отметить пункт Максимальная вместимость (рис. 1.4), поскольку в задании ничего не было сказано об ограничениях длины очереди.

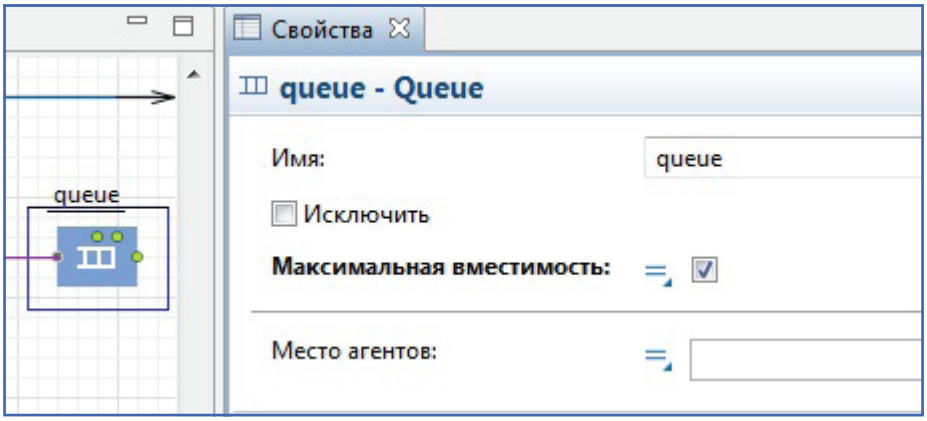

Рис. 1.4. Моделирование очереди покупателей

Шаг 4. Моделирование процесса покупки билетов

Для процесса покупки билетов требуется ресурс — кассиры. Для моделирования ресурсов в модели используется блок ResourcePool. Перетащите его на рабочее поле модели. Этот блок не связывается ни с каким блоком в модели, так как через него не должны проходить заявки.

В свойствах блока задайте его имя booking_clerk. Поскольку кассиры могут передвигаться от кассы к кассе, то задайте тип ресурса как Движущийся и укажите их скорость 2 км/ч. Пока не будем задавать расписание работы кассиров и в пункте Количество задано оставим режим Напрямую и укажем количество кассиров, например 2 (рис. 1.5).

Перед тем как моделировать сам процесс покупки билетов, используем блок выбора движения заявок, поскольку покупатели из очереди могут идти как в первую, так и во вторую кассу. Перетащите блок selectOutput на рабочее поле и соедините его с блоком Queue. Условие выбора касс введем позже.

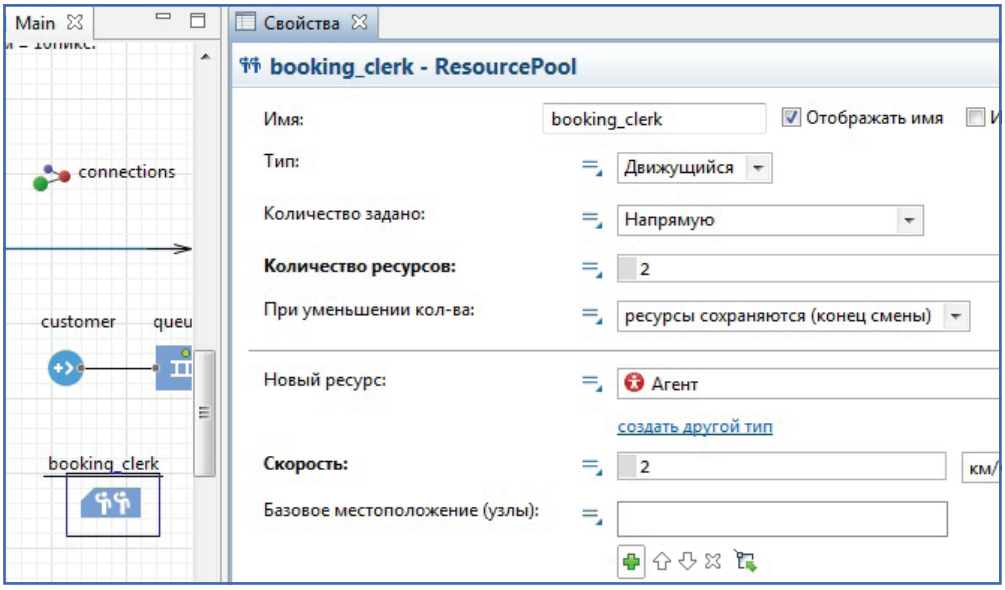

Рис. 1.5. Моделирование ресурсов для продажи билетов

Далее нужно промоделировать сам процесс покупки билетов. Процесс покупки билетов — это процесс обслуживания заявки. Для этого используется блок Service, который включает в себя логику работы трех блоков Sieze (захват ресурсов), Delay (удержание заявки и ресурсов), Release (освобождение ресурсов). Также блок Service имеет собственную очередь, то есть в него еще встроен блок Queue. Поэтому если бы к каждой кассе шла своя очередь, то блок Queue отдельно можно было бы не использовать.

Перетащите два блока Service на рабочее поле. В свойствах блоков задайте их имена booking1 и booking2. Задайте вместимость собственных очередей 2 и время задержки (рис. 1.6). Время задержки задается функцией треугольного распределения со средним значением 5 минут, минимальным значением 2 минуты и максимальным значением 15 минут. В пункте Набор(ы) ресурсов выберите из списка ресурсов только что созданный ресурс booking_clerk. Теперь в первой и во второй кассах будут работать указанные в ресурсах 2 кассира.

Теперь, когда обе кассы промоделированы, вернемся к условию выбора кассы покупателем. Пусть покупатель идет во вторую кассу, увидев, что в первой кассе собралось более трех человек. Такое поведение заявкам можно задать, если использовать в условии выбора функцию

объекта service — size (), которая возвращает количество клиентов, обслуживаемых в данный момент объектом. Если величина, возвращенная функцией size () объекта booking1, будет больше 3 (рис. 1.7), то значит, что в эту кассу покупатель не пойдет.

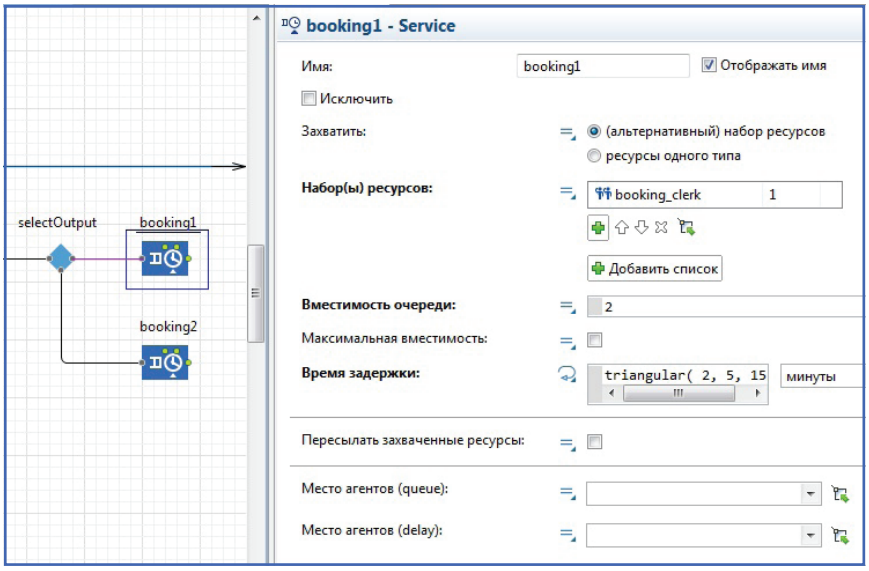

Рис. 1.6. Моделирование процесса продажи билетов

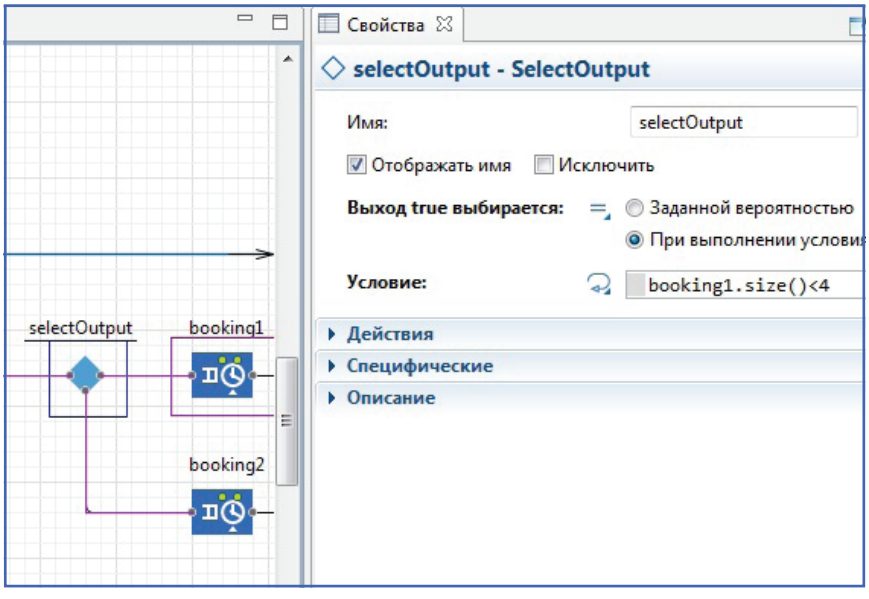

Рис. 1.7. Условие выбора касс

Шаг 5. Моделирование ухода покупателей из касс

Покупатели, купив билет, уходят из касс. Для моделирования этого события используется объект Sink (рис. 1.8). Перетащите объект Sink на рабочее поле и соедините его с выходами обеих касс. В свойствах объекта задайте его имя exit.

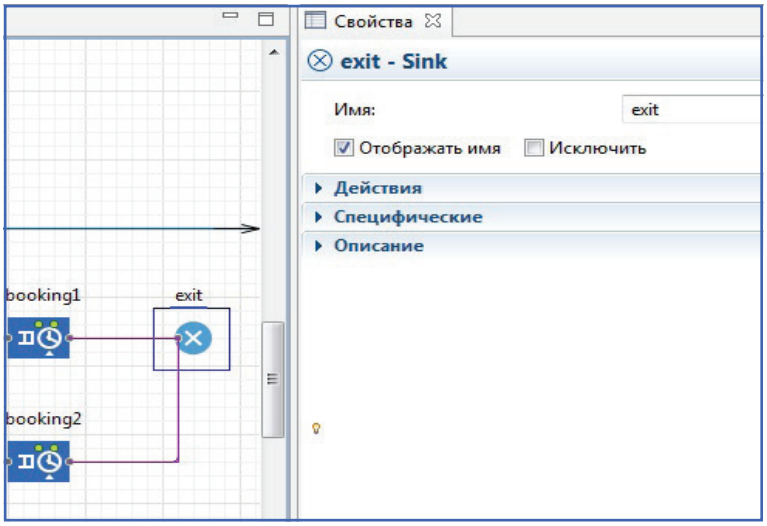

Рис. 1.8. Моделирование ухода покупателей из касс

Шаг 6. Проверка работоспособности модели

Набросок модели закончен. Он должен выглядеть, как на рис. 1.9.

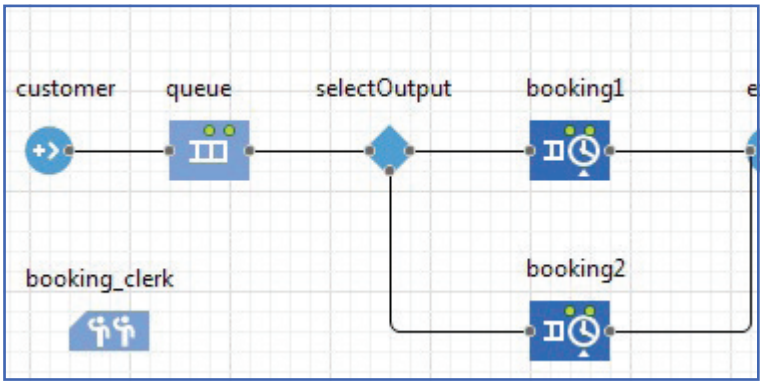

Рис. 1.9. Черновик модели

Запустите модель. Результат работы модели спустя некоторое модельное время должен выглядеть, как на рис. 1.10.

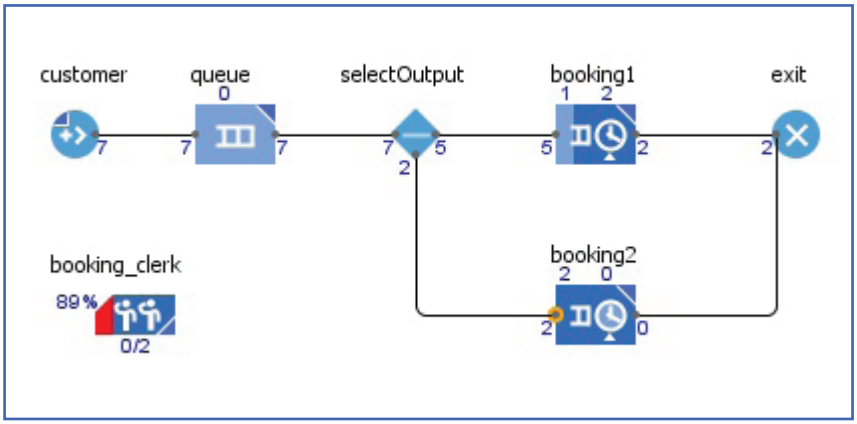

Рис. 1.10. Работа модели

Этап 2. Задание расписаний

Шаг 1. Задание расписания прихода покупателей

В задании было сказано, что интенсивность прихода покупателей меняется в течение дня и становится больше в выходные. Любые периодические изменения в модели удобно задавать объектом Расписание. Перетащите объект Расписание из библиотеки на рабочее поле модели. Этот объект не нужно объединять ни с каким объектом.

В свойствах объекта задайте его имя schedule_customer. Поскольку будет задаваться расписанием Изменение интенсивности прибытия покупателей, то выбираем тип расписания — Расписание интенсивности. Тип расписания задается в разделе Данные. Значение по умолчанию оставляем 0, это будет означать, что в часы, не указанные в расписании, покупатели не приходят. Расписание будет задавать интервалы, в которых интенсивность остается постоянной. Задаем расписание на неделю, поэтому длительность расписания выбираем Неделя. Само расписание задается в таблице в пункте Повторять расписание еженедельно. Справа от таблицы есть кнопки для добавления (+), удаления (x) и прокрутки расписания. Нажмите на кнопку «+» для добавления новой строки в таблице. Каждая строка в таблице расписания

задает один временной интервал, начало и конец которого задаются в соответствующих столбцах таблицы. В столбце Значение задайте значение интенсивности прибытия покупателей. В результате должно получиться расписание, как на рис. 1.11.

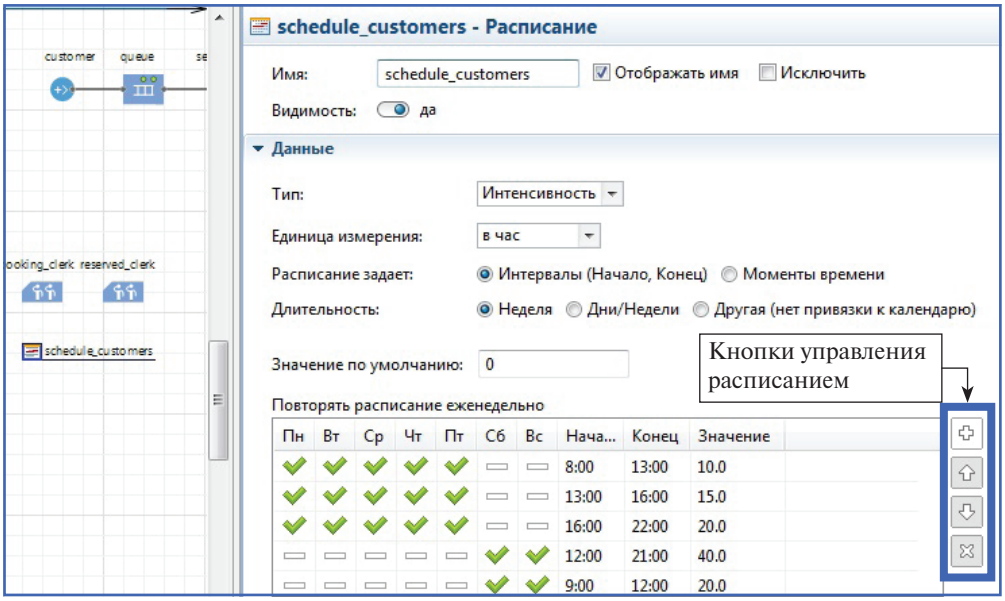

Рис. 1.11. Расписание интенсивности прибытия покупателей

Шаг 2. Задание расписания работы кассиров

Расписание работы кассиров будет задавать время работы кассиров и количество работающих кассиров (рис. 1.12). Поскольку число кассиров — это целое число, то в пункте Тип выбираем целое. Значение по умолчанию задаем 0, значение в течение рабочих часов — 3.

Шаг 3. Задание расписания перерывов в работе кассиров

В работе кассиров предполагаются 15-минутные перерывы, в ходе которых кассиры как ресурсы не доступны. Следовательно, для задания расписания перерывов нужно создать расписание доступности, для которого выбираем Тип: да/нет (рис. 1.13). По умолчанию делаем ресурс доступным, то есть значение да. В столбце перерывов указываем значение нет. Также в свойствах расписания задаем его имя schedule_timeout.

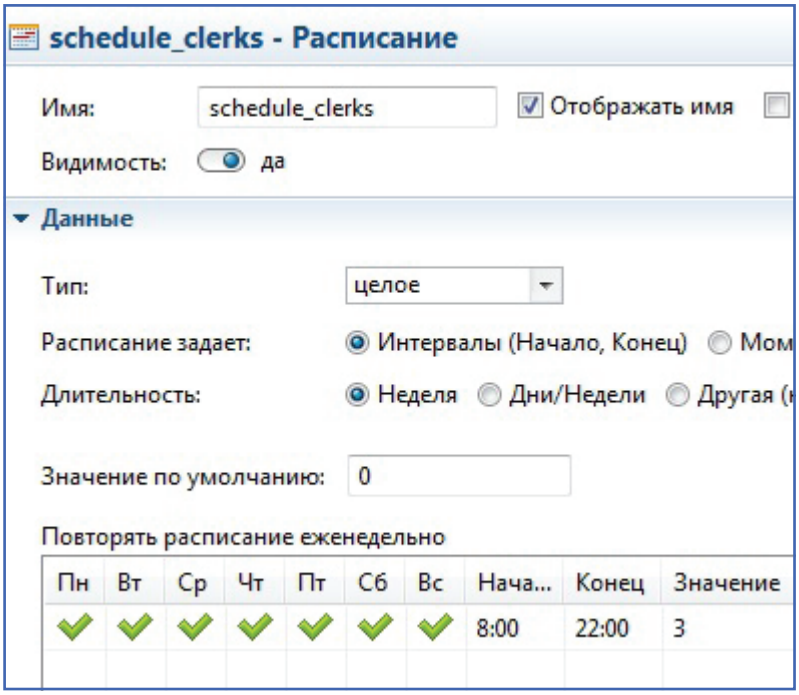

Рис. 1.12. Расписание работы кассиров

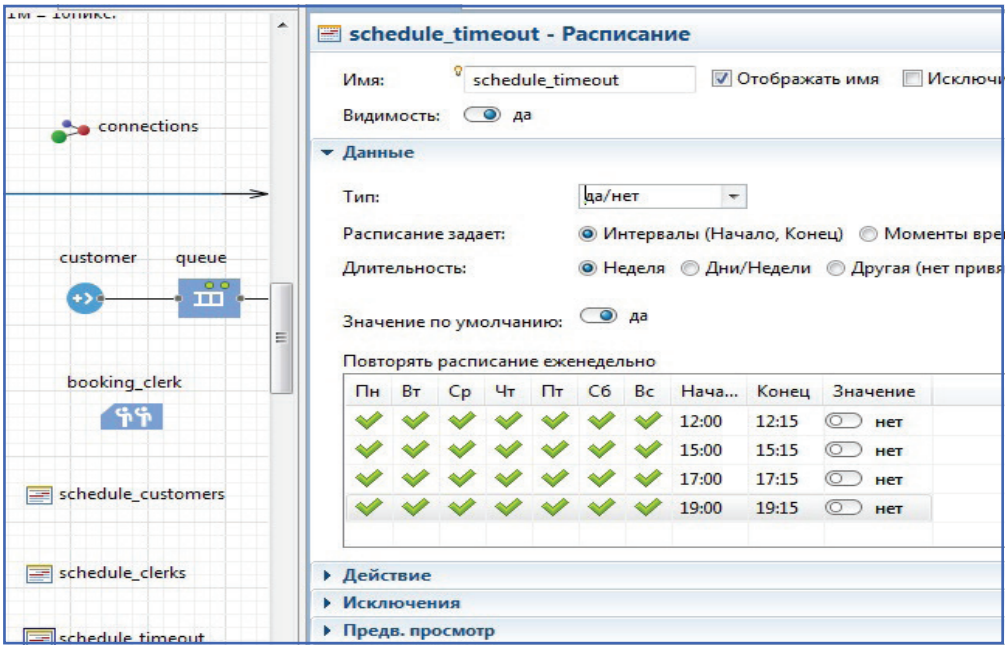

Рис. 1.13. Расписание перерывов работы кассиров

Шаг 4. Привязка расписания прихода покупателей к объекту source

Вначале мы создавали объект Source и указывали, что заявки прибывают согласно интенсивности, а затем задавали интенсивность прибытия. Теперь пришла пора исправить ситуацию. В свойствах объекта source в пункте Прибывают согласно выберите вариант Расписанию интенсивностей (рис. 1.14). Появится новый пункт в свойствах Расписание интенсивностей. Выберите в этом пункте созданное ранее расписание прибытия покупателей.

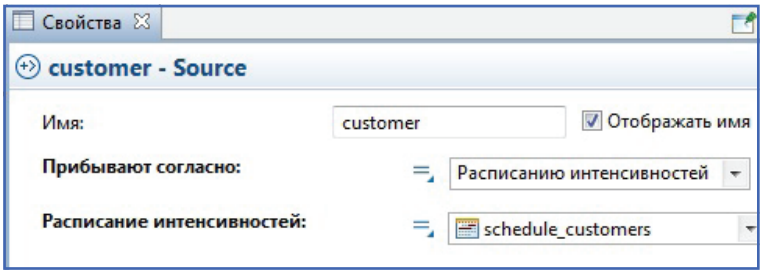

Рис. 1.14. Привязка расписания прибытия покупателей

Теперь покупатели будут приходить с разной интенсивностью в течение дня.

Шаг 5. Привязка расписания работы кассиров к объекту ResourcePool

Ранее нами количество ресурсов в объекте booking_clerk было задано напрямую. Теперь зададим расписание их работы (рис. 1.15). В пункте Количество задано выберите вариант Расписанием и в появившемся пункте Расписание выберите ранее созданное расписание работы кассиров.

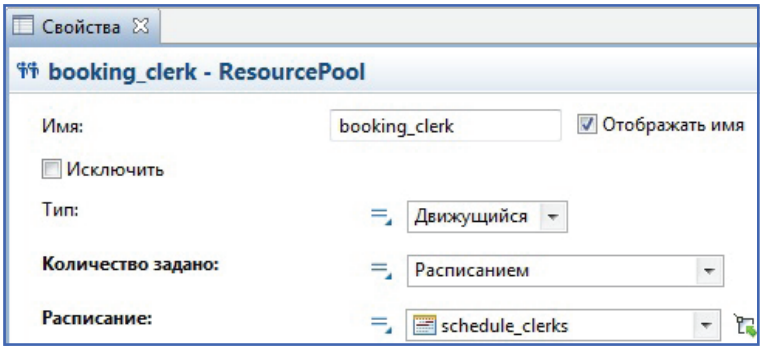

Рис. 1.15. Привязка расписания работы кассиров

Шаг 6. Привязка расписания перерывов в работе кассиров

Откройте раздел Смены, перерывы, аварии, обслуживание… в свойствах объекта booking_clerk (рис. 1.16). Поставьте галочку в пункте Перерывы. Появится окно, в котором нужно задать расписание перерывов. Выберите ранее созданное расписание перерывов работы кассиров в пункте Расписание перерывов.

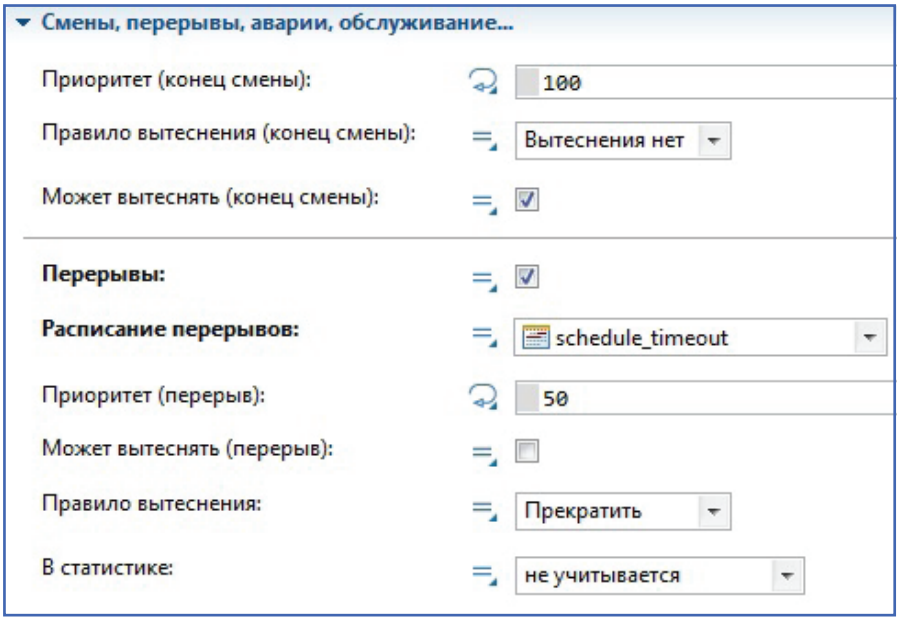

Рис. 1.16. Привязка расписания перерывов в работе кассиров

Этап 3. Создание резервной кассы и ухода клиентов, чье ожидание покупки превысило час

Шаг 1. Моделирование резервной кассы

Для моделирования резервной кассы используется блок service (рис. 1.17) и все свойства задаются такие же, как и у первых двух касс. Только время обработки задается чуть меньше, чем в первых двух кассах.

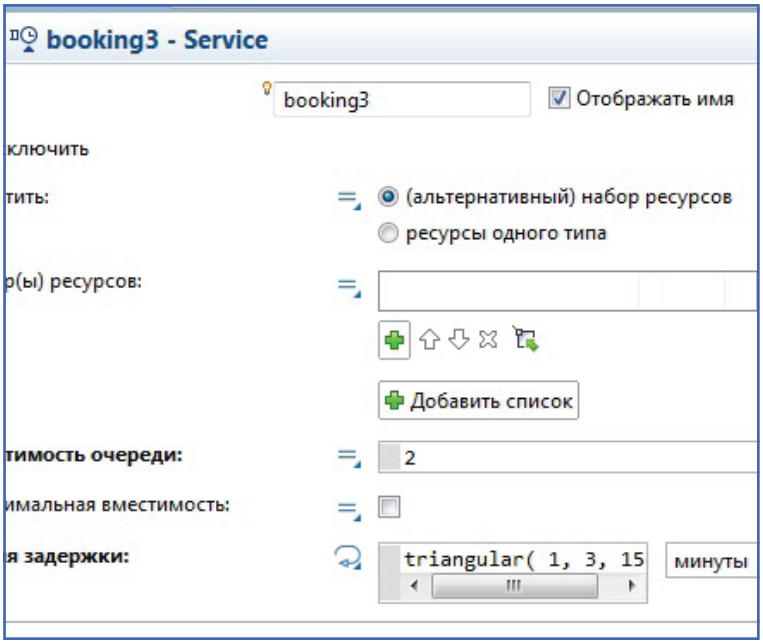

Рис. 1.17. Моделирование резервной кассы

Шаг 2. Задание ресурсов для резервной кассы

В резервной кассе также будут работать кассиры, но из другой бригады, поэтому нужно создать новый ресурс. Назовем его reserved_clerk (рис. 1.18). Расписание работы кассиров резерва такое же, как и у основной бригады.

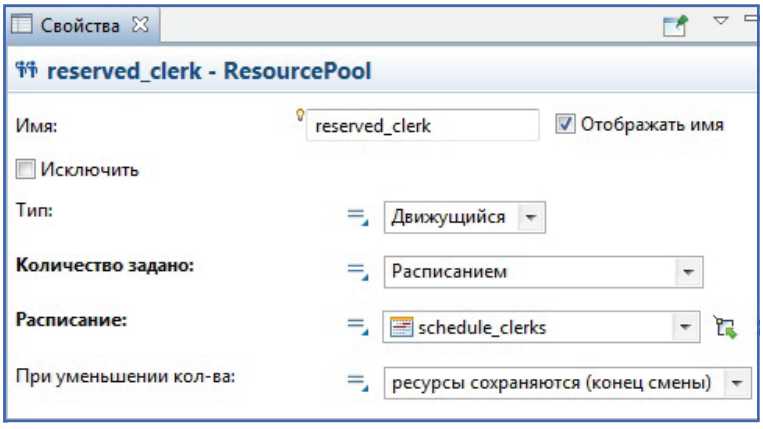

Рис. 1.18. Моделирование резервной бригады кассиров

Шаг 3. Привязка ресурсов к резервной кассе

Для того чтобы привязать резервную бригаду к резервной кассе, укажите в пункте Набор(ы) ресурсов только что созданный ресурс reserved_clerk (рис. 1.19). Выход кассы соедините с объектом exit.

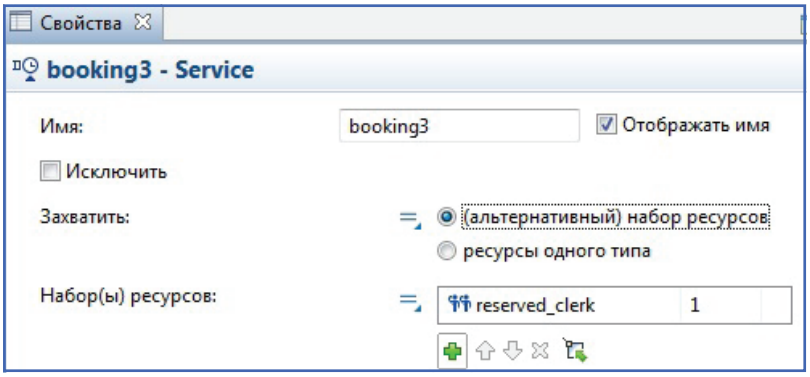

Рис. 1.19. Привязка ресурсов к резервной кассе

Шаг 4. Организация ограничения времени ожидания в кассах

В условии задачи было сказано, что клиенты, чье ожидание покупки билета превысило час, уходят из касс, не купив билеты. Для реализации этого нужно задать тайм-аут, после которого заявка уходит из объекта service через выход OutTimeout. Время тайм-аута задается в разделе Специфические (рис. 1.20). Задайте тайм-ауты в первой и второй кассе, равные 30 минутам, в третьей кассе — 20 минут.

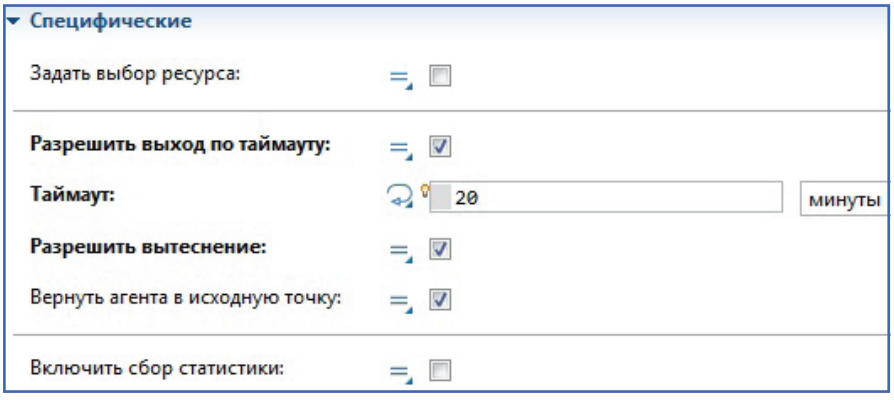

Рис. 1.20. Задание тайм-аута

Шаг 5. Организация выхода заявок из первой и второй кассы, чье время ожидание превысило 30 минут

Соедините выходы OutTimeout объектов booking1 и booking2 со входом в объект booking3. Таким образом, покупатели, чье время ожидания в первой и второй кассах превысило 30 минут, идут в резервную кассу.

В итоге модель должна получиться, как показано на рис. 1.21.

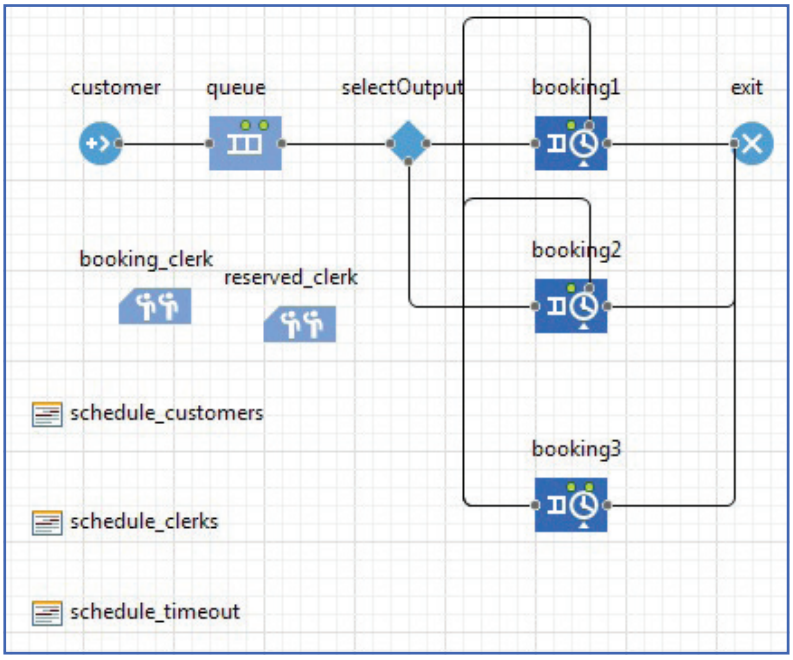

Рис. 1.21. Модель с учетом резервной кассы

Шаг 6. Проверка работоспособности модели

Запустите модель. Спустя некоторое модельное время должно получиться, как показано на рис. 1.22.

Шаг 7. Добавление объекта Часы

Для удобства просмотра работы модели нужно поместить на рабочее поле объект Часы, которые будут показывать текущее модельное время (рис. 1.23). Этот объект находится на вкладке Картинки в палитре инструментов. Перетащите его на рабочее поле.

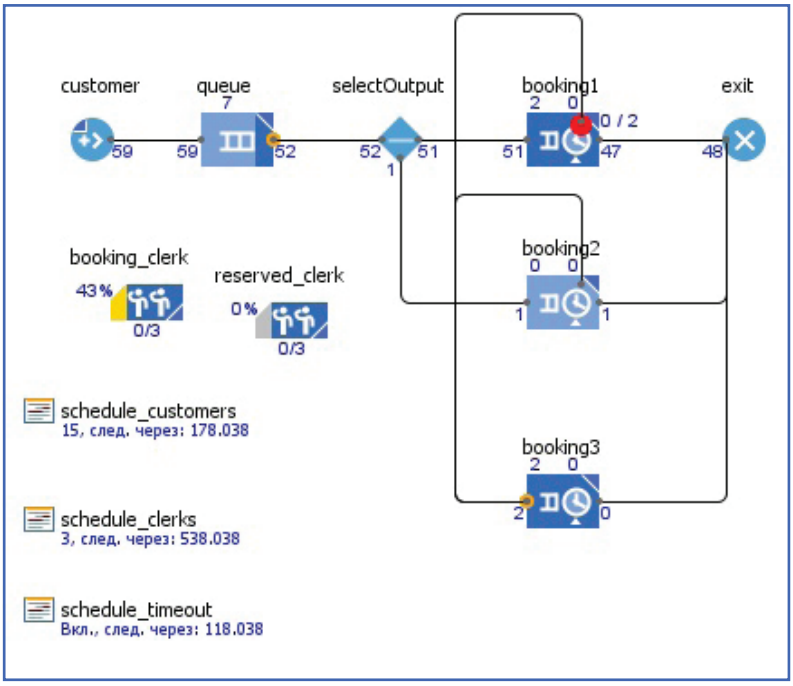

Рис. 1.22. Работа модели с учетом резервной кассы

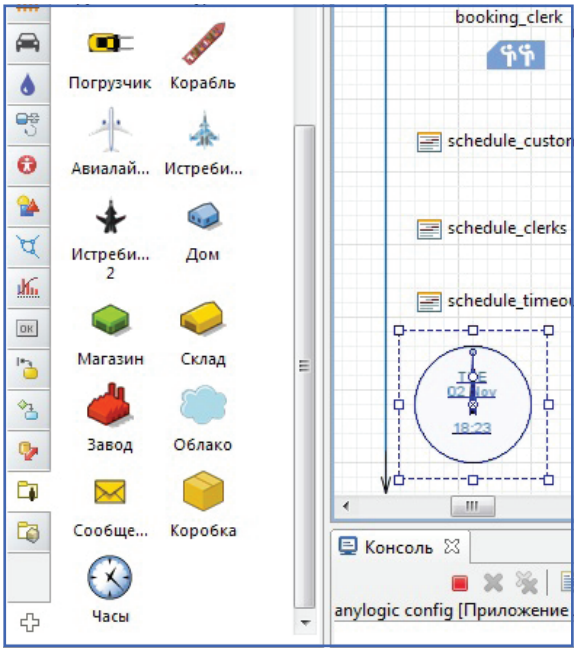

Рис. 1.23. Объект Часы

Шаг 8. Организация ухода покупателей, не купивших билет

Уход покупателей, не купивших билет, моделируется так же, как и уход покупателей, купивших билет (рис. 1.24).

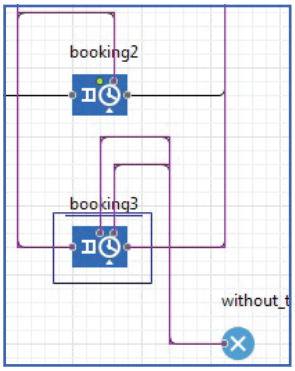

Рис. 1.24. Уход покупателей, не купивших билет

Шаг 9. Проверка работоспособности модели

Запустите модель. Должно получиться, как на рис. 1.25.

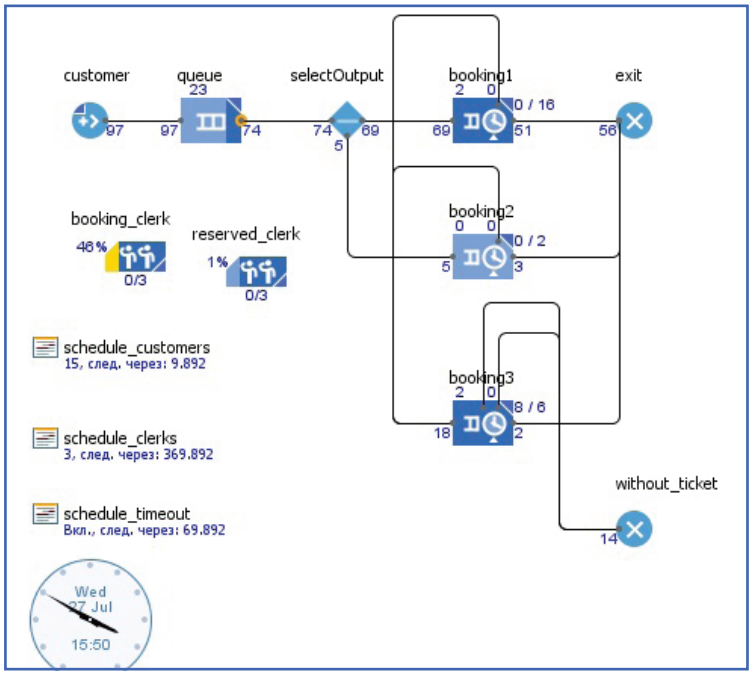

Рис. 1.25. Работа модели с учетом ухода покупателей, не купивших билет

Этап 4. Добавление динамических графиков в модель

Шаг 1. Создание набора данных по проданным билетам

Будем полагать, что каждый покупатель покупал 1 билет. Конечно, это немного упрощенный подход, но мы еще не волшебники, а только учимся. Для сбора данных по количеству проданных билетов во время работы модели используется объект Набор данных (рис. 1.26). Этот объект представляет собой двумерный массив, хранящий значения X и Y. Этот объект находится в библиотеке Статистика на палитре инструментов. Перетащите этот объект на рабочее поле. Перейдите в его свойства. Задайте имя набора — sold_tickets. Поскольку нас интересует динамика продаж билетов, то по оси X зададим модельное время. По оси Y будем сохранять количество покупателей, прошедших через выход exit. Эту величину возвращает функция size () объекта Sink. Данные в наборе будут обновляться каждые три минуты.

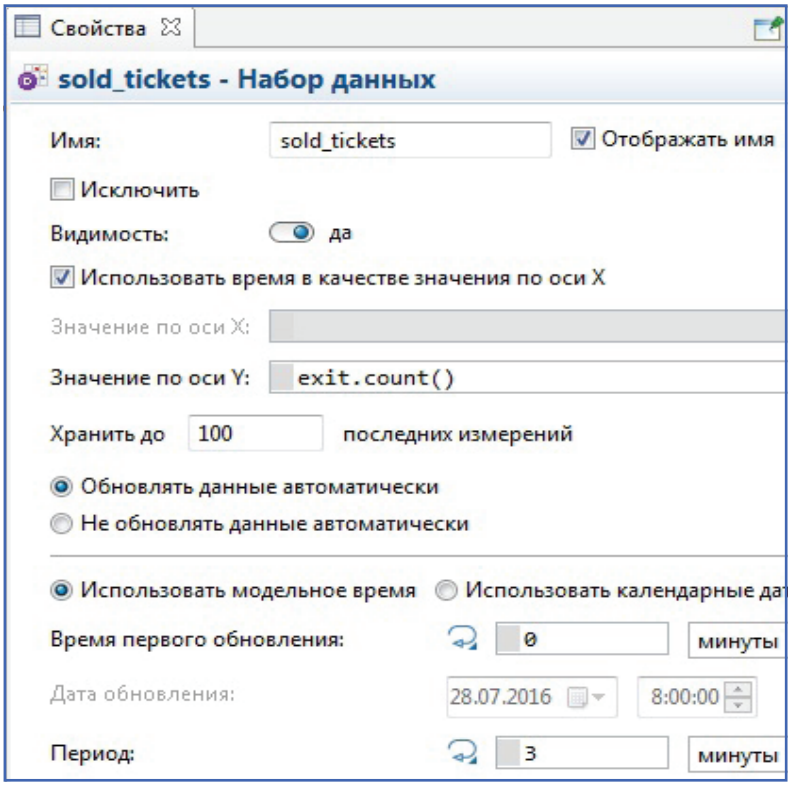

Рис. 1.26. Набор данных покупателей, купивших билет

Шаг 2. Создание набора данных по непроданным билетам

Все ушедшие покупатели могли купить минимум один билет. Будем считать минимальные потери. Для их учета создадим также набор данных lost_tickets по той же технологии, как и в шаге 1 (рис. 1.27).

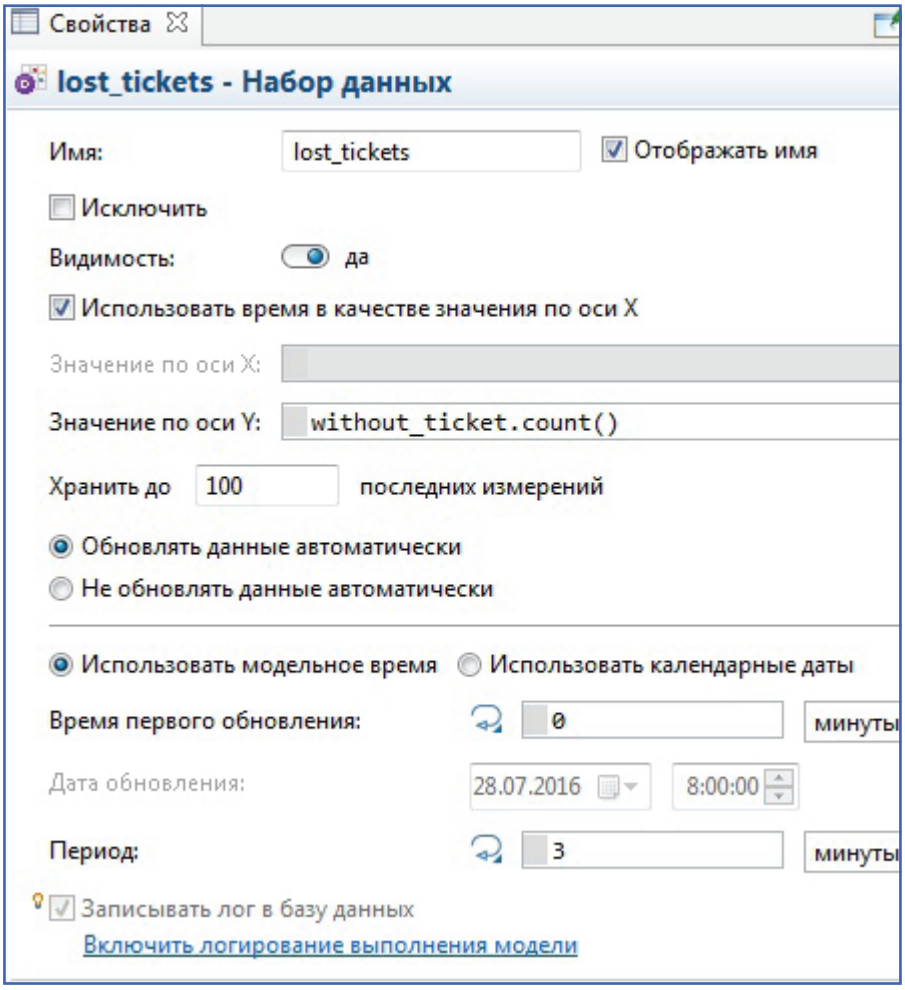

Рис. 1.27. Набор данных покупателей, не купивших билет

Шаг 3. Создание круговой диаграммы по покупателям

Для того чтобы отслеживать в режиме работы модели соотношение покупателей, купивших и не купивших билет, используем инструмент

Круговая диаграмма (рис. 1.28). Этот инструмент также находится в библиотеке Статистика. Перетащите его на рабочее поле. Перейдите в его свойства и в разделе Данные нажмите на «+», задайте название для данных, а в поле Значение вызовите метод count() объекта exit. Аналогично задайте еще один набор данных в диаграмме.

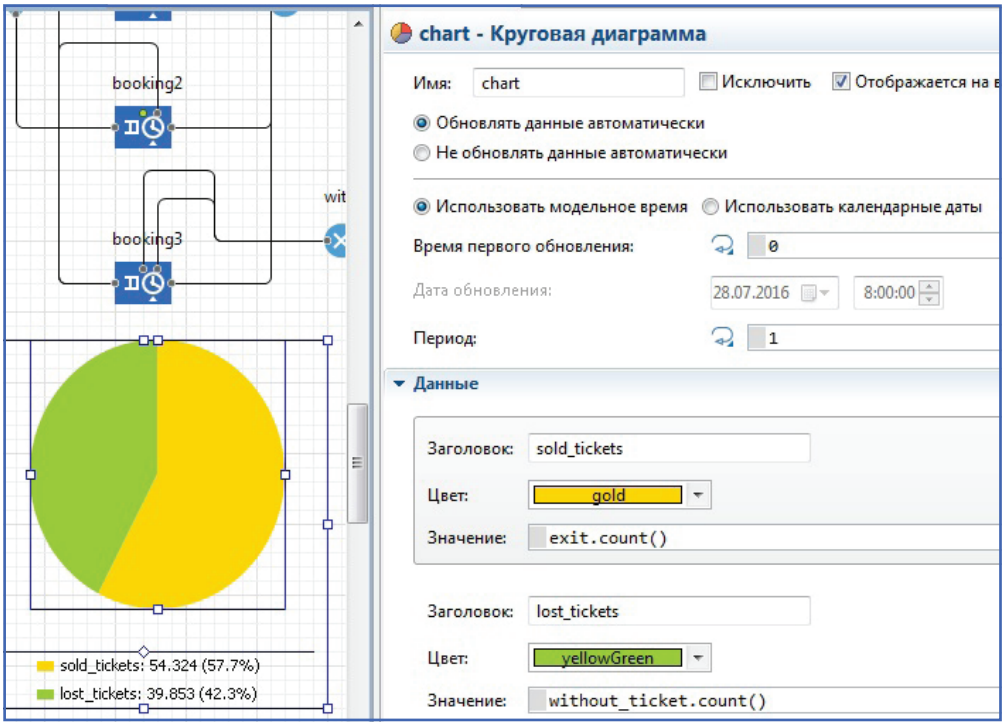

Рис. 1.28. Круговая диаграмма

Шаг 4. Создание временного графика, отражающего количество покупателей, купивших и не купивших билет

Для отображения содержимого наборов данных, созданных в шаге 1 и шаге 2, используем инструмент Временной график (рис. 1.29). Этот инструмент позволяет отслеживать во время работы модели динамически меняющиеся наборы данных или значения. Инструмент также находится на панели Статистика. Перетащите его на рабочее поле и задайте в его свойствах созданные ранее наборы данных.

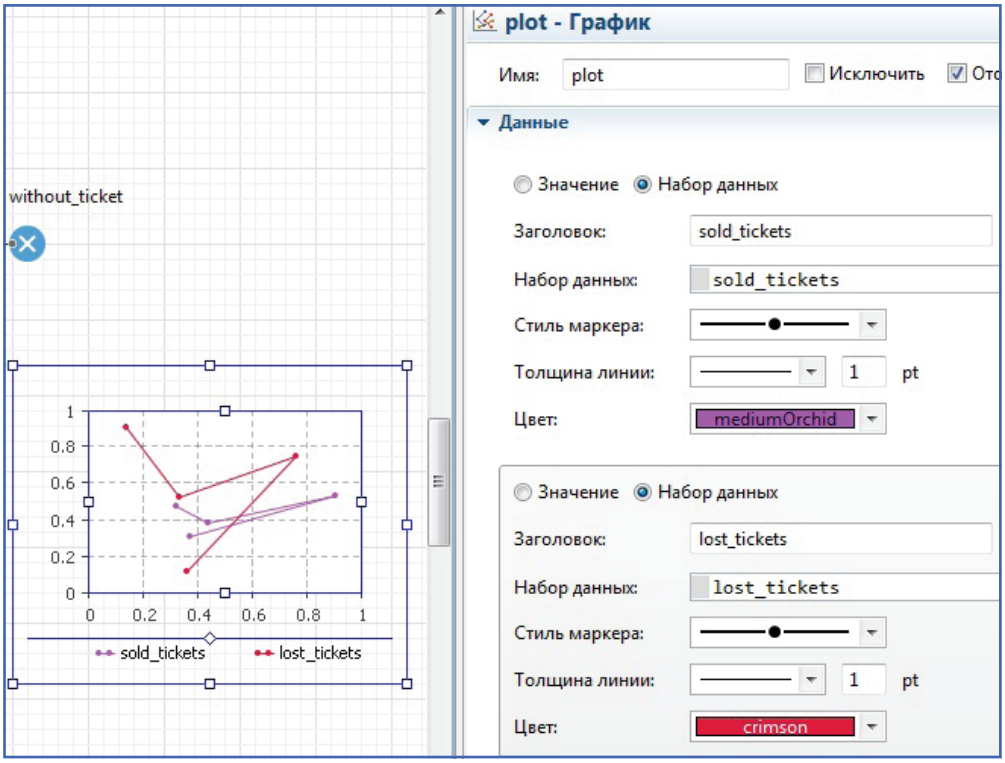

Рис. 1.29. Временной график

Шаг 5. Проверка работоспособности модели

Запустите модель. Спустя некоторое время работы модели должно получиться, как на рис. 1.30.

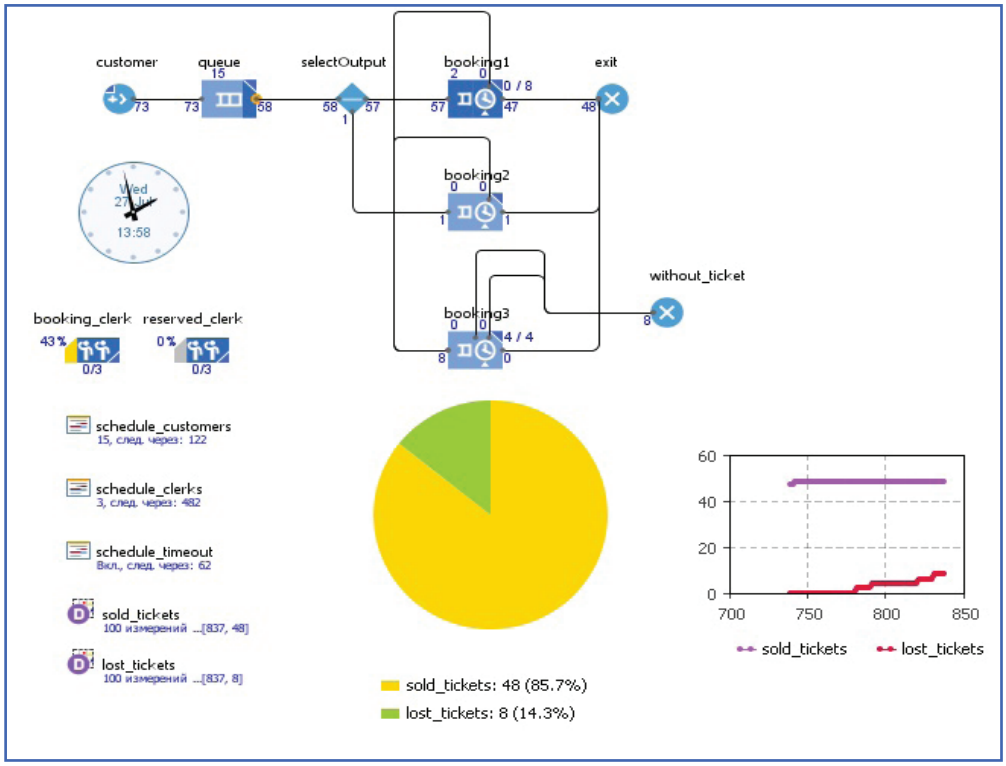

Рис. 1.30. Временной график

Самостоятельно

1. Выявите узкие места в модели и предложите решение.
2. Творческий проект. Создайте модель массового обслуживания. Принимаются собственные инициативы. Если таковых нет, то обратитесь к преподавателю.

Лабораторная работа № 2
Дискретно-событийное моделирование. Моделирование производственных систем

Задача

Промоделировать процесс производства мороженого. Мороженое производится из молока, сахара и масла в пропорциях 60:10:30. Ингредиенты поступают в реактор-смеситель из резервуаров по трубопроводам — молоко и сахар, по контейнеру — масло. В смесителе составляющие смешиваются в заданных пропорциях и смесь гомогенизируется 10 минут. Далее смесь по трубопроводу поступает в реактор заморозки. Процесс замораживания проходит 10 минут. Полученное мороженое нарезается порциями по 100 граммов и помещается в стаканчики. Стаканчики мороженого пакуются по 50 штук. Упаковки мороженого увозятся с производства.

Решение

Этап 1. Моделирование процесса смешивания ингредиентов

Шаг 1. Моделирование источников ингредиентов

Поскольку составляющие мороженого являются жидкостями или сыпучими материалами, то для моделирования работы с ними нужно использовать библиотеку моделирования потоков. Библиотека моделирования потоков находится на вкладке Палитра и содержит в себе блоки для моделирования работы с потоками и разметки пространства. Подробно описание работы каждого блока библиотеки дано в учебном пособии О. В. Лимановской «Имитационное моделирование в AnyLogic 7» (В 2 ч., ч. 1. Екатеринбург : Изд-во Урал. ун-та, 2017. 152 с.). Вид библиотеки приведен на рис. 2.1.

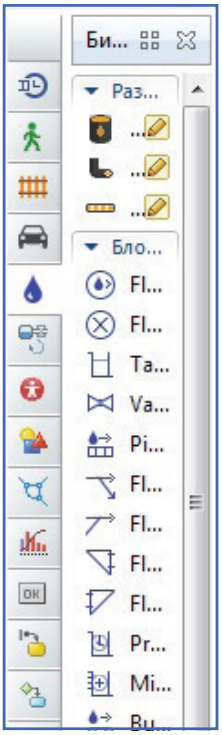

Рис. 2.1. Библиотека моделирования потоков

Источники потоков моделируются блоком FluidSource. Перетащите на рабочее поле 3 блока FluidSource. Первый из них будет моделировать источник молока в модели, второй — сахара, третий — масла (рис. 2.2).

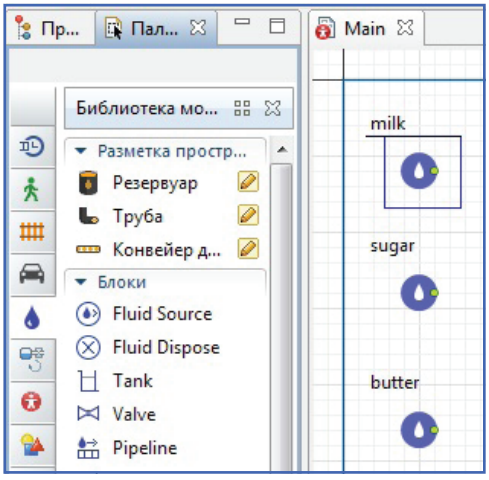

Рис. 2.2. Моделирование источников ингредиентов

В свойствах блоков задайте скорость потоков, с которой они будут поступать в модель, и ограничьте объем, который может быть произведен источником (рис. 2.3).

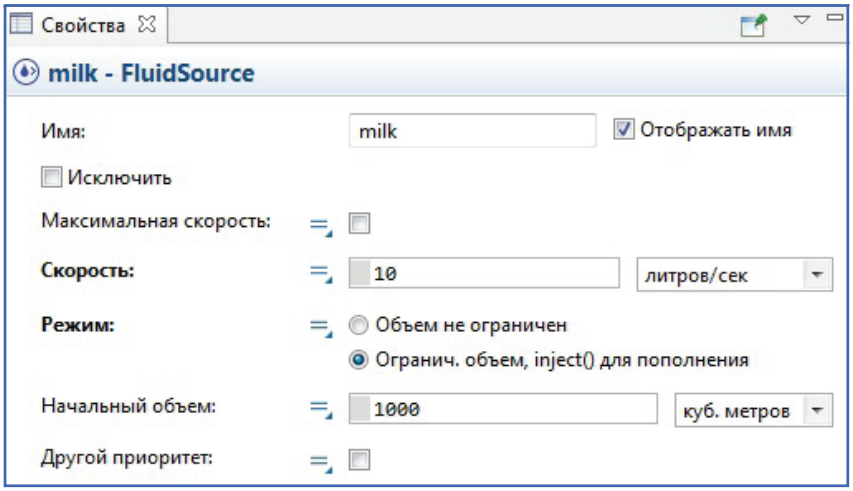

Рис. 2.3. Свойства источников ингредиентов

Шаг 2. Моделирование резервуаров ингредиентов

Поскольку скорости потребления ингредиентов в модели будут разные, нужно предусмотреть резервуары для их хранения после того, как они потупили в модель. Резервуары моделируются блоком Tank. Перетащите три блока Tank на рабочее поле и соедините их с источниками ингредиентов. Дайте им имена — milk_tank, sugar_tank, butter_tank. Должно получиться, как на рис. 2.4.

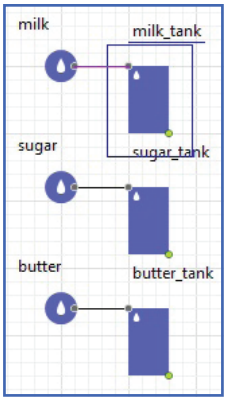

Рис. 2.4. Резервуары источников ингредиентов

В свойствах объектов Tank задайте объем резервуаров и скорости потоков на выходе из них (рис. 2.5).

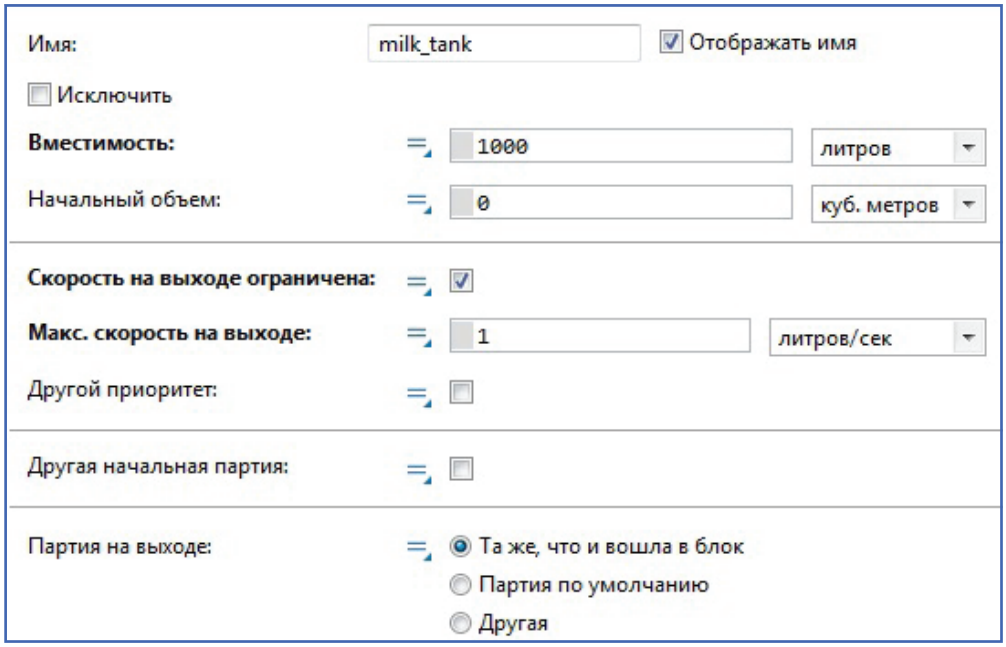

Рис. 2.5. Свойства объектов Tank

Шаг 3. Моделирование доставки ингредиентов в смеситель

Доставка жидких ингредиентов в смеситель осуществляется по трубопроводу. Трубопровод моделируется объектом Pipeline. Перетащите два объекта Pipeline на рабочее поле модели и соедините их входы с выходами резервуаров. Дайте названия трубопроводам pipeline_milk и pipeline_sugar. Доставка конденсированных ингредиентов (масла) осуществляется конвейером. Конвейер конденсированных веществ моделируется объектом BulkConveyor. Перетащите этот объект на рабочее поле модели и соедините его с выходом резервуара масла. В результате должно получиться, как на рис. 2.6.

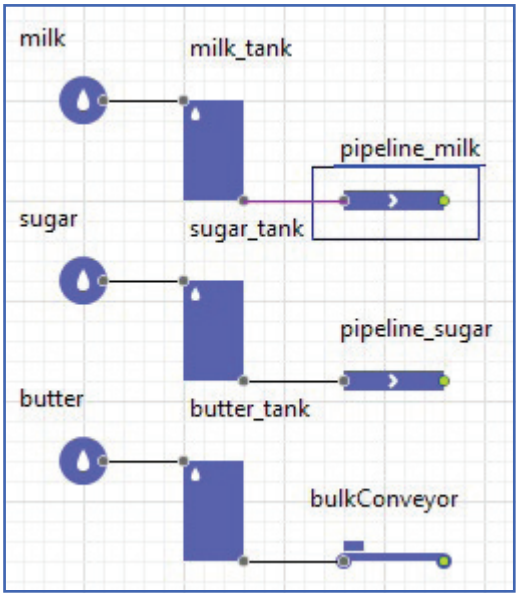

Рис. 2.6. Моделирование доставки ингредиентов в смеситель

В свойствах объектов Pipeline укажите объем трубы и скорость потока (рис. 2.7).

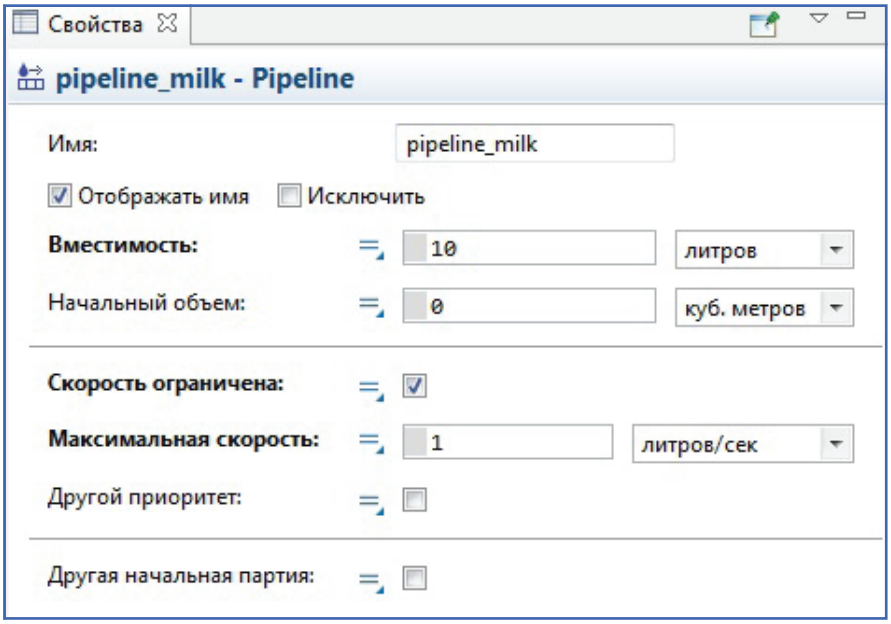

Рис. 2.7. Свойства трубопроводов молока и сахара

В свойствах объекта BulkConveyor укажите его длину и скорость конвейера. Также укажите скорость входного потока жидкости (рис. 2.8).

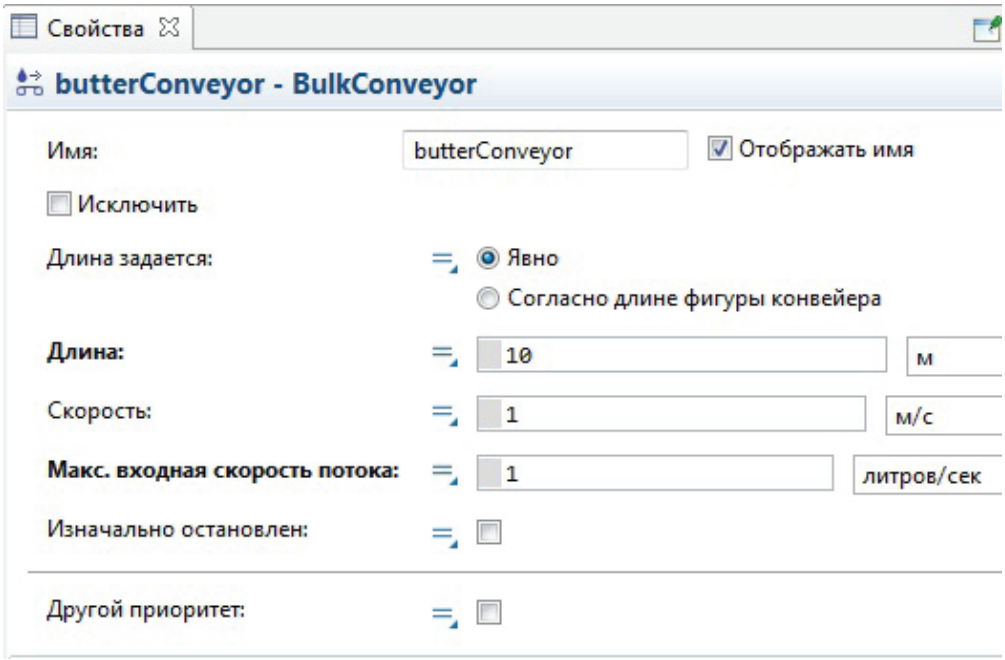

Рис. 2.8. Свойства конвейера масла

Шаг 4. Моделирование процесса смешивания

Процесс смешивания ингредиентов моделируется блоком MixTank. Этот блок имеет пять входов и один выход. Он принимает на вход составные части смеси и выдает на выходе смесь, сделанную в заданных пропорциях. Перетащите блок на рабочее поле модели и соедините его первый вход с выходом молокопровода, его второй вход — с выходом сахаропровода, а третий вход — с конвейером масла. Должно получиться, как на рис. 2.9.

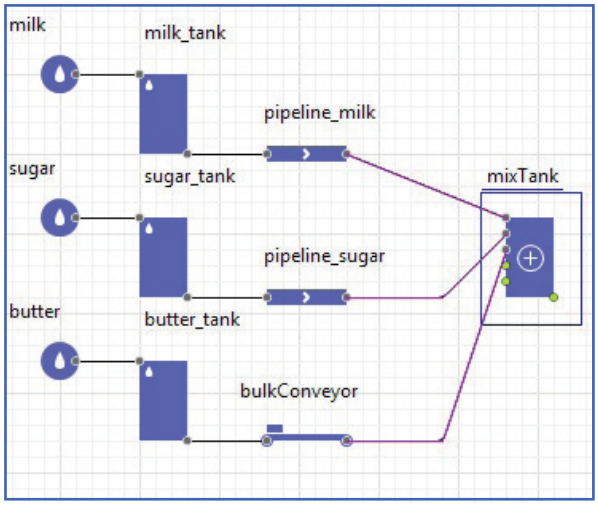

Рис. 2.9. Моделирование приготовления смеси

В свойствах объекта MixTank задайте объем смесителя, пропорции ингредиентов смеси и время смешивания (рис. 2.10). Также в свойствах объекта MixTank задайте скорость выходного потока смеси.

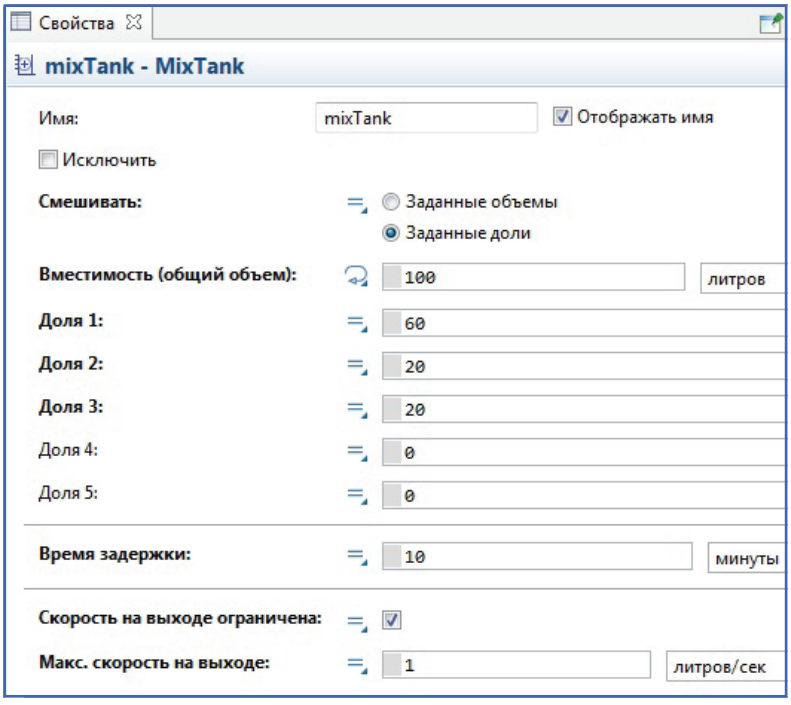

Рис. 2.10. Свойства смесителя

Также в свойствах смесителя укажем, что смесь на выходе из него имеет другие свойства, чем входные ингредиенты. Для этого в пункте Партия на выходе укажем, что образуется другая партия, дадим ей название и поменяем цвет (рис. 2.11).

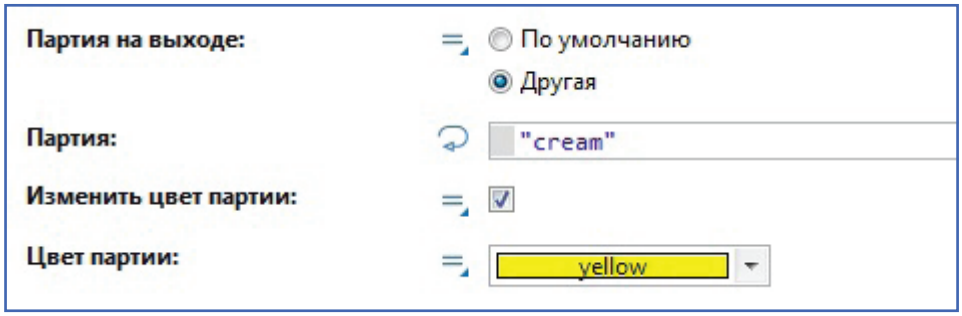

Рис. 2.11. Моделирование изменения свойств смеси

Этап 2. Моделирование процесса замораживания смеси

Шаг 1. Моделирование доставки смеси до реактора замораживания

Доставка смеси до реактора заморозки осуществляется по трубопроводу (рис. 2.12). Промоделируйте его таким же, как молокопровод.

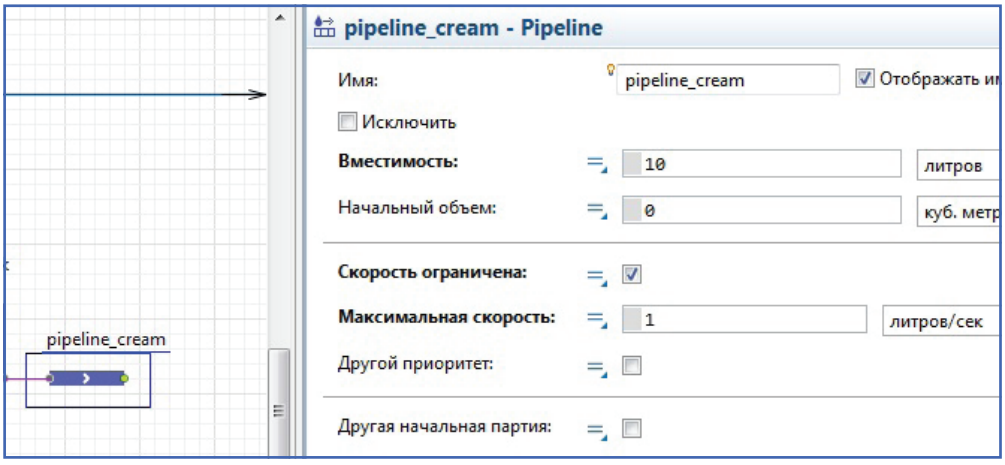

Рис. 2.12. Моделирование доставки смеси до заморозки

Шаг 2. Моделирование процесса заморозки

Процесс заморозки моделируется объектом ProcessTank. Этот объект моделирует наполнение резервуара и процесс в нем. Перетащите этот объект на рабочее поле и соедините его вход с выходом трубопровода смеси.

В свойствах объекта ProcessTank задайте объем реактора, время заморозки и скорость смеси на выходе из реактора. Также в свойствах укажем тот факт, что смесь в нем приобретает другие качества. Для этого укажем в пункте Партия на выходе, что партия на выходе из реактора образуется другая, зададим ее название и поменяем ей цвет (рис. 2.13).

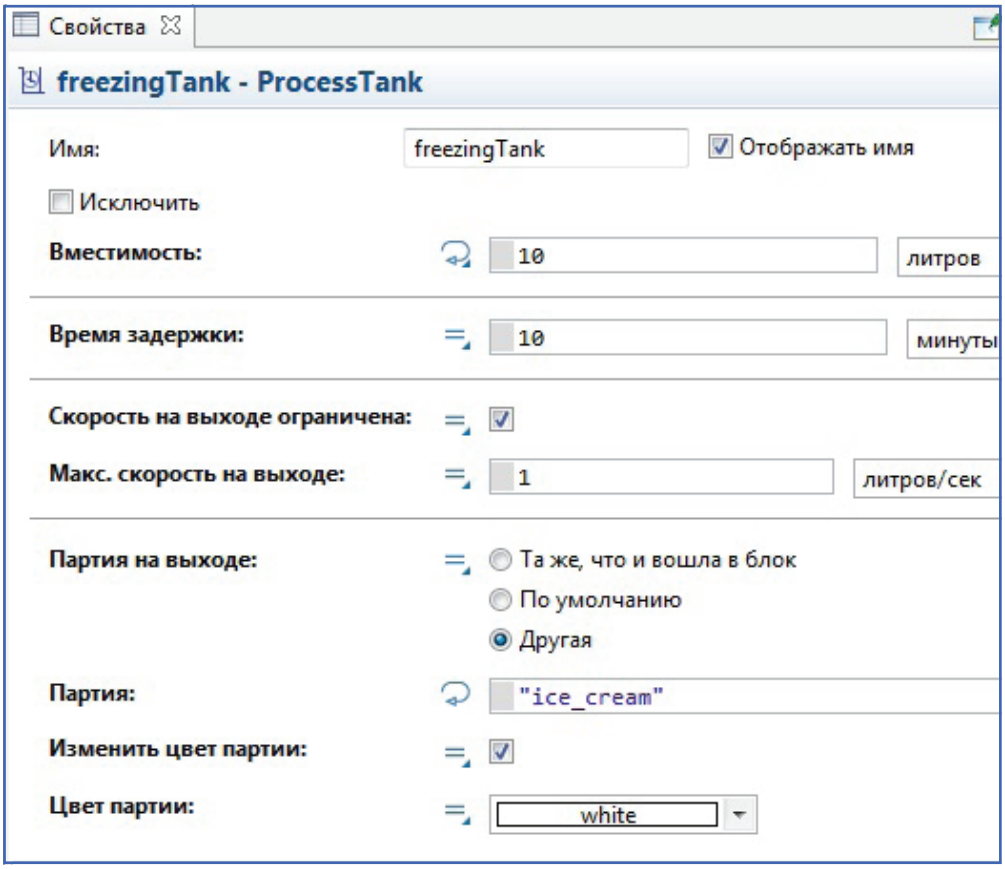

Рис. 2.13. Моделирование процесса заморозки

Этап 3. Моделирование процесса разделения на порции и упаковки мороженого

Шаг 1. Моделирование доставки замороженной смеси на стадию разделения на порции

Поскольку смесь заморожена, то доставка ее осуществляется конвейером для конденсированных веществ также, как и доставка масла от его резервуара. Промоделируете этот конвейер аналогично конвейеру масла (рис. 2.14). В свойствах задайте его длину, скорость и скорость потока, входящего на него.

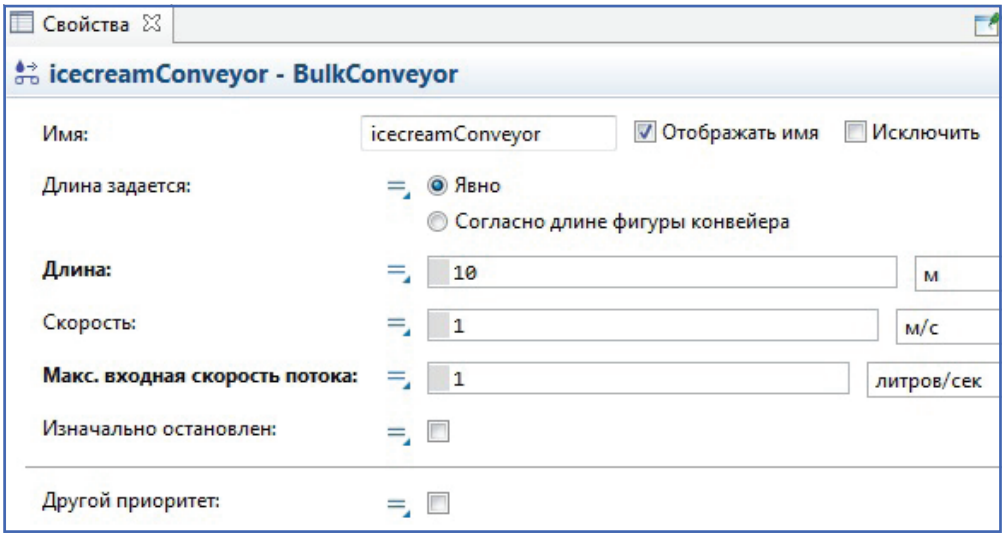

Рис. 2.14. Моделирование доставки смеси до процесса разделения на порции

Шаг 2. Моделирование процесса разделения на порции

Порция мороженого — это уже не поток вещества, а единичная заявка, поэтому процесс разделения на порции моделируем блоком FluidToAgent. Этот блок создает агентов из заданного объема жидкости. Перетащите его на рабочее поле и соедините с конвейером. В свойствах блока задайте объем смеси, из которого получается один агент (рис. 2.15).

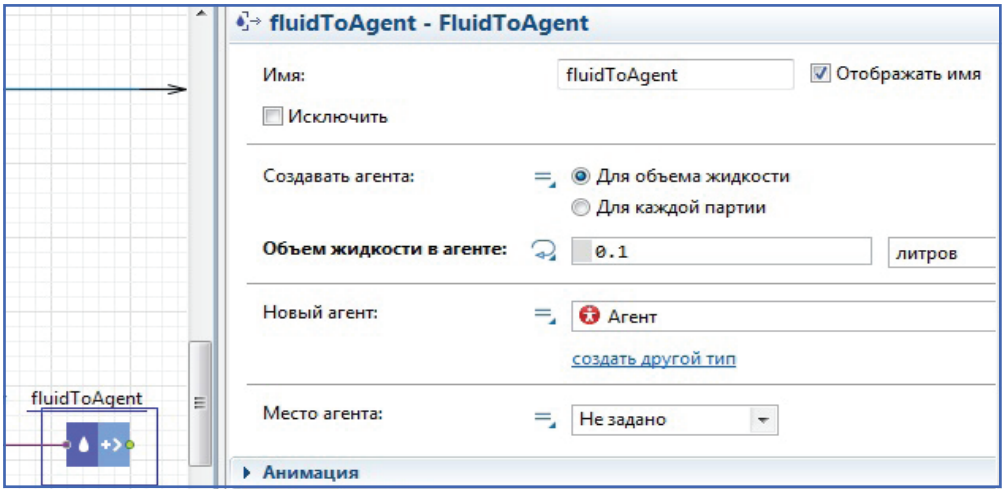

Рис. 2.15. Моделирование процесса разделения на порции

Этап 4. Моделирование процесса раскладки мороженого по стаканчикам

Шаг 1. Моделирование поставки стаканчиков для мороженого

Стаканчики для мороженого — товар штучный, поэтому для моделирования их появления в модели используется библиотека моделирования процессов. Для моделирования появления стаканчиков используется блок Source. Перетащите его на рабочее поле и задайте его свойства, а именно интенсивность появления стаканчиков — 10 штук в минуту (рис. 2.16).

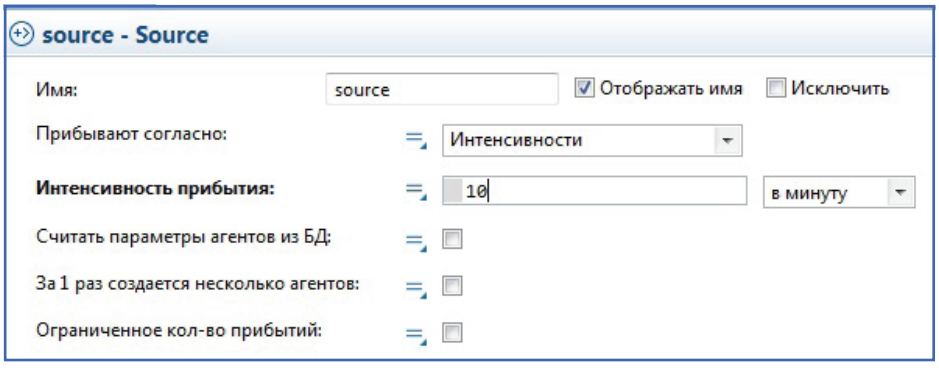

Рис. 2.16. Моделирование поставки стаканчиков

Поскольку мороженое производится 20 минут в модели, то создавать стаканчики раньше не нужно. Нужно указать в свойствах блока Source, что время начала его работы отложено. Для этого в разделе свойств Специфические установите галочку в пункте Установить время начала и задайте время задержки начала его работы (рис. 2.17).

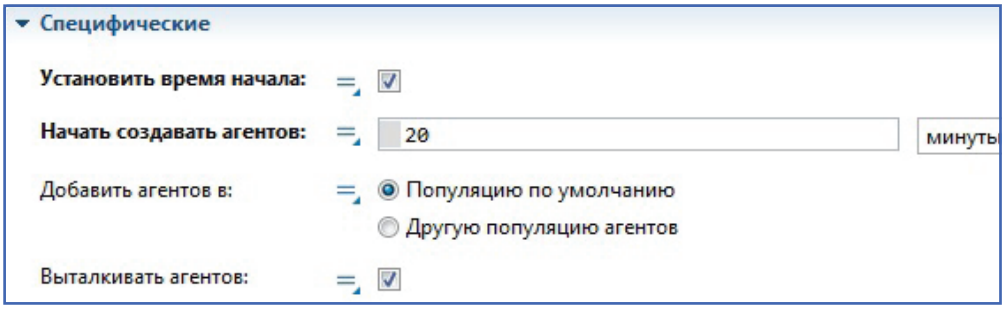

Рис. 2.17. Моделирование отложенного начала работы

Шаг 2. Моделирование накопителей мороженого и стаканчиков

Поскольку скорость производства мороженого и стаканчиков в модели разная, то необходимы их накопители. Накопители моделируются блоком Queue. Перетащите два блока в рабочее поле и соедините один из них с источником стаканчиков, второй — с блоком FluidToAgent (рис. 2.18).

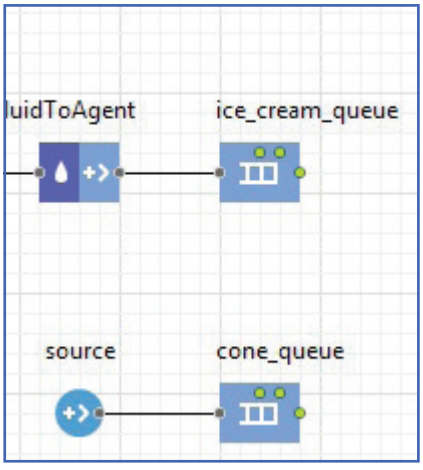

Рис. 2.18. Моделирование накопителей мороженого и стаканчиков

В свойствах очередей отметьте пункт Максимальная вместимость.

Шаг 3. Моделирование сборки мороженого

Сборка штучных заявок моделируется блоком Assembler. Этот блок имеет пять входов и один выход. Он может принимать до пяти агентов и собирать из них нового агента. Перетащите этот блок на рабочее поле модели (рис. 2.19). К первому его входу присоедините выход очереди мороженого, ко второму — выход очереди стаканчиков. В свойствах блока укажите количество каждого ресурса для сборки конечного продукта. В нашем случае для одного стаканчика мороженого требуется одна порция мороженого и один стаканчик. Также в свойствах блока задайте время сборки.

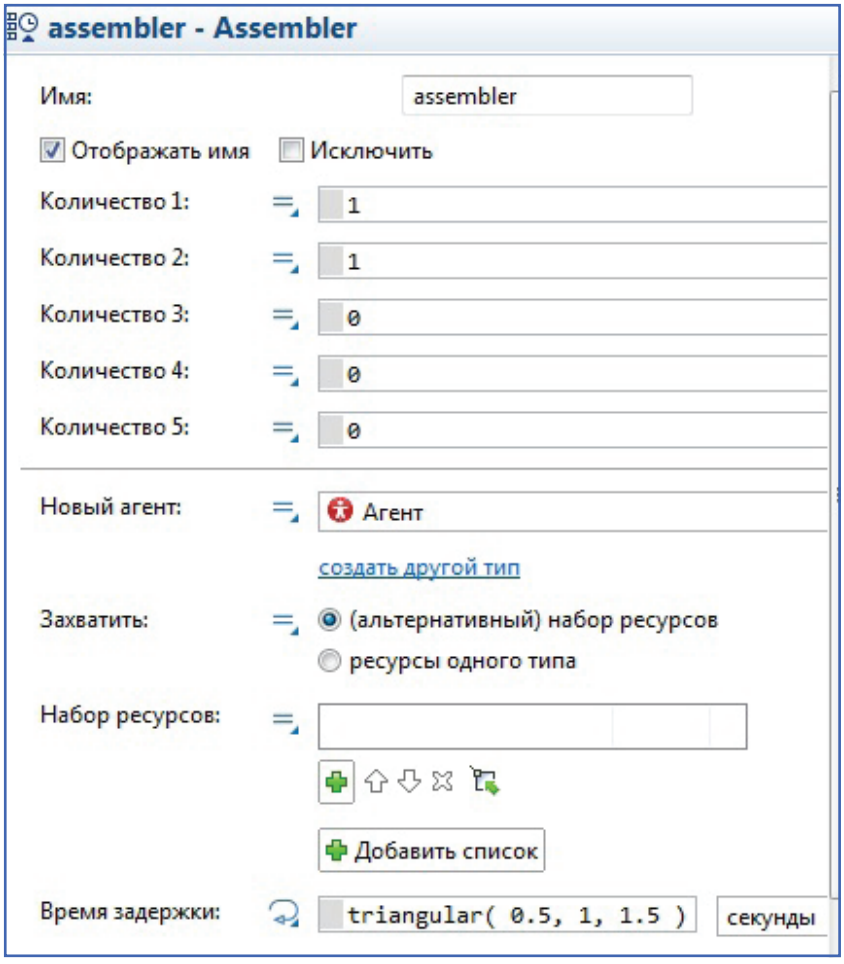

Рис. 2.19. Моделирование сборки мороженого

Этап 5. Моделирование упаковки мороженого

Шаг 1. Моделирование доставки стаканчиков мороженого до упаковщика

Этот процесс моделируется конвейером (блоком Conveyor). Промоделируйте его аналогично конвейеру порции мороженого (рис. 2.20).

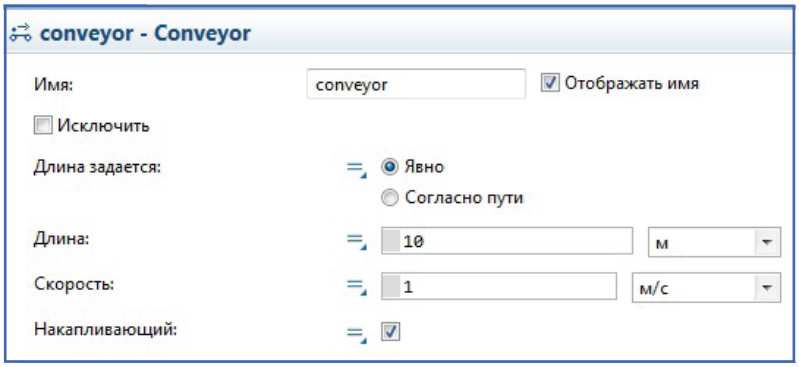

Рис. 2.20. Моделирование доставки мороженого до места упаковки

Шаг 2. Моделирование процесса упаковки

Любой процесс моделируется объектом Service, в свойствах которого задается время процесса и его ресурсы. Ресурсы мы создадим позже. Пока просто укажем время процесса. Перетащите объект Service на рабочее поле и соедините его с конвейером стаканчиков мороженого (рис. 2.21). Задайте в его свойствах время упаковки — среднее 1 секунда.

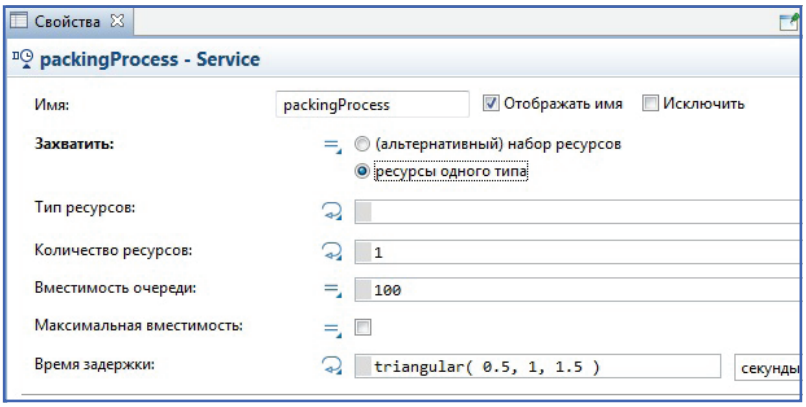

Рис. 2.21. Моделирование процесса упаковки

Шаг 3. Моделирование упаковки мороженого

Упаковку мороженого промоделируем объектом Batch, который собирает партии из входящих в него заявок (рис. 2.22). В свойствах блока задается объем партии. Перетащите его на рабочее поле и задайте в его свойствах объем партии — 50 штук.

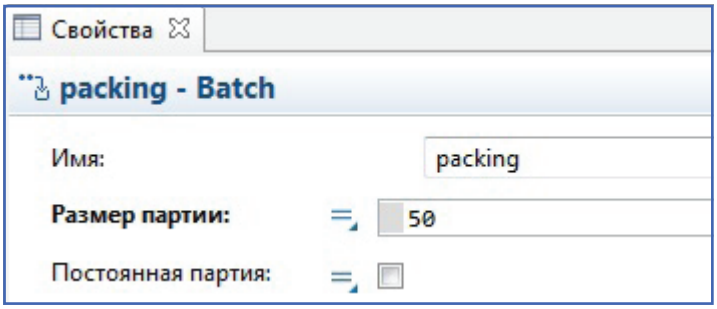

Рис. 2.22. Моделирование упаковки

Шаг 4. Моделирование увоза упаковок мороженого

Увоз упаковок промоделируйте объектом Sink (рис. 2.23).

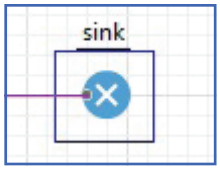

Рис. 2.23. Моделирование увоза упаковок мороженого

В результате модель должна выглядеть примерно, как на рис. 2.24.

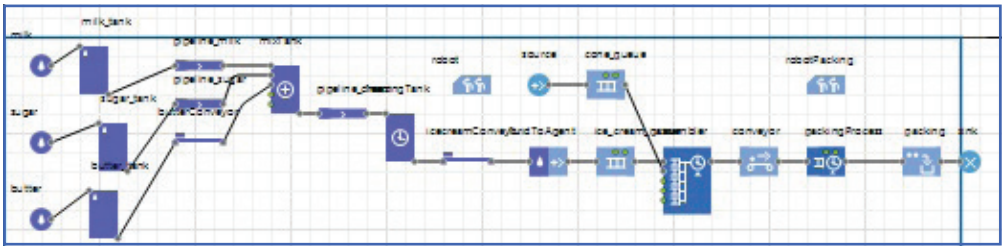

Рис. 2.24. Модель производства мороженого

Шаг 5. Проверка работоспособности модели

Запустите модель. Должно получиться примерно, как на рис. 2.25.

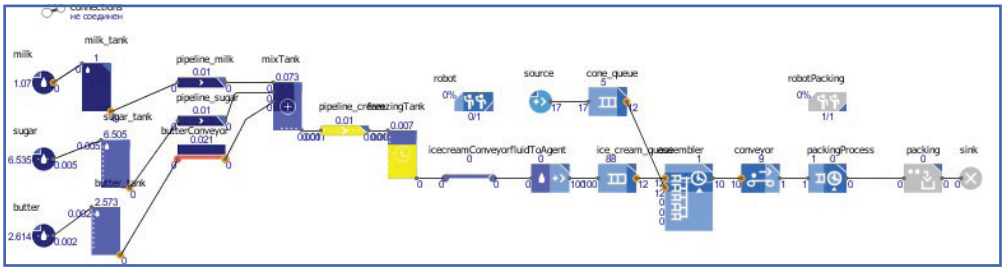

Рис. 2.25. Работа модели производства мороженого

Этап 6. Создание ресурсов в модели

Шаг 1. Моделирование ресурсов для сборки мороженого

Ресурсы моделируются блоком ResourcePool, в котором можно указать количество ресурсов и расписание их работы. В нашем случае сборка мороженого производится роботом, который работает круглосуточно. Поэтому расписание его работы составлять не нужно. Просто перетащите объект ResourcePool на рабочее поле и задайте его имя (рис. 2.26).

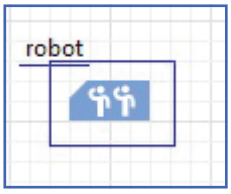

Рис. 2.26. Моделирование ресурсов для сборки мороженого

В свойствах объекта — тип ресурса Переносной, поскольку сам робот не может передвигаться от объекта к объекту (рис. 2.27). В пункте Количество ресурсов задайте 1. Поскольку расписание работы робота не задается, то в пункте Количество задано, выбираем вариант Напрямую.

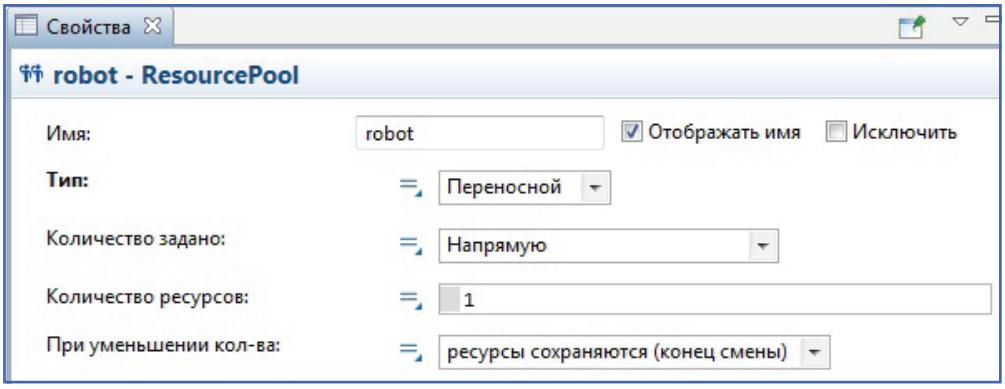

Рис. 2.27. Свойства ресурсов для сборки мороженого

Шаг 2. Моделирование ресурсов для упаковки мороженого

Поскольку упаковка мороженого также производится роботом, то моделирование ресурсов для упаковки мороженого проводится аналогично моделированию ресурсов для сборки мороженого, только для различия ресурсов задайте ему имя robotPacking.

Шаг 3. Привязка ресурсов к процессам

Для привязки ресурсов к процессам нужно в свойствах блока Assembler и блока Service указать ресурсы. Поскольку используется по одному роботу, то указываем ресурсы одного типа (рис. 2.28).

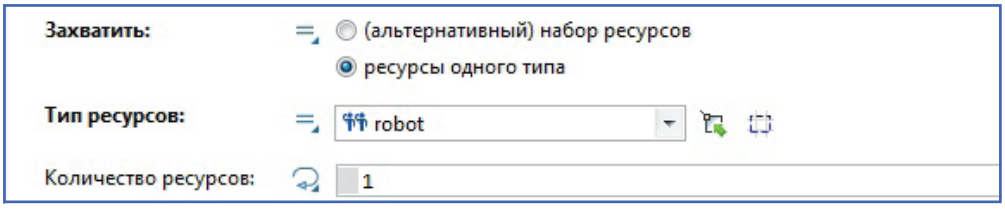

Рис. 2.28. Захват ресурсов для процесса сборки мороженого

Этап 7. Добавление новых агентов в модели

Для наглядности работы модели создадим агентов, имитирующих робота, стаканчик для мороженого, мороженое и стаканчик мороженого с собственной анимацией.

Шаг 1. Добавление агента Робот

Агент Робот является ресурсом в модели, поэтому для его создания используем инструмент Тип ресурса из библиотеки моделирования процессов. Перетащите на рабочее поле этот объект, и откроется окно Мастера по созданию нового агента-ресурса (см. рис. 2.27). В нем укажите имя агента-ресурса Robot (рис. 2.29).

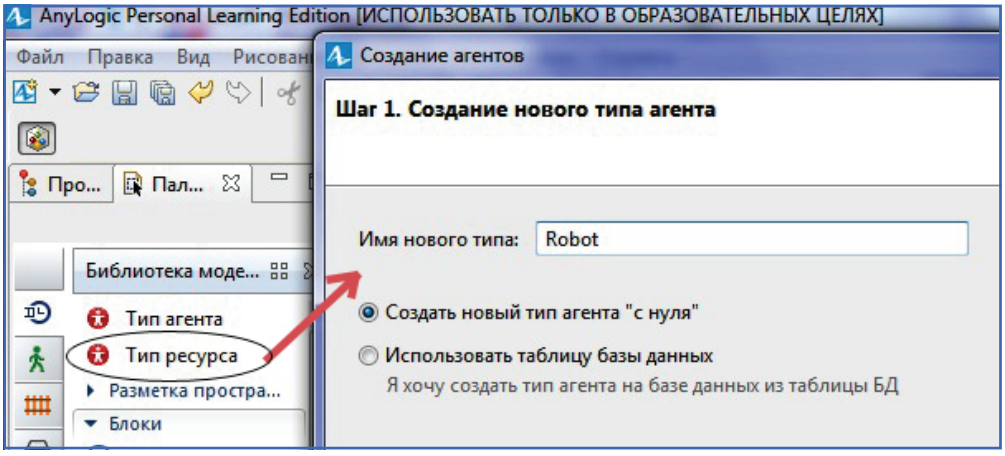

Рис. 2.29. Первый шаг Мастера по созданию нового агента-ресурса

Нажмите кнопку Далее. Появится окно второго шага Мастера (рис. 2.30). В нем нужно выбрать анимацию агента-ресурса. Выберите из раздела Производство Робот1.

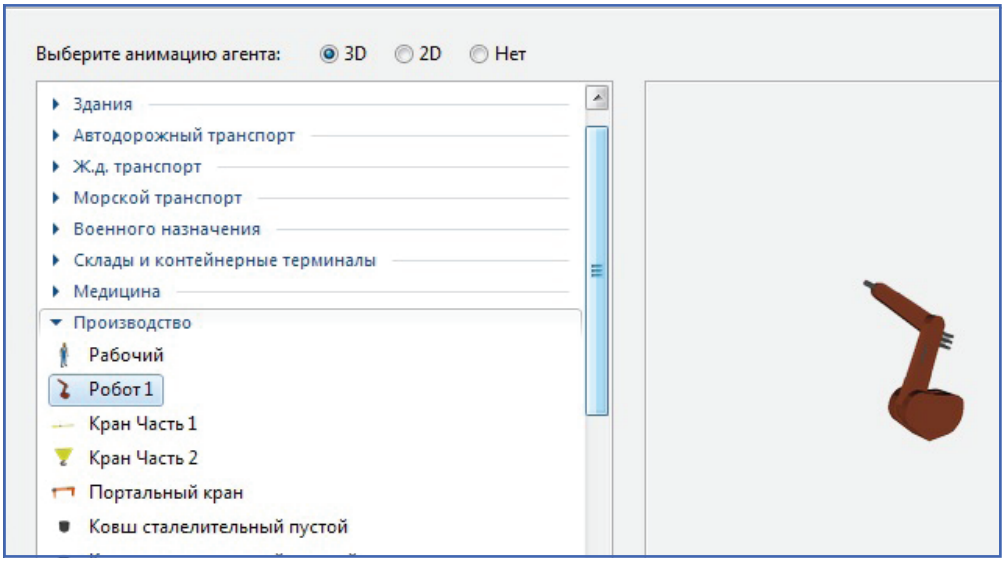

Рис. 2.30. Второй шаг Мастера по созданию нового агента-ресурса

Шаг 2. Привязка созданного агента-ресурса к объекту ResourcePool

Поскольку операции сборки и упаковки мороженого выполняет робот, то к обоим объектам resourcePool (robot и packingRobot) привязываем агент-ресурс Robot. Для этого в свойствах объектов в разделе Специфические укажите тип ресурса Robot (рис. 2.31).

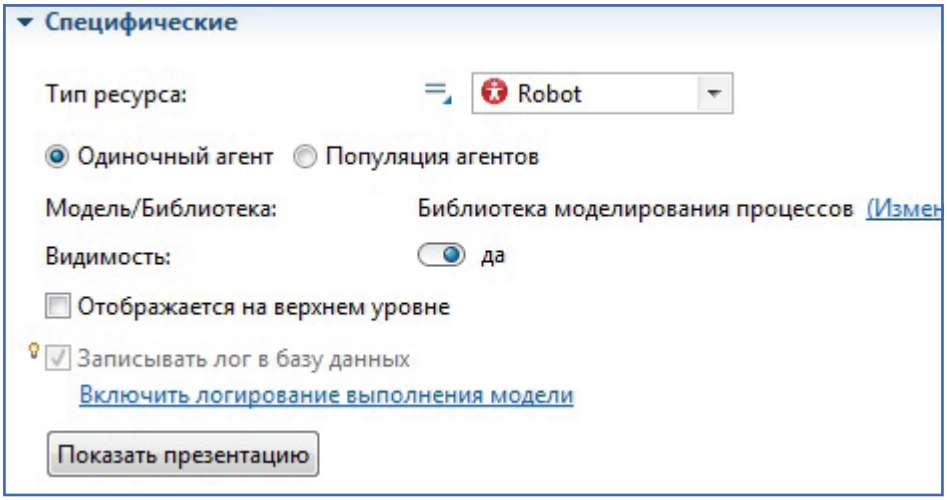

Рис. 2.31. Привязка агента-ресурса к объекту resourcePool

Шаг 3. Добавление агента Мороженое

Мороженое является заявкой в модели, поэтому для создания агента Мороженое используется инструмент Тип агента из библиотеки моделирования процессов. Перетащите его на рабочее поле модели, и откроется первый шаг мастера по созданию агентов-заявок (рис. 2.32).

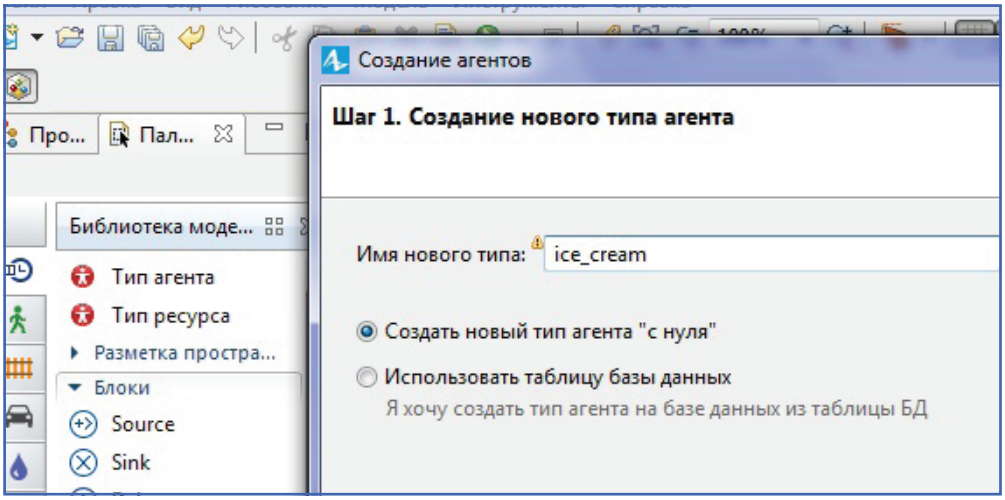

Рис. 2.32. Первый шаг Мастера по созданию агента-заявки

На первом шаге задайте имя агенту-заявке ice_cream и нажмите кнопку Готово. После закрытия Мастера откроется окно только что созданного агента. Анимацию агента создадим сами. Для этого перейдите на вкладку Презентация (рис. 2.33). На этой вкладке собраны все инструменты рисования в модели.

Изобразим мороженое в виде круга. Выберите инструмент Овал двойным кликом мыши и нарисуйте круг вначале координат рабочего поля агента. Задайте в свойствах фигуры ее размер, цвет и координаты (рис. 2.34).

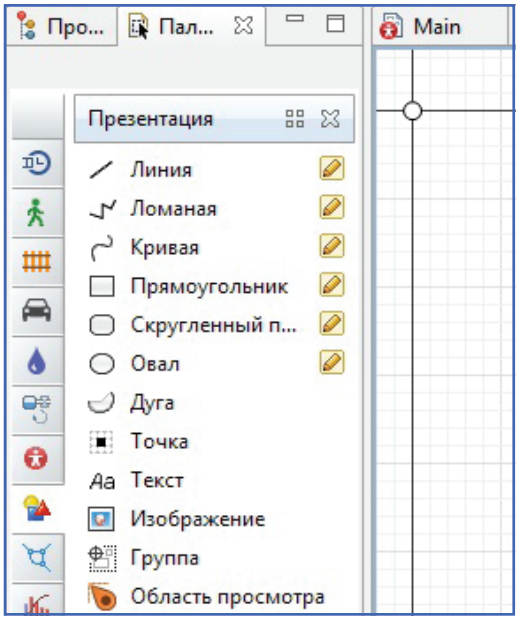

Рис. 2.33. Вкладка Презентация

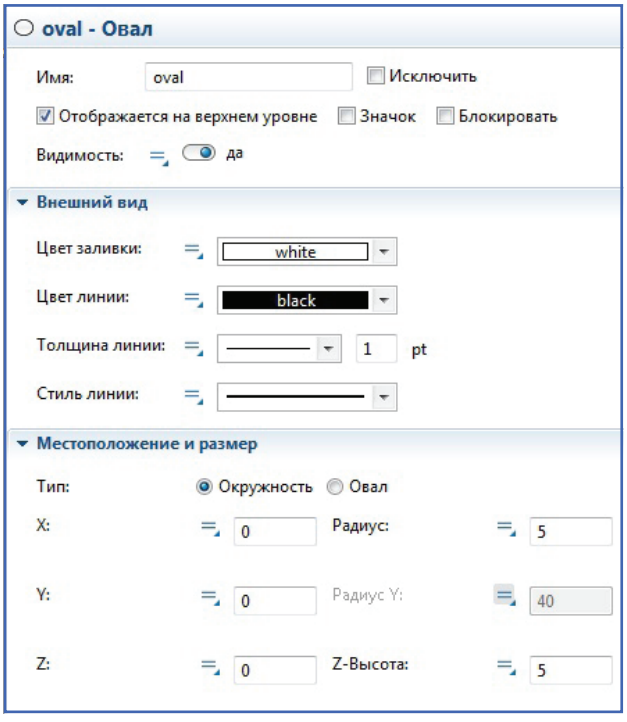

Рис. 2.34. Свойства круга

Шаг 4. Добавление агента стаканчик для мороженого

Повторите действия шага 3. Назовите агента cone, а в качестве анимации задайте прямоугольник с параметрами, как на рис. 2.35.

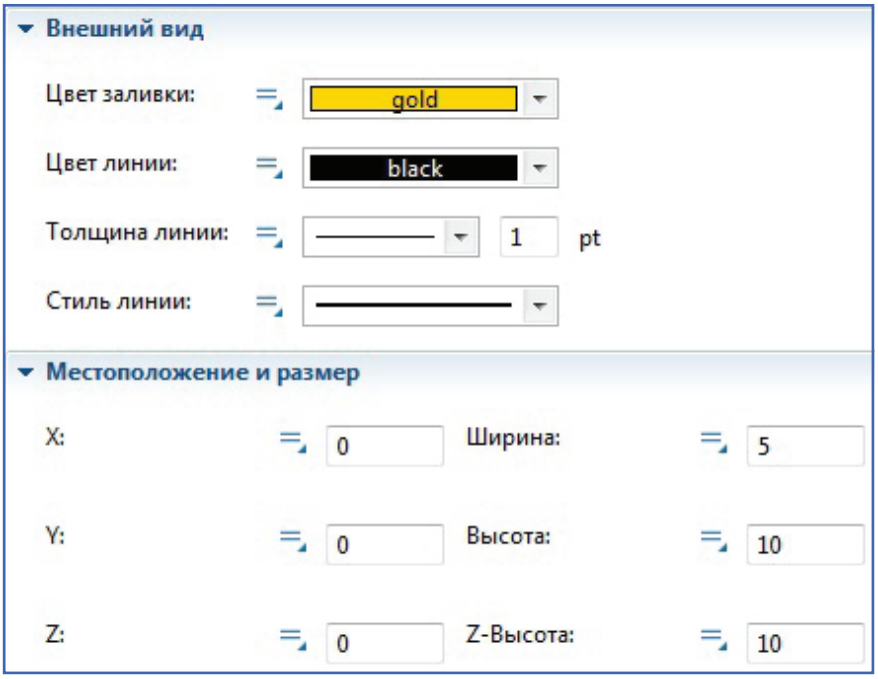

Рис. 2.35. Свойства стаканчика

Шаг 5. Добавление агента стаканчик мороженого

Повторите действия шага 3. Назовите агента ice_cream_cone. В качестве анимации соедините в агенте круг и прямоугольник (рис. 2.36).

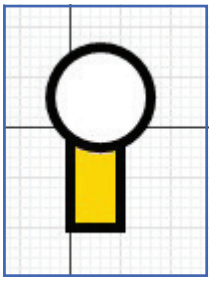

Рис. 2.36. Анимация агента стаканчик мороженого

Этап 8. Анимация модели

Шаг 1. Создание примерного плана цеха

В любом графическом редакторе нарисуйте примерный план цеха, который должен выглядеть, как на рис. 2.37.

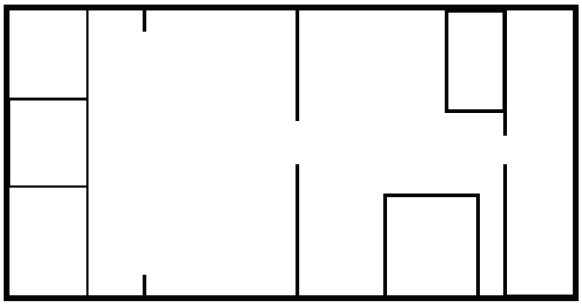

Рис. 2.37. План цеха

Шаг 2. Перенос плана цеха в модель

Для размещения любых изображений в модели используется инструмент Изображение из палитры Презентация. Перетащите инструмент на рабочее поле модели, и откроется окно проводника, в котором нужно выбрать файл с изображением (рис. 2.38).

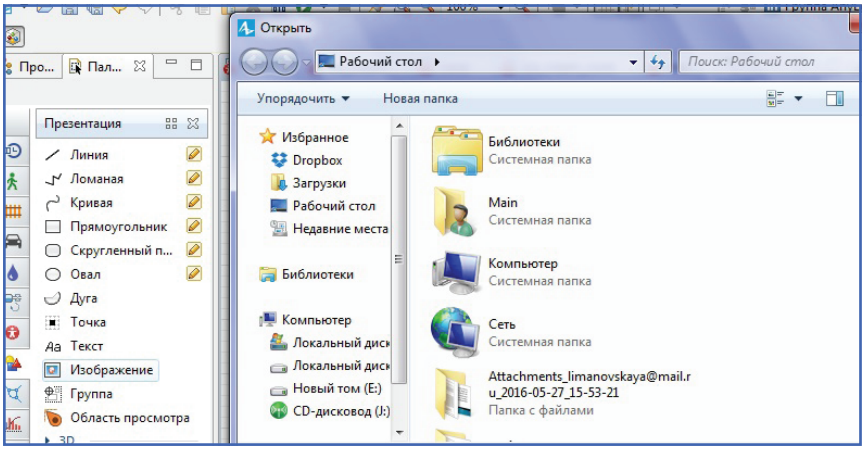

Рис. 2.38. Вставка изображения в модель

После выбора файла изображение плана цеха вставится в модель.

Шаг 3. Анимация резервуаров с ингредиентами

Для анимации резервуаров используется инструмент Tank из библиотеки моделирования потоков. Двойным щелчком мыши выберите этот инструмент и нарисуйте в левом верхнем квадрате плана резервуар (рис. 2.39).

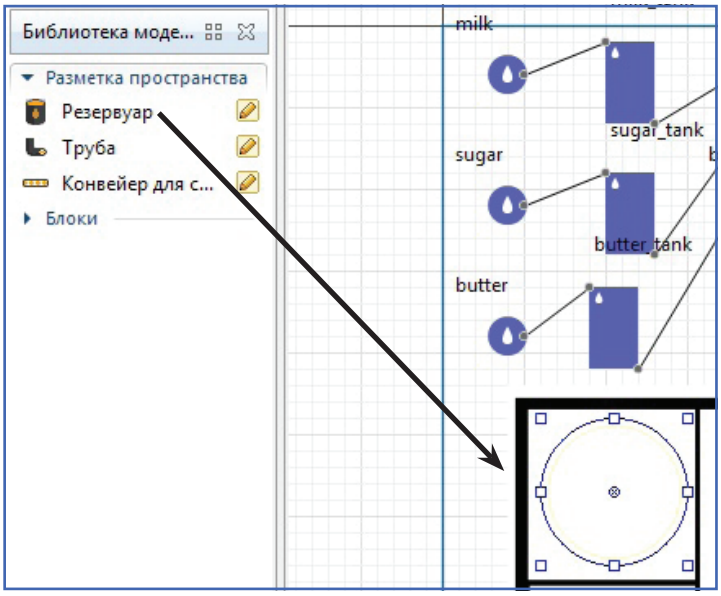

Рис. 2.39. Анимация резервуаров ингредиентов

В свойствах объекта Резервуар задайте его имя milkTank, цвет и размеры (рис. 2.40).

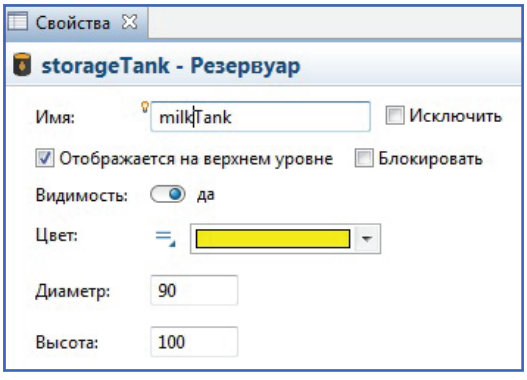

Рис. 2.40. Свойства резервуара молока

Аналогичным образом создайте еще два резервуара: один — для сахара, другой — для масла.

Шаг 4. Привязка анимации к резервуарам

Выделите в модели объект milk_tank и перейдите в его свойства (рис. 2.41). В свойствах объекта в разделе Анимация в пункте Резервуар выберите созданный на шаге 3 резервуар milkTank.

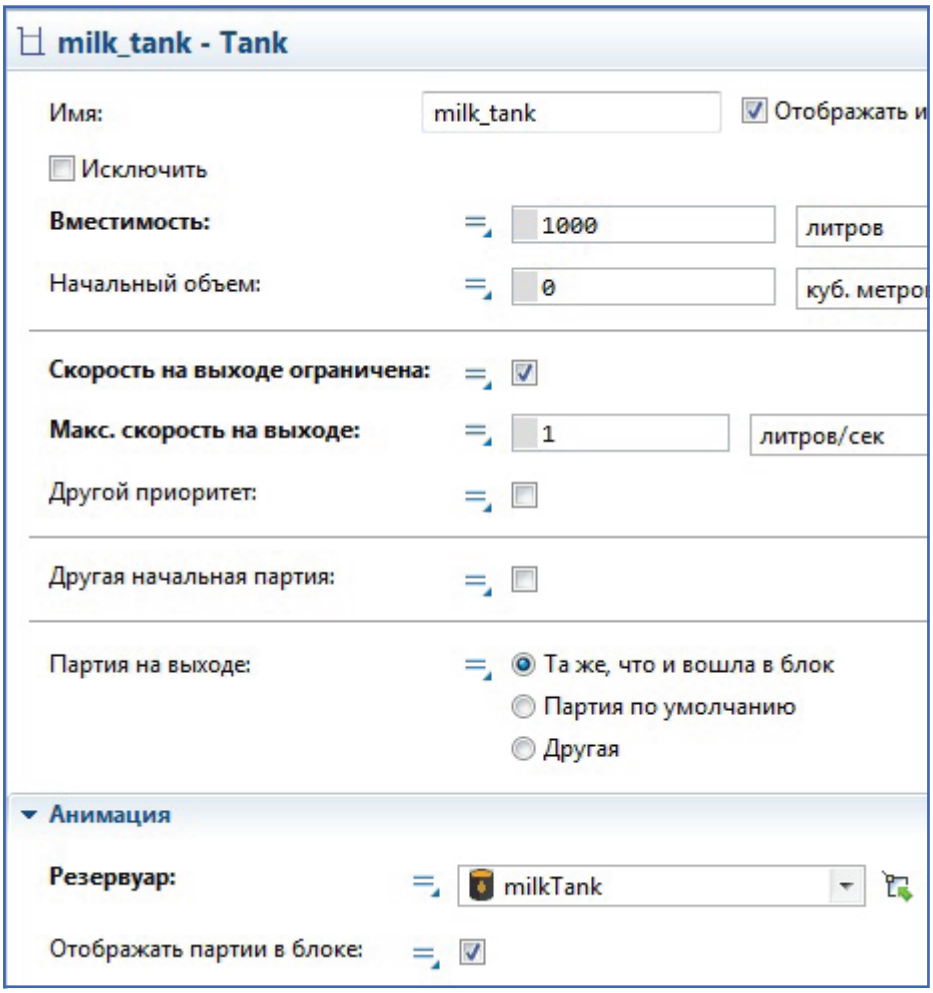

Рис. 2.41. Анимация резервуара молока

Аналогичным образом анимируйте резервуары сахара и масла.

Шаг 5. Анимация смесителя

Для анимации смесителя используется также инструмент Резервуар (рис. 2.42). Нарисуйте его посередине первого помещения в цехе. В свойствах задайте его имя mixTank, размеры и цвет.

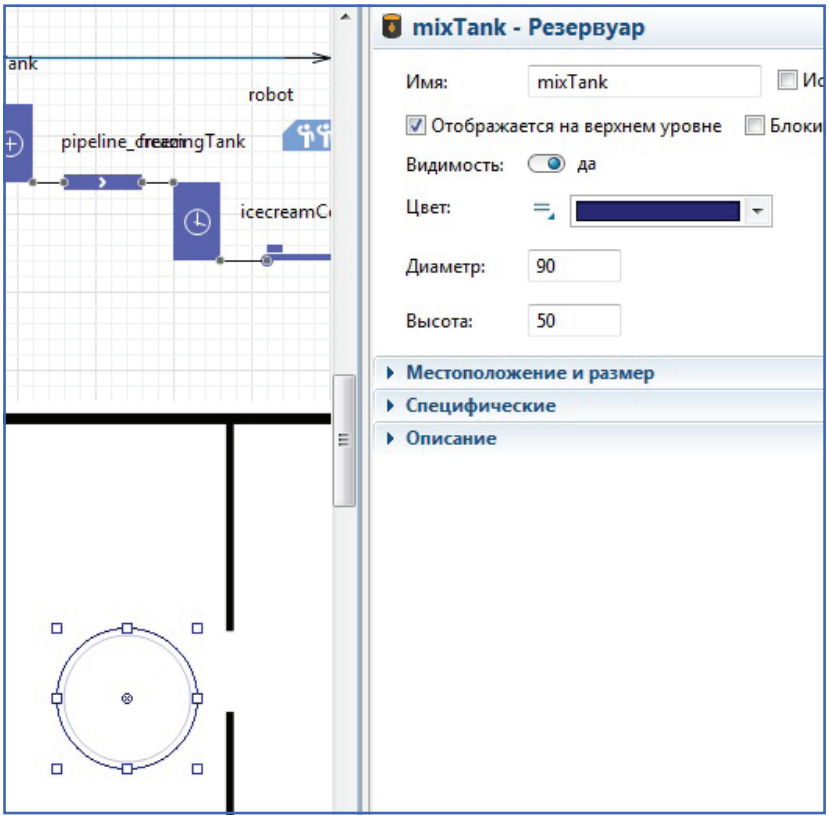

Рис. 2.42. Анимация смесителя

Привяжите созданный резервуар к блоку mixTank в модели (рис. 2.43).

Рис. 2.43. Привязка анимации к смесителю

Будьте внимательны: если имя блока совпадает с именем резервуара, то компилятор модели выдаст ошибку. Поменяйте имя резервуара.

Шаг 6. Анимация доставки ингредиентов до смесителя

Молоко и сахар до смесителя доставляются по трубопроводам. Трубопроводы анимируются с помощью инструмента Труба из библиотеки моделирования потоков (рис. 2.44). Активируйте элемент Труба, щелкнув на нем дважды, и нарисуйте трубу от резервуара с молоком до смесителя. В свойствах объекта Труба задайте ее имя, цвет и диаметр.

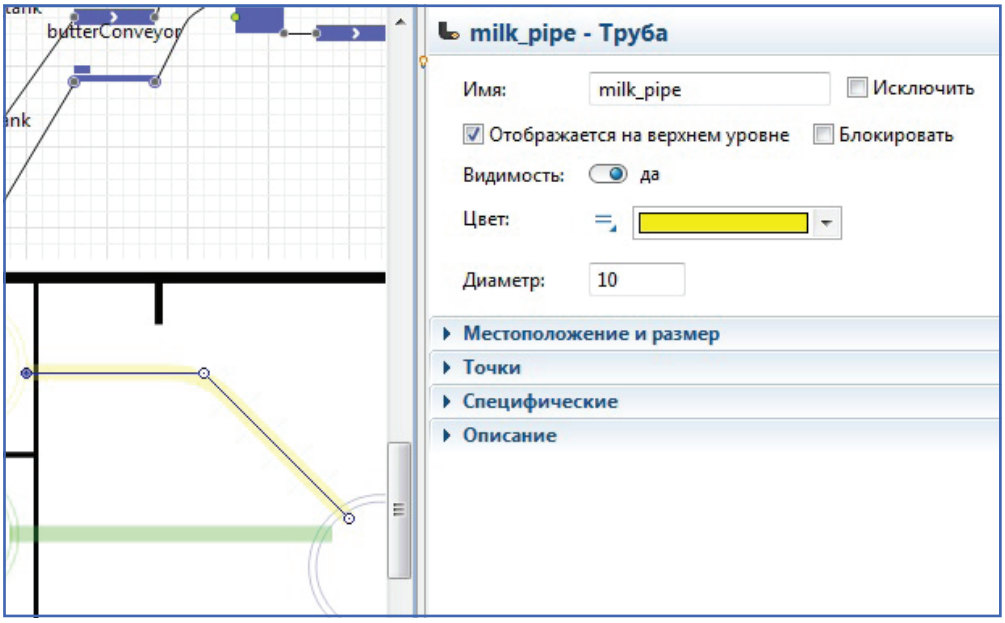

Рис. 2.44. Анимация трубопровода молока

Аналогичным образом нарисуйте трубопровод для сахара.

Доставка масла осуществляется конвейером. Для анимации конвейера используется инструмент Конвейер из библиотеки моделирования потоков (рис. 2.45). Активируйте его, щелкнув на нем дважды, и нарисуйте конвейер от резервуара с маслом до смесителя. В свойствах объекта задайте его имя и ширину.

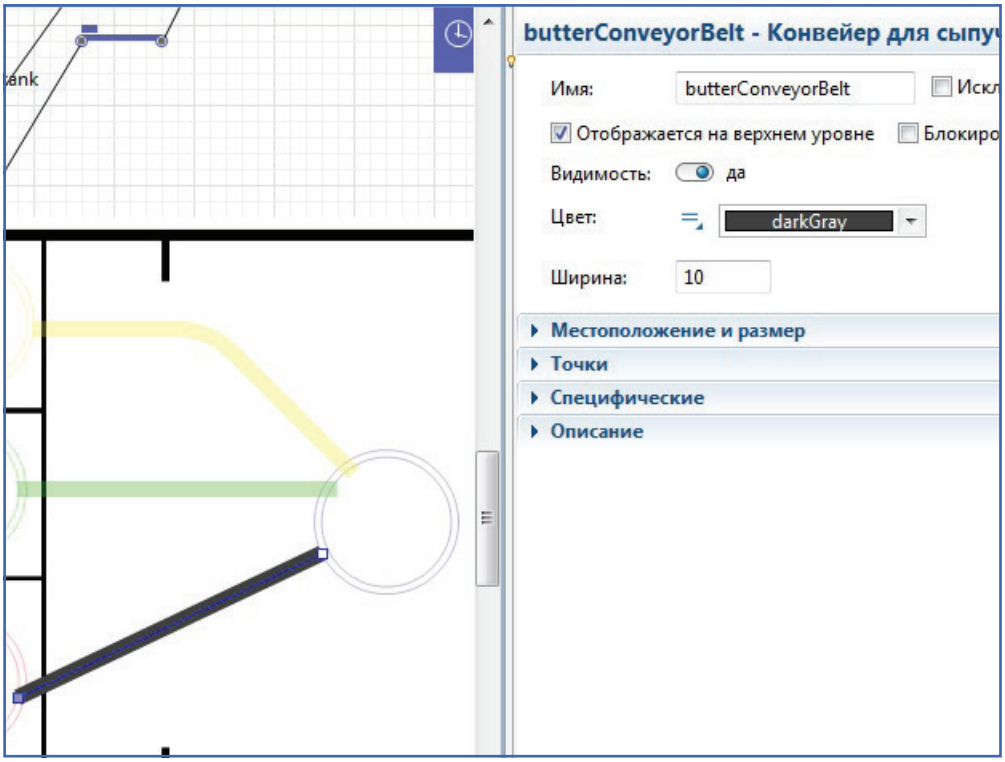

Рис. 2.45. Анимация конвейера масла

Шаг 7. Привязка анимации доставки ингредиентов до смесителя к трубопроводам и конвейеру

Для привязки анимации к объекту Pipeline, который моделирует трубопровод в модели, выделите объект pipeline_milk и перейдите в его свойства. В свойствах объекта в разделе Анимация выберите трубу (рис. 2.46).

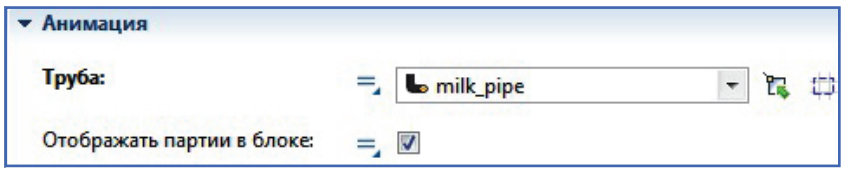

Рис. 2.46. Привязка анимации к молокопроводу

Аналогичным образом привяжите трубу к сахаропроводу и конвейер к конвейеру масла.

Далее поправьте свойства объекта конвейера масла, указав на то, что его длина теперь определяется длиной объекта Конвейер (рис. 2.47).

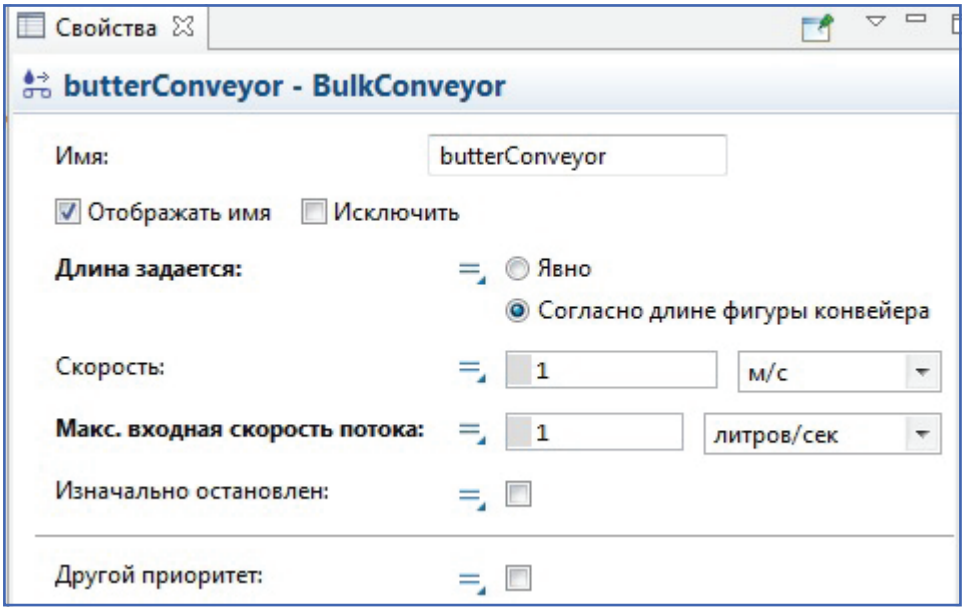

Рис. 2.47. Исправление задания длины конвейера

Шаг 8. Анимация реактора заморозки

Для анимации реактора также используется инструмент Резервуар, поэтому просто повторите действия шага 5.

Шаг 9. Анимация доставки смеси на заморозку

Доставка смеси на заморозку осуществляется конвейером. Анимация этого конвейера аналогична анимации конвейера масла. Поэтому просто повторите ту часть шага 6, где идет анимация конвейера, и шаг 7 по привязке его к объекту в модели.

Шаг 10. Анимация места накопления стаканчиков для мороженого

Для анимации узлов сети, которые могут принимать форму как точки, так и прямоугольника, используются инструменты из раздела Разметка пространства из библиотеки моделирования процессов. Акти-

вируйте элемент Прямоугольный узел и нарисуйте его в плане цеха (рис. 2.48).

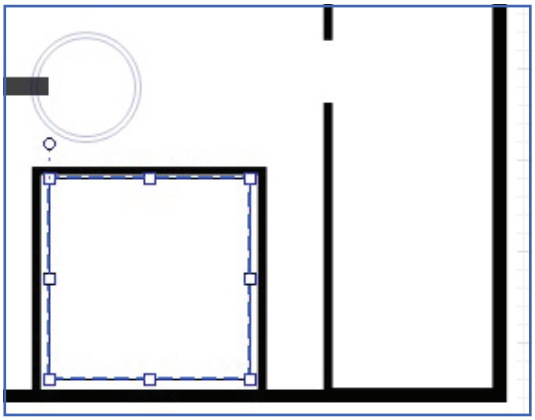

Рис. 2.48. Анимации накопления стаканчиков для мороженого

В свойствах объекта задайте его имя conenode.

В свойствах объекта cone_queue в пункте Место агентов выберите только что созданный узел сети (рис. 2.49).

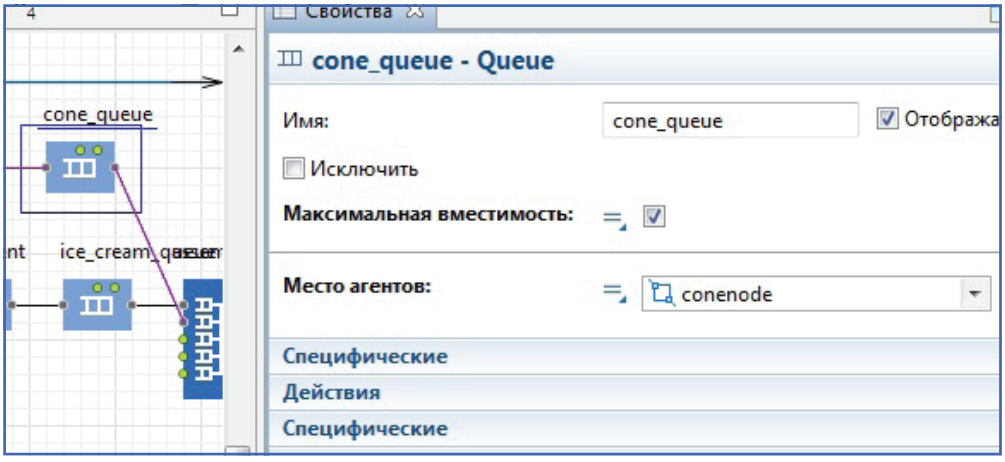

Рис. 2.49. Привязка анимации накопления стаканчиков для мороженого к объекту cone_queue

Шаг 11. Анимация сборки мороженого

Поскольку объект Assembler требует указания места входов агентов, места их сборки и места выхода собранного агента, то нужно создать

несколько точечных узлов сети для стаканчиков, мороженого и собранного мороженого.

Перетащите элемент Точечный узел из библиотеки моделирования процессов. Назовите его cone_point (рис. 2.50).

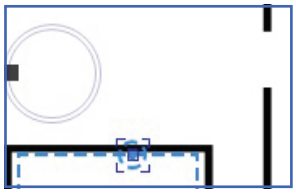

Рис. 2.50. Создание точечного узла для стаканчиков

Перетащите еще один точечный узел и назовите его ice_cream_point (рис. 2.51).

Рис. 2.51. Создание точечного узла для мороженого

Перетащите еще один точечный узел и назовите его ice_cream_cone_point (рис. 2.52).

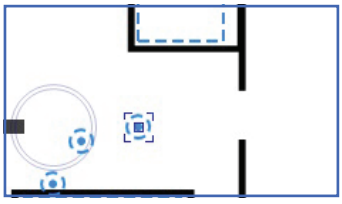

Рис. 2.52. Создание точечного узла для стаканчиков мороженого

Перетащите еще один точечный узел и назовите его robot_point (рис. 2.53).

Рис. 2.53. Создание точечного узла для робота-сборщика

Привяжите все созданные узлы в свойствах объекта Assembler в разделе Анимация (рис. 2.54).

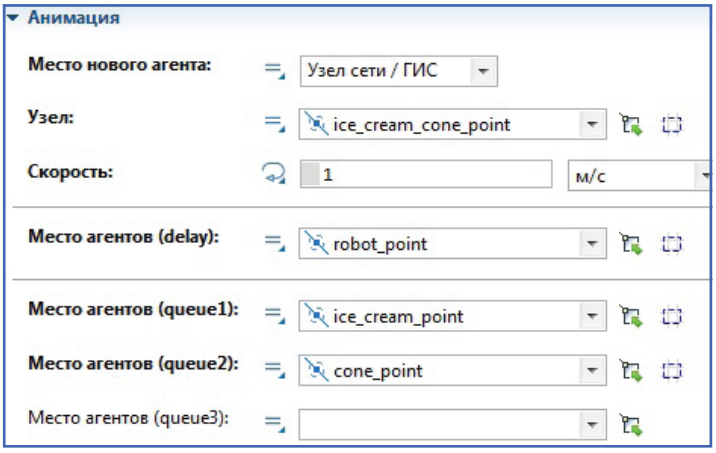

Рис. 2.54. Анимация процесса сборки

Шаг 12. Анимация упаковки мороженого

Нарисуйте прямоугольный узел сети в области, где будет проходить упаковка мороженого и назовите его packing_node. Нарисуйте к нему путь, задайте его имя ice_cream_cone_path. В разделе Внешний вид выберите тип Конвейер и задайте его ширину 30 см (рис. 2.55).

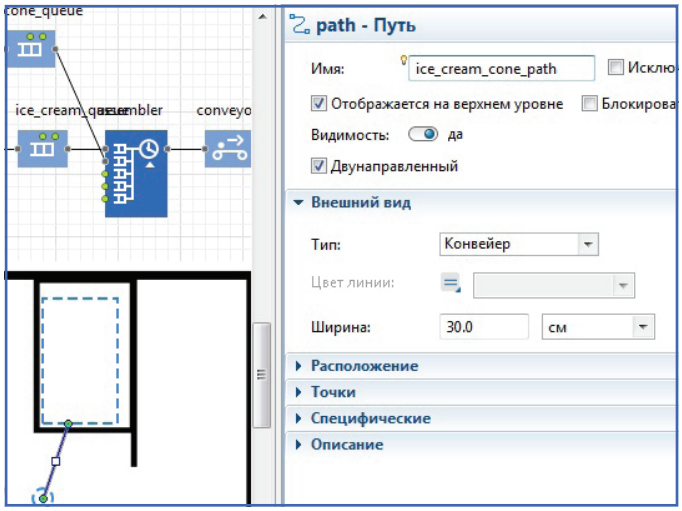

Рис. 2.55. Анимация процесса упаковки

Привяжите только что созданный путь к конвейеру доставки мороженого на упаковку (рис. 2.56).

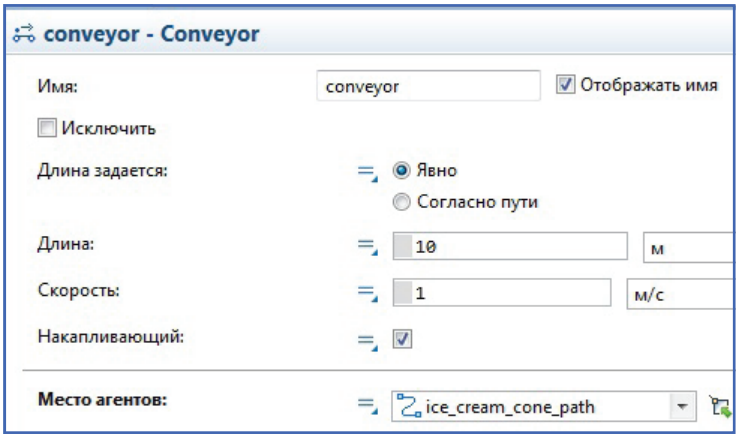

Рис. 2.56. Привязка анимации процесса упаковки

Шаг 13. Конечная модель без 3D-просмотра

Модель в конце должна получиться, как на рис. 2.57.

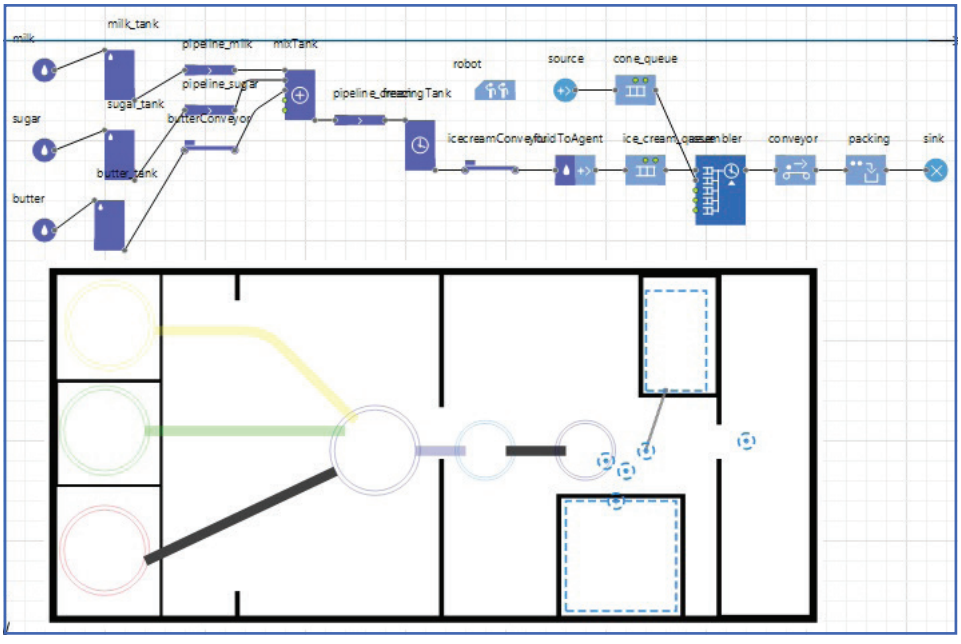

Рис. 2.57. Конечная модель

Шаг 14. Добавление окна 3D-просмотра

Инструмент 3D-окно позволяет просматривать работу модели в 3D-режиме (рис. 2.58). Все графические элементы модели, попавшие в область действия этого инструмента, становятся видны в своем объемном варианте, если таковой есть. Если элемент не имеет объемного варианта, то он остается в плоском варианте. Этот инструмент находится на панели Презентация. Перетащите его на рабочее поле модели и растяните его так, чтобы вся анимация модели попала в него.

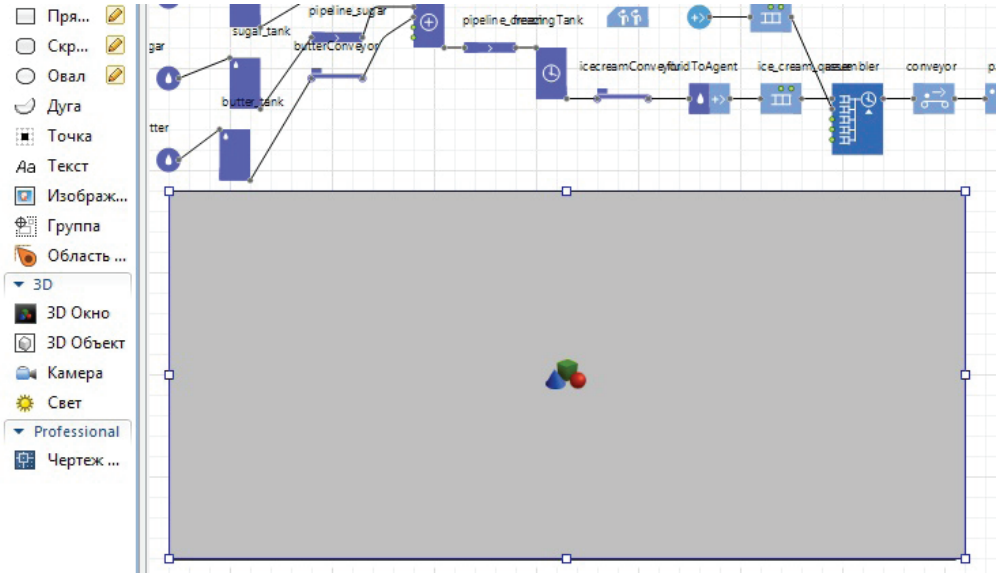

Рис. 2.58. Конечная модель с окном объемного просмотра

Шаг 15. Запуск модели

Запустите модель. Спустя некоторое время должно получиться примерно, как на рис. 2.59.

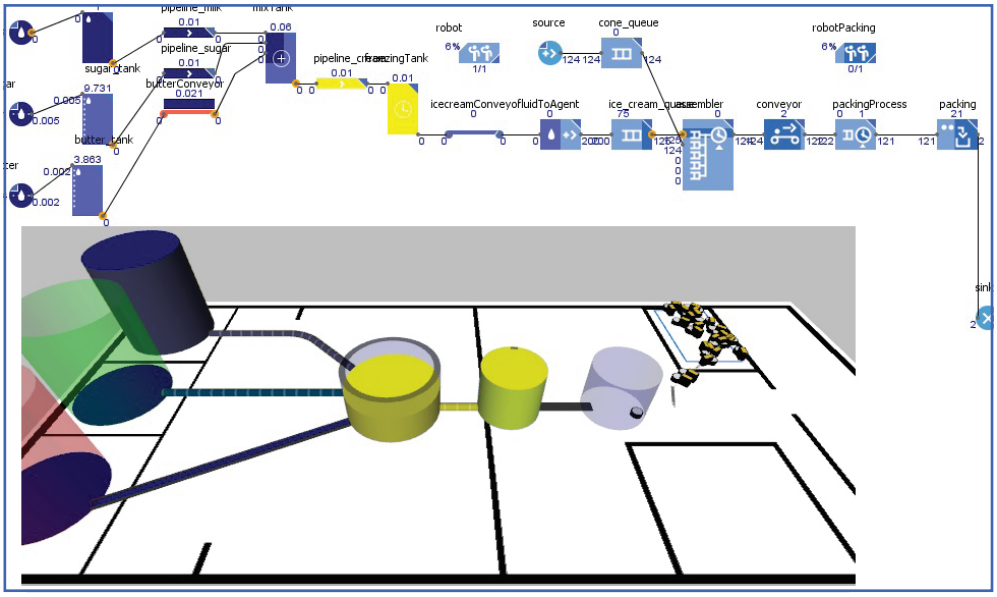

Рис. 2.59. Работа конечной модели с окном объемного просмотра

Самостоятельно

1. При прогоне модели выявите ее узкие места и предложите решения. Промоделируйте варианты решений.
2. Творческий проект. Создайте модель производственного процесса.

Лабораторная работа № 3
Агентное моделирование. Моделирование системы доставки мороженого

Задача

Промоделировать систему доставки мороженого с завода до складов и из складов до магазинов.

Решение

Этап 1. Разработка структуры модели

Шаг 1. Определение агентов в модели

Исходя из условий задачи, в модели должны действовать следующие агенты: завод, склады, магазины, заказы и грузовики, причем грузовики при заводе должны иметь большую грузоподъемность, чем грузовики при складах, следовательно, это разные агенты.

Шаг 2. Определение взаимодействия агентов в модели

Агент Завод имеет свои грузовики. В этом агенте производятся коробки мороженого. Агенты Склады имеют запас коробок мороженого, и, когда этот запас подходит к концу (пусть это будет менее 5 коробок), посылается заказ на завод. В агенте Завод ищется свободный грузовик, и он отправляется на склад с заказом. Агенты Магазины имеют свои запасы коробок мороженого и продают их. Как только запасы в магазине становятся меньше критического уровня, посылается заказ на склад. Агент Склад имеет свои грузовики и, получив заказ, ищет свободный грузовик у себя. Далее отправляет его с заказом в магазин, из которого поступил заказ.

Этап 2. Создание агентов в модели

Шаг 1. Создание агента Завод

Создайте новую модель и задайте ее имя и единицы модельного времени минуты (рис. 3.1).

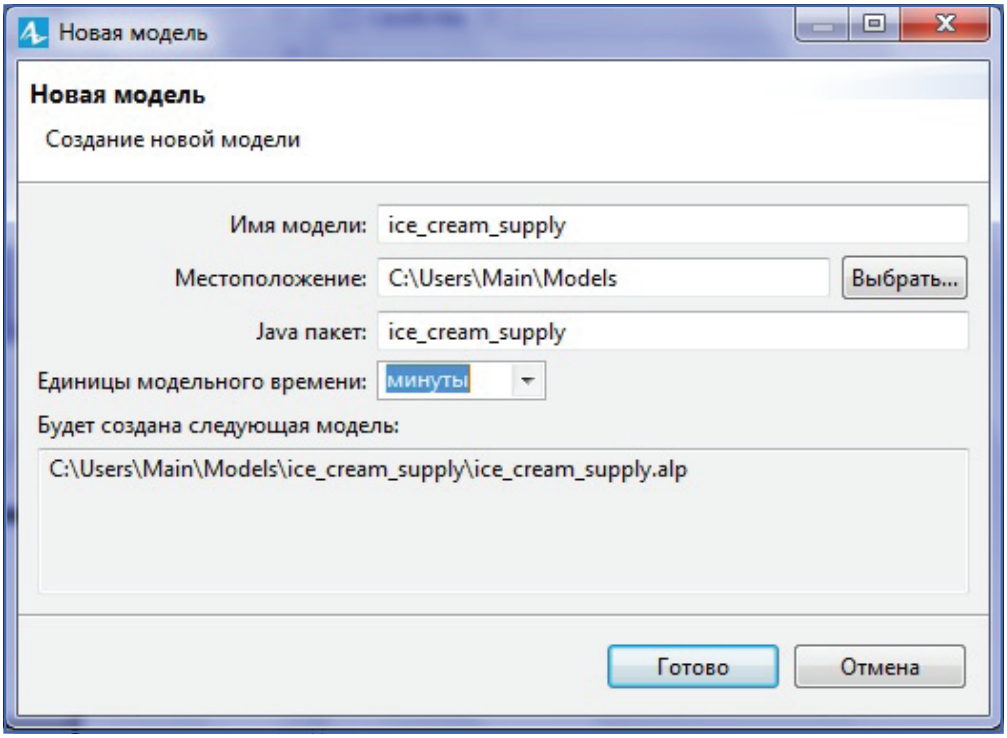

Рис. 3.1. Окно создания новой модели

Перейдите в библиотеку Агентное моделирование. Перетащите из нее объект Агент на рабочее поле модели. Откроется окно Мастера создания нового агента. На первом шаге Мастера выберите то, что создается. Поскольку агент Завод будет один в модели, то создаем единственного агента.

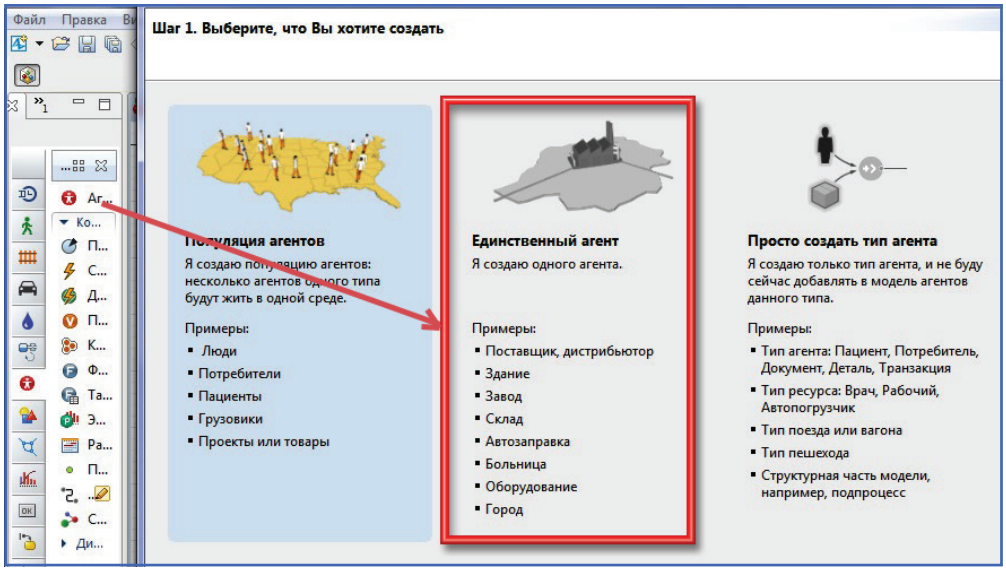

Рис. 3.2. Первый шаг Мастера создания агента Завод

На втором шаге Мастера задайте имя агента plant (рис. 3.3).

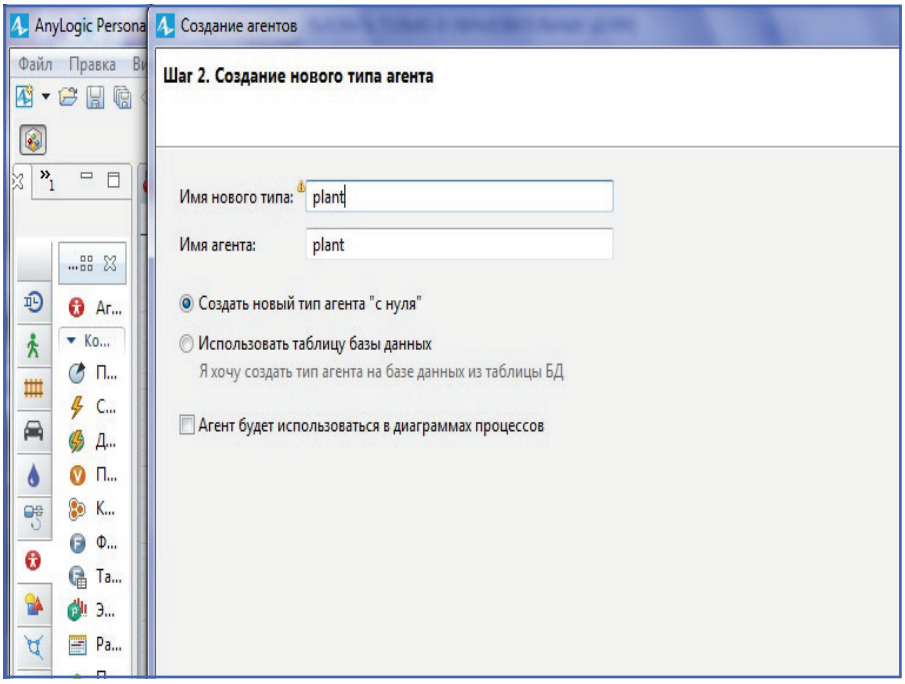

Рис. 3.3. Второй шаг Мастера создания агента Завод

На третьем шаге Мастера выберите анимацию агента (рис. 3.4).

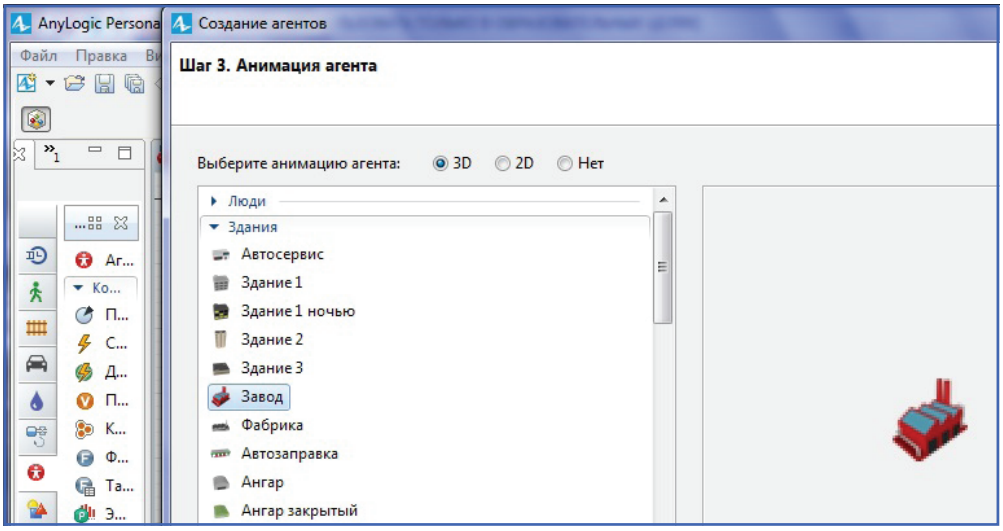

Рис. 3.4. Третий шаг Мастера создания агента Завод

На четвертом шаге Мастера создайте параметр агента, который будет содержать количество произведенных коробок (рис. 3.5).

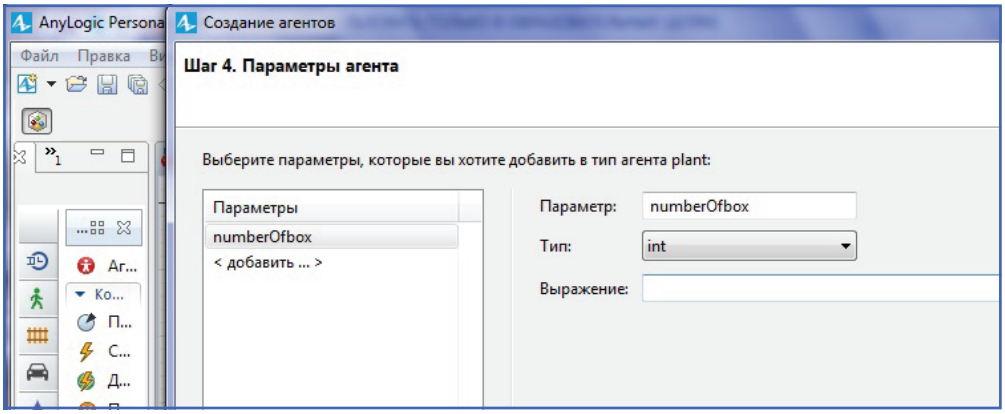

Рис. 3.5. Четвертый шаг Мастера создания агента Завод

На пятом шаге Мастера задайте среду обитания агентов модели (рис. 3.6).

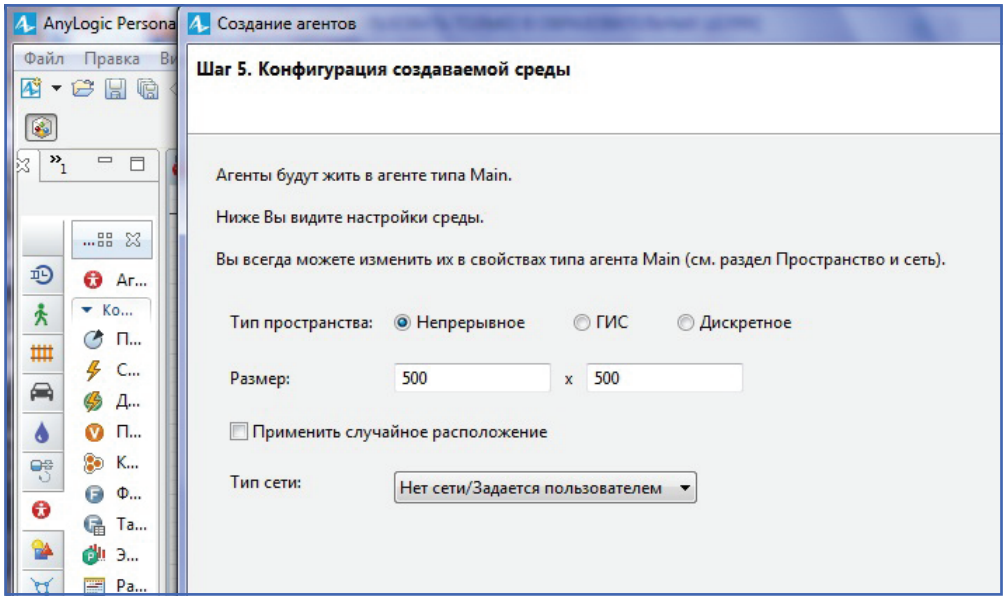

Рис. 3.6. Пятый шаг Мастера создания агента Завод

После окончания работы Мастера в модели появится новый агент plant (рис. 3.7).

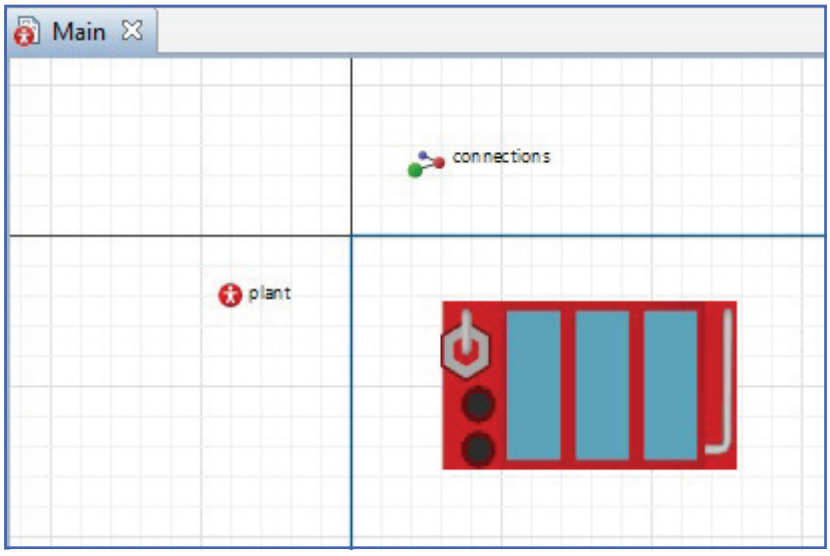

Рис. 3.7. Агент Завод в модели

Шаг 2. Задание агента грузовик при заводе

Поскольку грузовиков будет несколько, то создаем популяцию агентов. Перетащите объект Агент на рабочее поле и на первом шаге Мастера выберите Популяция агентов (рис. 3.8).

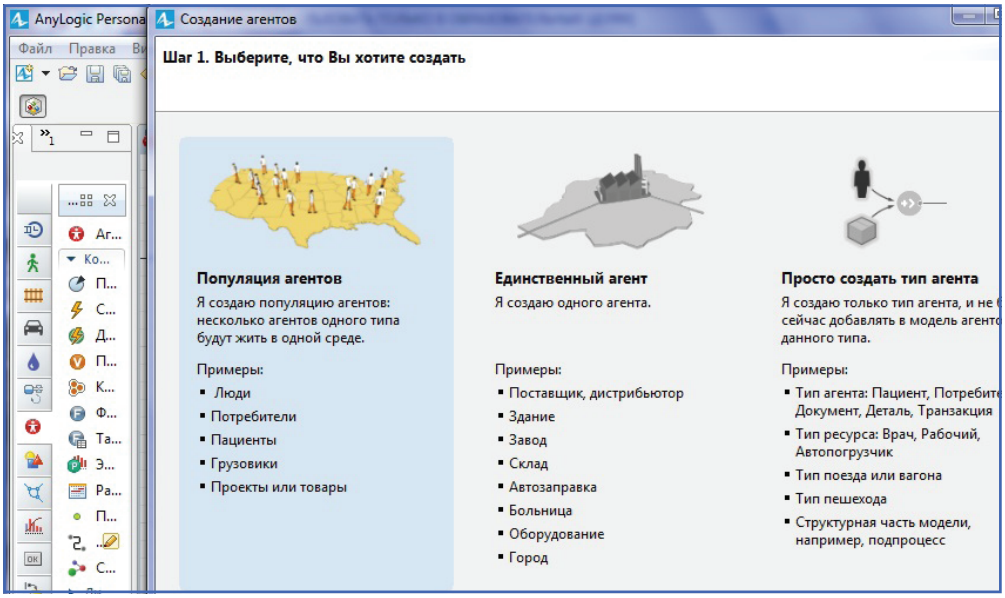

Рис. 3.8. Первый шаг Мастера по созданию популяции агентов Грузовик при заводе

Поскольку популяцию агентов можно создать из уже имеющихся агентов, то на втором шаге Мастера будет предложен выбор между созданием популяции новых агентов и популяции уже имеющихся. Выберите популяцию новых агентов.

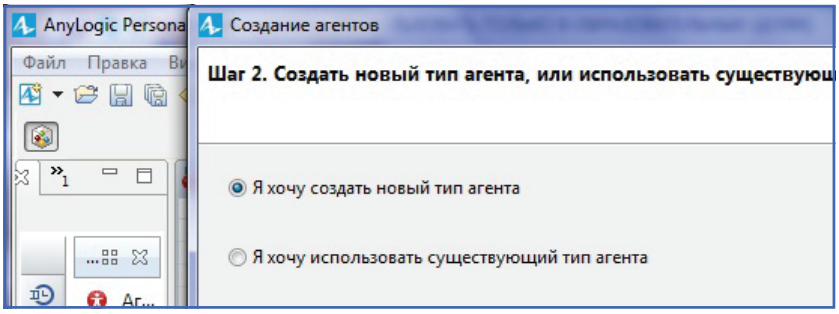

Рис. 3.9. Второй шаг Мастера по созданию популяции агентов Грузовик при заводе

На следующем шаге Мастера задайте имя агентов (рис. 3.10).

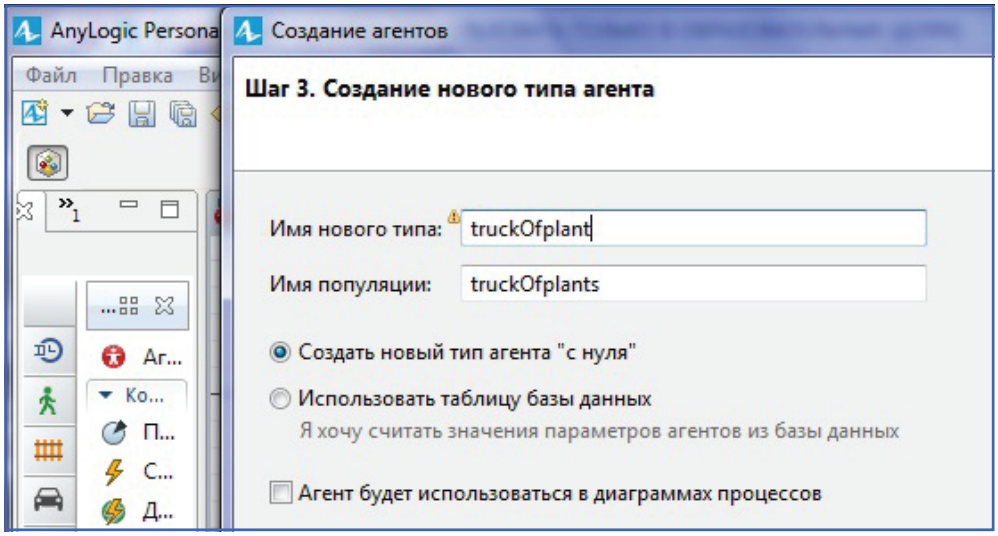

Рис. 3.10. Третий шаг Мастера по созданию популяции агентов Грузовик при заводе

Далее выберите анимацию агента (рис. 3.11).

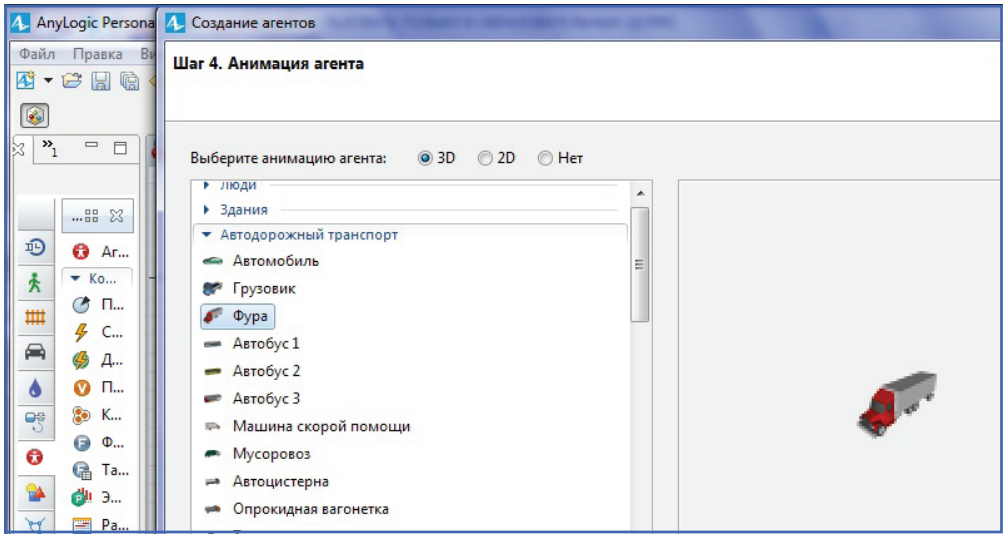

Рис. 3.11. Четвертый шаг Мастера по созданию популяции агентов Грузовик при заводе

На следующем шаге просто нажмите Далее, поскольку параметры Агенты будут созданы позже.

На шестом шаге Мастера задайте объем популяции агентов — 3 грузовика (рис. 3.12).

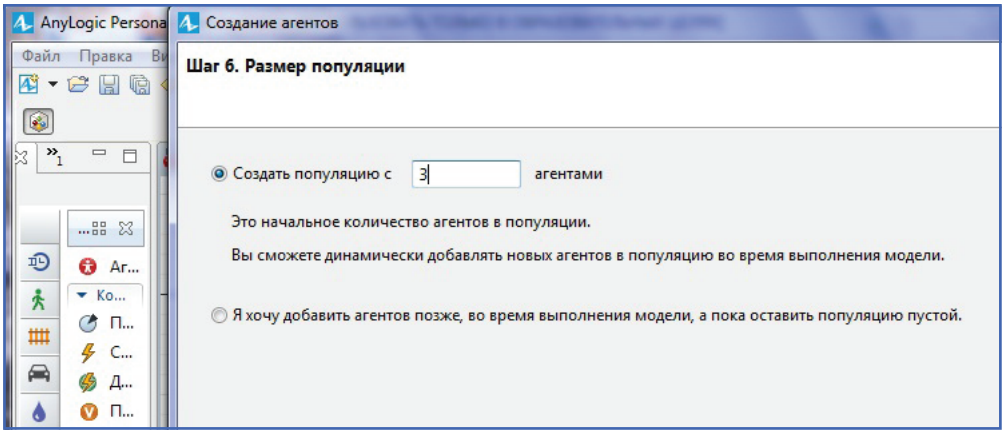

Рис. 3.12. Шестой шаг Мастера по созданию популяции агентов Грузовик при заводе

На следующем шаге Мастера ничего не меняйте. После окончания работы Мастера в модели появится популяция агентов truckOfplants, которая содержит 3 грузовика при заводе (рис. 3.13).

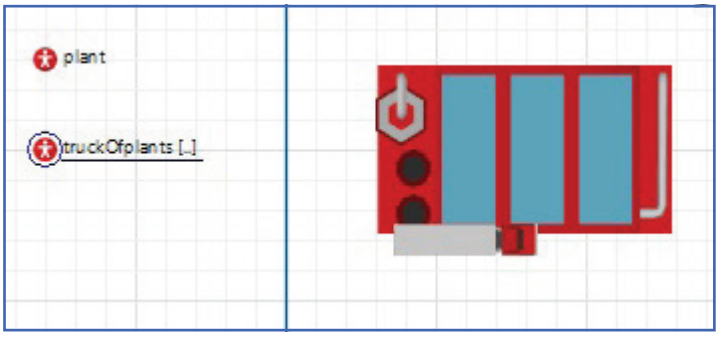

Рис. 3.13. Популяция агентов Грузовик при заводе в модели

В свойствах популяции будет указано имя популяции и ее начальный объем (рис. 3.14).

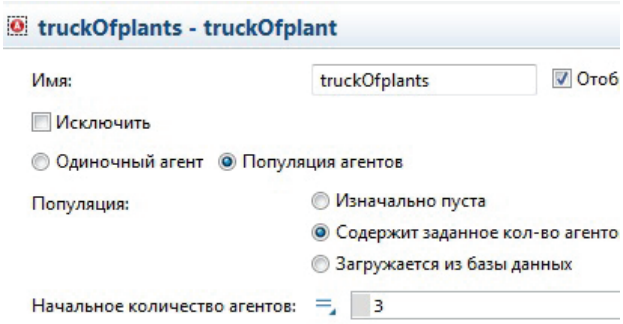

Рис. 3.14. Свойства популяция агентов Грузовик при заводе в модели

Шаг 3. Добавление популяции агентов Заказ

Повторите действия шага 2 по созданию популяции агентов. Дайте имя агентам order. Агент не содержит анимации и параметров. Популяция изначально создается пустой. В конце работы Мастера в модели должны быть один агент и две популяции (рис. 3.15).

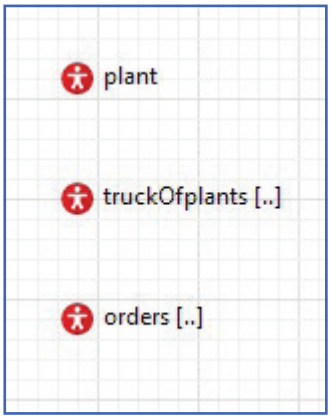

Рис. 3.15. Популяция агентов Заказ в модели

Шаг 4. Добавление популяции агентов Склады

Создайте популяцию агентов storage с двумя агентами в ней (рис. 3.16). Агенты не содержат параметров. В качестве анимации выберите изображения склада.

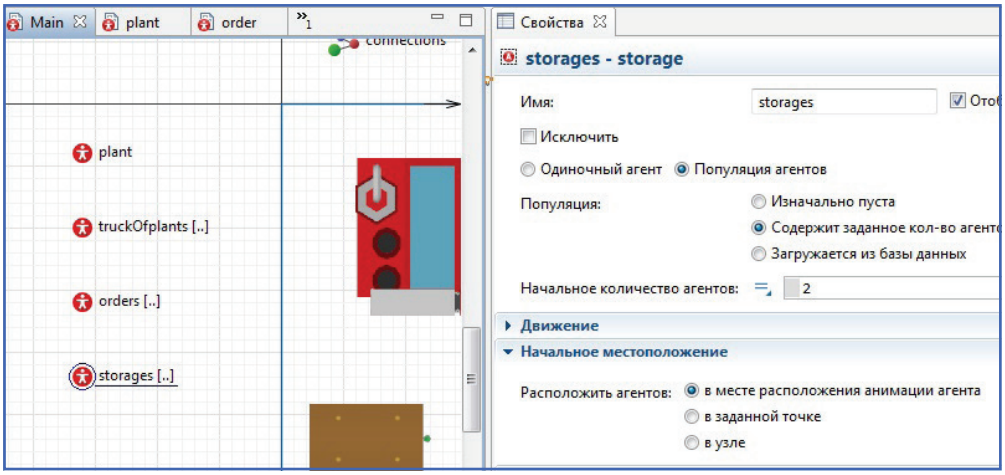

Рис. 3.16. Популяция агентов Склады в модели

Шаг 5. Создание популяции агентов Магазины

Создайте популяцию агентов shops с четырьмя агентами в ней (рис. 3.17). Агенты не содержат параметров. В качестве анимации выберите изображения магазина.

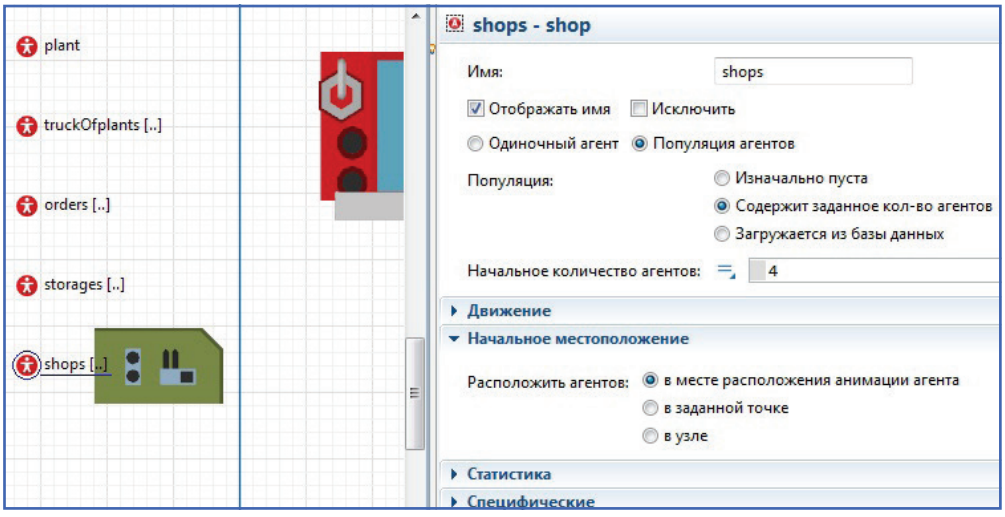

Рис. 3.17. Популяция агентов Магазины в модели

Шаг 6. Создание популяции агентов Грузовики при складах

Создайте популяцию агентов truckOfstorages с десятью агентами в ней (рис. 3.18). Агенты не содержат параметров. В качестве анимации выберите изображение маленького грузовика.

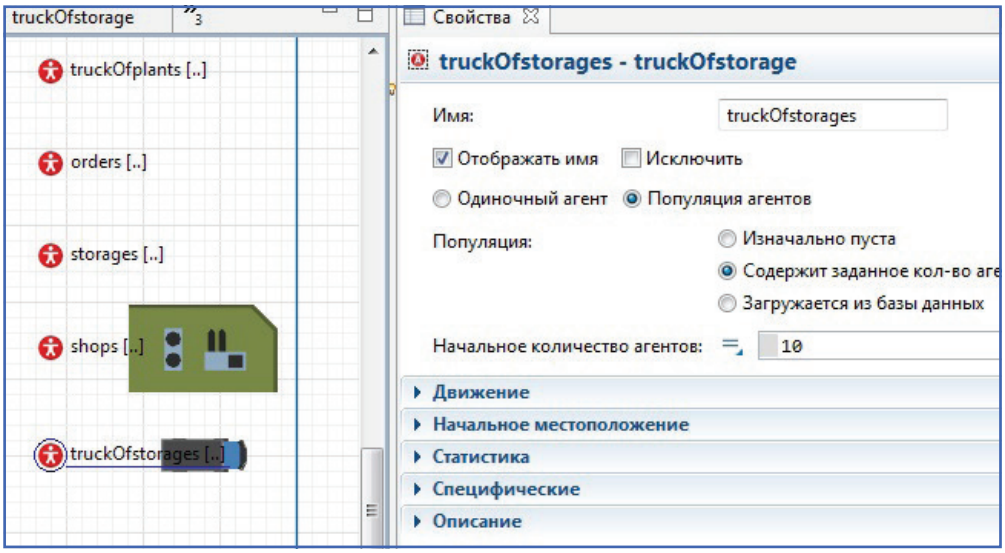

Рис. 3.18. Популяция агентов Грузовики при складах в модели

Шаг 7. Расстановка складов в модели

Для расстановки складов в модели используется функция, которая задает местоположение агентов из популяции storages. Функция реализована с помощью инструмента Диаграмма действий. Элементы этого инструмента находятся на отдельной вкладке Диаграмма действий в палитре. Перейдите на нее и перетащите элемент Начало диаграммы на рабочее поле. В свойствах элемента задайте имя функции setStorage и тип функции (рис. 3.19).

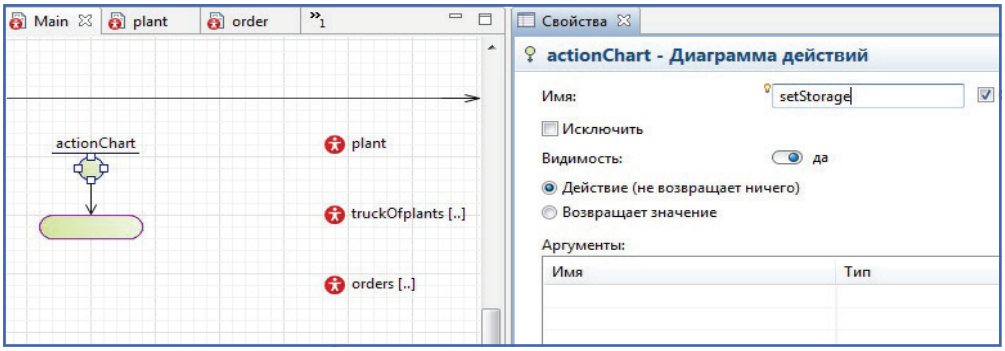

Рис. 3.19. Создание функции расстановки складов. Первый шаг

Далее вставьте два блока для локальных переменных *x* и *y*. В свойствах блока задайте имя переменных, их тип — double и начальные значения (рис. 3.20).

Затем перетащите блок цикла со счетчиком и в его свойствах задайте изменения счетчика от 0 до количества элементов в популяции. Для создания тела цикла перетащите блок Код и в его свойствах напишите код цикла (рис. 3.20).

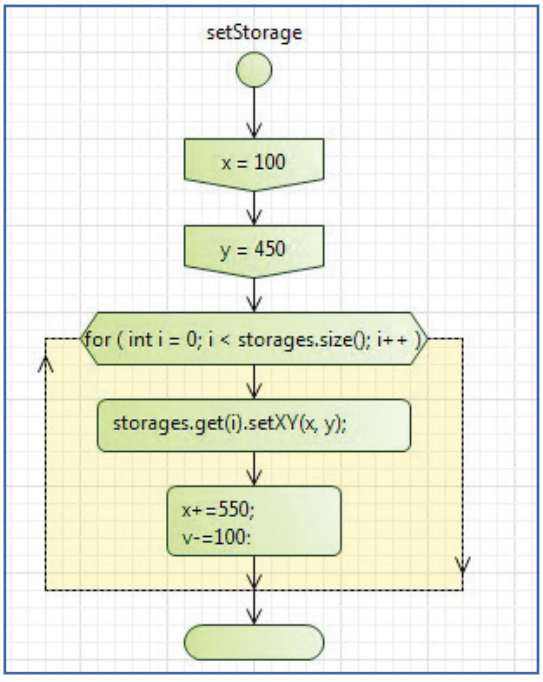

Рис. 3.20. Конечный вид функции расстановки складов

В цикле идет обращение к каждому элементу коллекции storages с помощью метода get, у которого в качестве параметра принимается номер элемента коллекции. Затем каждый элемент коллекции помещается в точку с координатами *x* и *y* с помощью метода setXY().

Для запуска функции перейдите в свойства агента Main и в разделе Действия агента в пункте При запуске напишите код вызова функции (рис. 3.21).

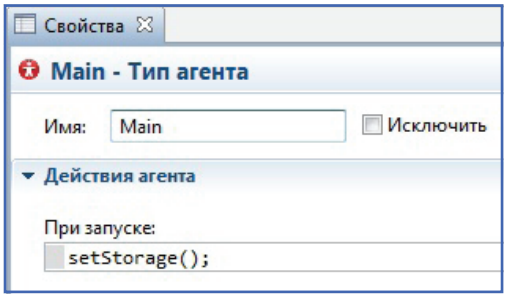

Рис. 3.21. Вызов функции расстановки складов

Проверьте работу функции, запустив модель на выполнение (рис. 3.22) (склады должны быть расставлены).

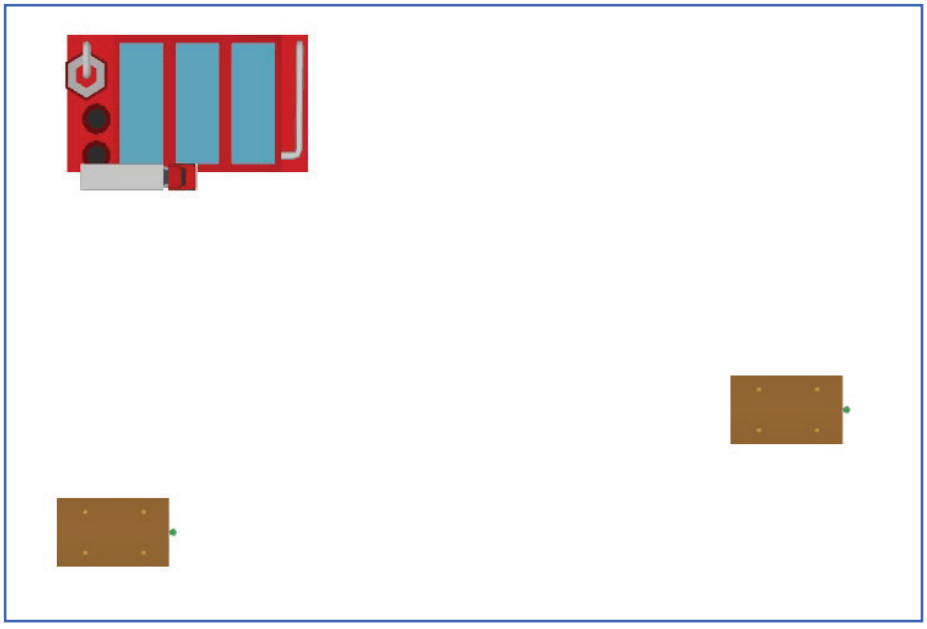

Рис. 3.22. Результат работы функции расстановки складов

Шаг 8. Расстановка магазинов в модели

В функцию setStorage добавьте следующий код (рис. 3.23).

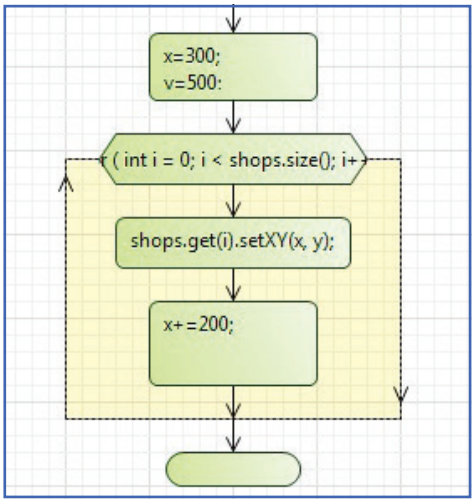

Рис. 3.23. Расширение функции setStorage

Проверьте работу функции, запустив модель (магазины должны быть расставлены).

Этап 3. Задание взаимодействия агентов Завод и Склады

Шаг 1. Задание поведения агента Завод

Агент Завод (рис. 3.24) производит коробки мороженого. Для реализации этого поведения создайте параметр, который будет содержать количество произведенных коробок, и событие, которое с заданной интенсивностью увеличивает значение параметра (рис. 3.25).

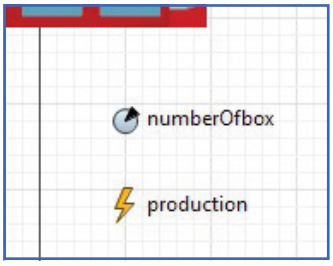

Рис. 3.24. Элементы агента Завод

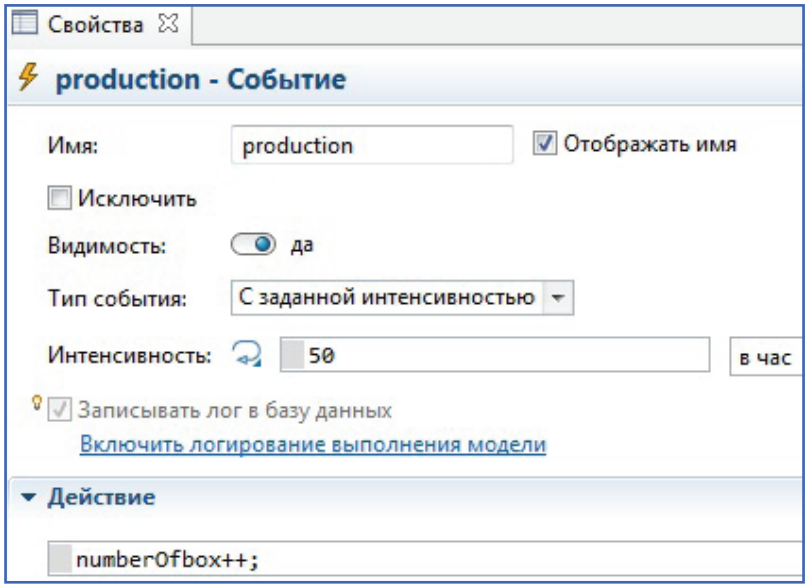

Рис. 3.25. Задание поведения агента Завод

Проверьте правильность поведения агента Завод, запустив модель. В эксперименте выберите агент plant (рис. 3.26) и понаблюдайте поведение агента в модельном времени (рис. 3.27).

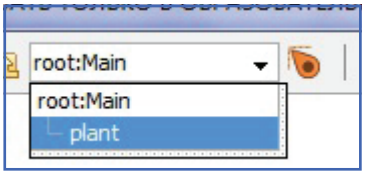

Рис. 3.26. Переход на агент Завод

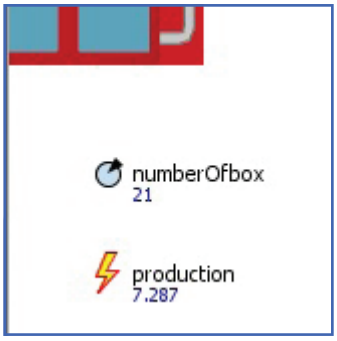

Рис. 3.27. Поведение агента Завод в модельном времени

Шаг 2. Задание поведения агента Заказ

Агент Заказ содержит в себе количество коробок и заказчика (рис. 3.28). Для хранения количества коробок создайте в агенте order параметр amount целого типа.

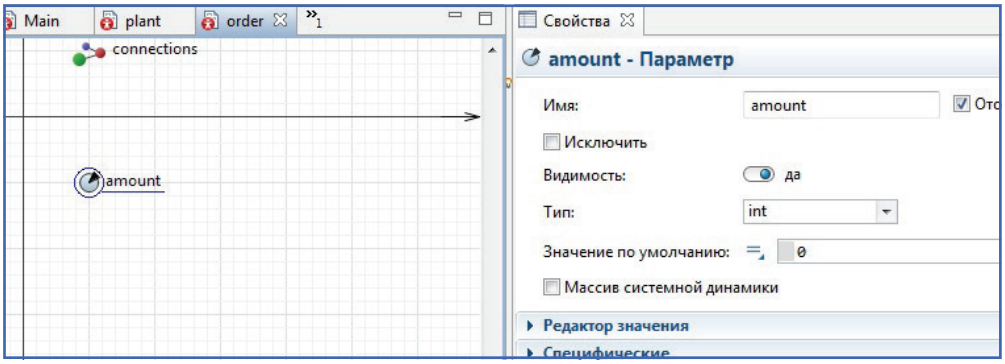

Рис. 3.28. Параметр агента Заказ для хранения количества коробок в заказе

Для хранения заказчика-склад создайте параметр storage с типом данных storage (рис. 3.29).

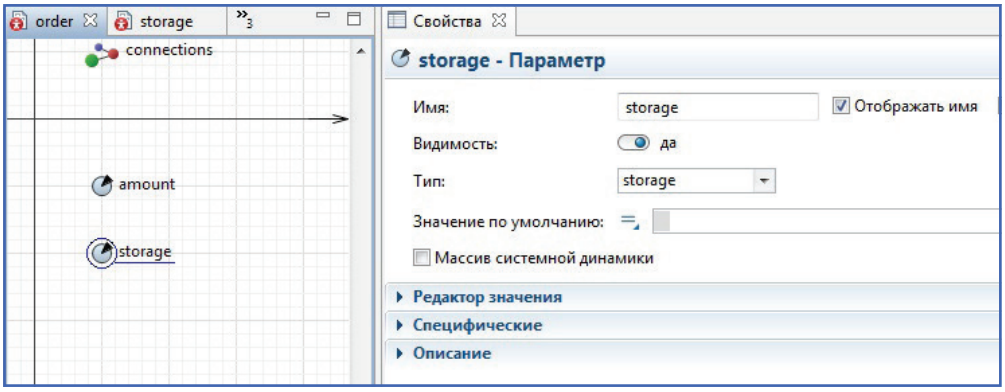

Рис. 3.29. Параметр агента Заказ для хранения заказчика-склад

Для хранения заказчика-магазин создайте параметр shop с типом данных shop.

Шаг 3. Задание поведения агента Грузовик при заводе

Агент Грузовик при заводе получает заказ от склада и едет туда с заказом, а выполнив заказ, возвращается обратно на завод. Это поведение реализовано с помощью переменной order типа order, которая хранит заказ, пришедший со склада, и диаграммы поведения.

Для создания переменной перетащите объект Переменная из агентной библиотеки и задайте ее имя и тип (рис. 3.30).

Инструменты для создания диаграммы поведения есть как в агентной библиотеке, так и на отдельной вкладке Диаграмма поведения. Воспользуйтесь любым источником и, перетаскивая объекты диаграммы, постройте диаграмму поведения, как на рис. 3.30. Переходы в этой диаграмме пока просто нарисуйте и выберите их тип, действия к ним привяжем позже. Переход от состояния atPlant в состояние movingToStorage имеет тип При получении сообщения, остальные переходы — по прибытии агента.

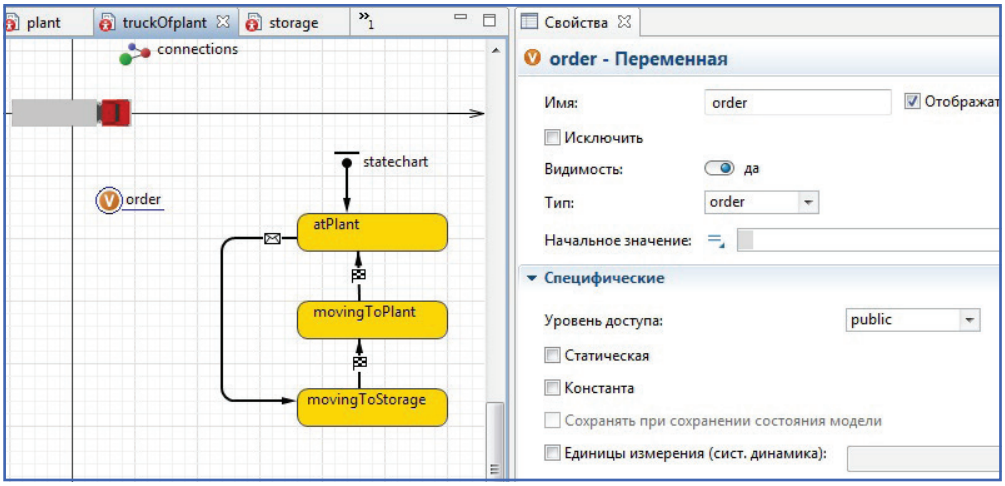

Рис. 3.30. Элементы агента Грузовик при заводе

Эта диаграмма имеет три состояния: atPlant, movingToPlant, movingToStorage. Первое состояние atPlant показывает, что грузовик находится на заводе, то есть он свободен от заказов на текущий момент времени. Далее, получив сообщение от склада, грузовик переходит в состояние movingToStorage. Приехав на склад, грузовик возвращается на завод, чему соответствует состояние movingToPlant.

Шаг 4. Создание функции поиска свободного грузовика при заводе

В агенте Завод нужно осуществлять поиск свободного грузовика для выполнения заказа (рис. 3.31). Эту задачу реализует функция findFreeTruck. Функция возвращает грузовик и не имеет параметров. Создайте функцию с помощью инструмента Диаграмма действий.

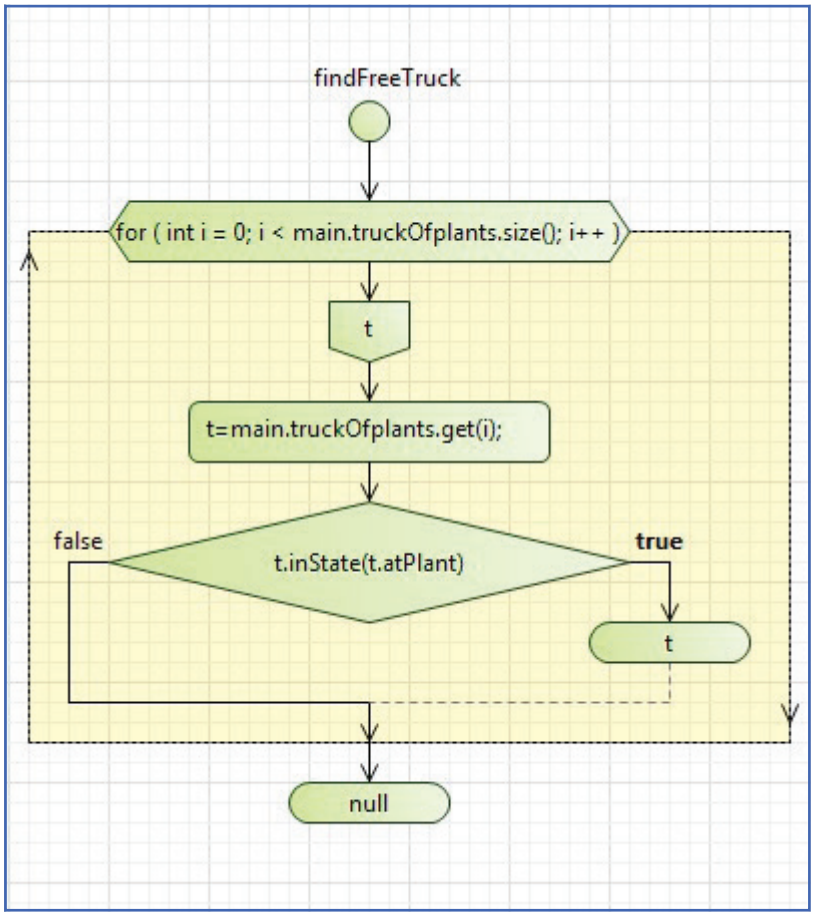

Рис. 3.31. Функция поиска свободного грузовика на заводе

В этой функции организован цикл, проходящий по всей популяции грузовиков. В цикле создается локальная переменная *t* типа truckOfplant. Далее, в эту переменную записывается очередной грузовик из коллекции. В условном операторе проверяется состояние грузовика: если оно atPlant, то значит, что грузовик свободен и он возвраща-

ется из функции; если в другом состоянии, то переходим к следующему грузовику. Если не найдется ни одного грузовика в состоянии atPlant, то функция вернет значение null.

Шаг 5. Задание поведения агента Склад

Агент Склад имеет свои запасы коробок, из которых он отпускает заказы в магазины, и свои грузовики, которые выполняют заказы из магазинов. Также агент посылает заказы на завод, если запасы в агенте стали меньше 30 коробок. Создайте параметр amountOfbox для хранения количества коробок в запасе в агенте (рис. 3.32).

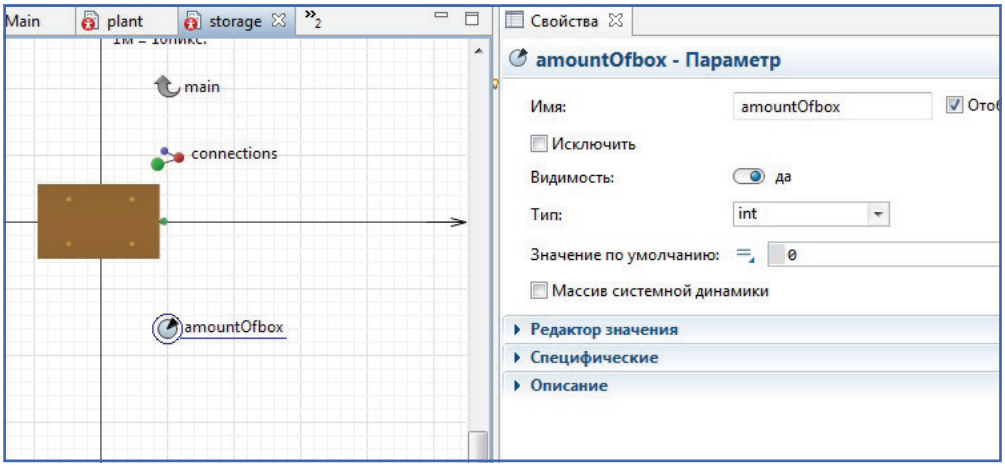

Рис. 3.32. Параметр агента Склад для хранения количества коробок в запасе

Отправку заказа на завод реализует функция request. Для ее создания используйте инструмент Функция из агентной библиотеки (рис. 3.33). Перетащите элемент Функция на рабочее поле. В свойствах объекта задайте имя функции, тип возвращаемого значения и список параметров. Функция request просто осуществляет действие и ничего не возвращает. Функция имеет один параметр amount целого типа, который содержит количество коробок в заказе. В разделе свойств Тело функции введите код функции.

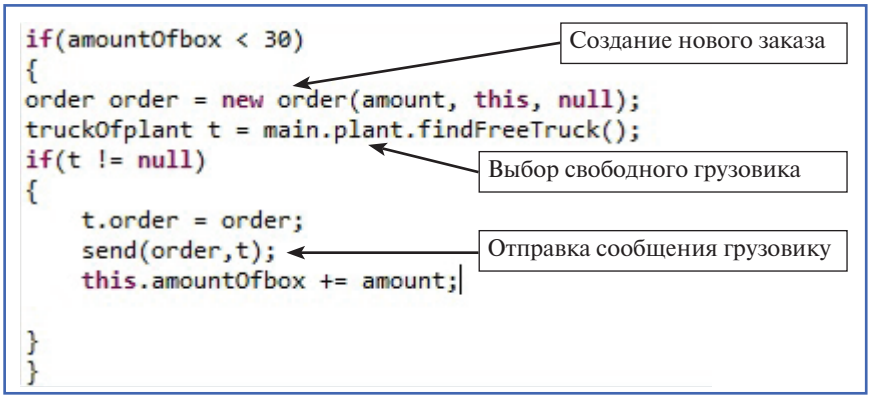

Рис. 3.33. Функция обработки и отправки заказа со склада на завод

Функция начинает свою работу, если количество коробок в запасе стало меньше 30. Сначала создается новый заказ, в параметрах которого передается количество коробок в заказе и склад-отправитель. Далее в переменную *t* сохраняется свободный грузовик на заводе. Если таковой найдется, то в его параметр order записывается передаваемый заказ и отправляется сообщение типа order. Получив это сообщение, грузовик начнет движение на склад. Далее идет обновление значения параметра amountOfbox.

Запускать функцию формирования заказа будет событие requestBox, которое будет запускаться с интенсивностью 10 раз в час (рис. 3.34).

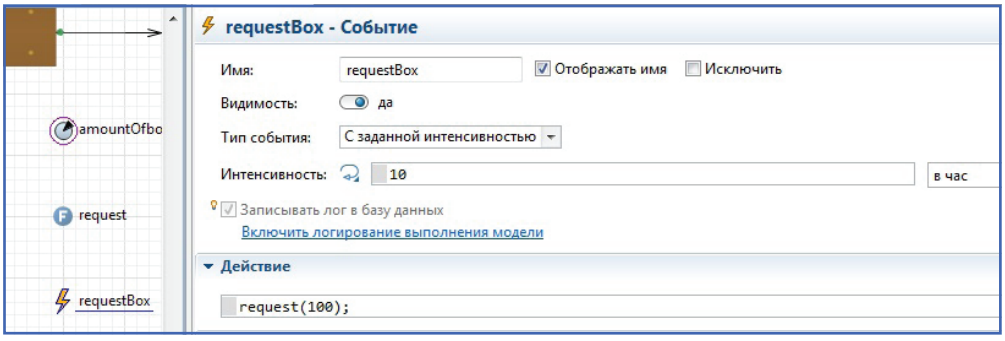

Рис. 3.34. Событие, вызывающее функцию обработки и отправки заказа со склада на завод

Шаг 6. Добавление действий в переходы диаграммы состоянии агента Грузовик при заводе

Настало время «оживить» диаграмму состояния грузовиков, чтобы они начали свое движение в модели. Для этого в свойствах перехода из состояния atPlant в состояние movingTostorage добавьте действие: moveTo() (рис. 3.35). Эта функция задает движение агента в точку, координаты которой заданы в параметрах. В качестве координат взяты координаты склада, пославшего заказ.

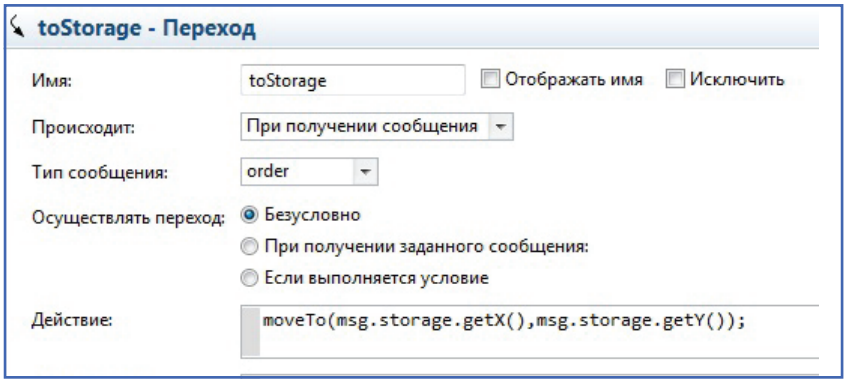

Рис. 3.35. Действия перехода из состояния atPlant в состояние movingTostorage

В действия перехода из состояния movingTostorage в состояние movingToplant добавьте код, как на рис. 3.36.

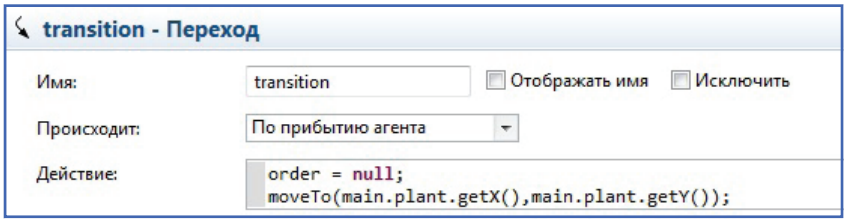

Рис. 3.36. Действия перехода из состояния movingTostorage в состояние movingToplant

Шаг 7. Проверка работоспособности модели

Запустите модель (рис. 3.37) и понаблюдайте за движением грузовиков от завода на склад и обратно. Если движение грузовиков будет

слишком быстрое, уменьшите скорость агента truckOfplant в свойствах агента в разделе Движение.

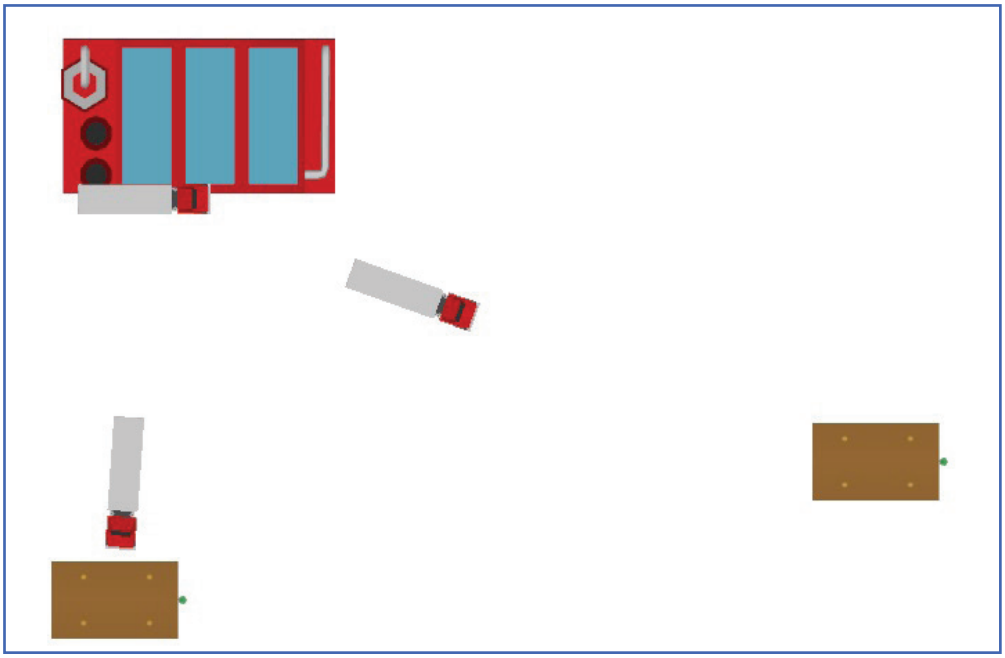

Рис. 3.37. Работа модели

Этап 4. Задание взаимодействия агентов Склады и Магазины

Шаг 1. Задание поведения агента Грузовик при складе

По сути, поведение агента Грузовик при складе (рис. 3.38) такое же, как агента Грузовик при заводе, разница — в типе получаемого заказа. Здесь агент получает заказ от магазинов, а не от склада. Кроме того, поскольку склада два, нужен параметр в агенте Грузовик, который будет содержать склад-хозяин грузовика. Поэтому повторите все действия шага 3 предыдущего этапа, но уже для агента Грузовик при складе и добавьте к нему еще один параметр. В конечном итоге должен получиться агент с набором элементов, как на рис. 3.38.

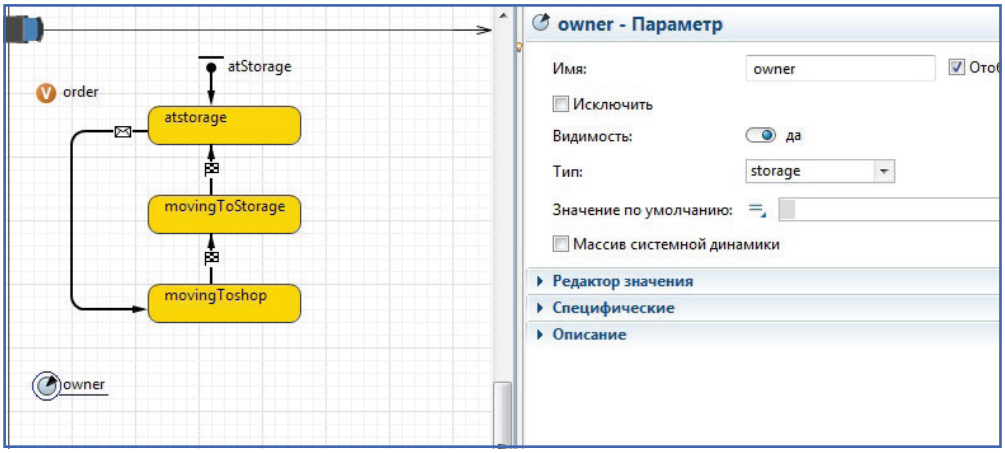

Рис. 3.38. Элементы агента Грузовик при складе

Шаг 2. Привязка грузовиков к складам

Добавьте в функцию setStorage код, как на рис. 3.39.

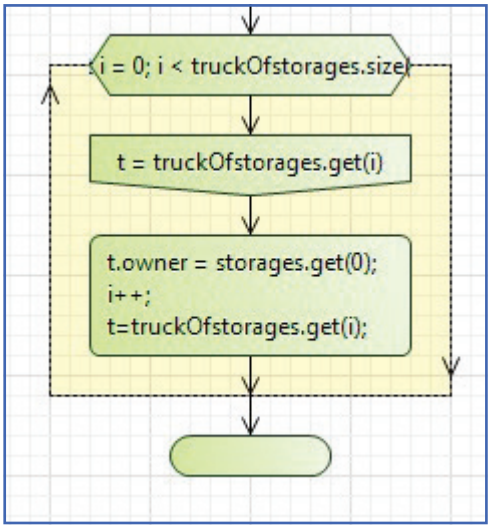

Рис. 3.39. Расширение функции setStorage, привязывающей грузовики к складам

Проверьте работу функции, запустив модель и перейдя в агент Склады (рис. 3.40). В параметре owner будет отображен склад-хозяин грузовика.

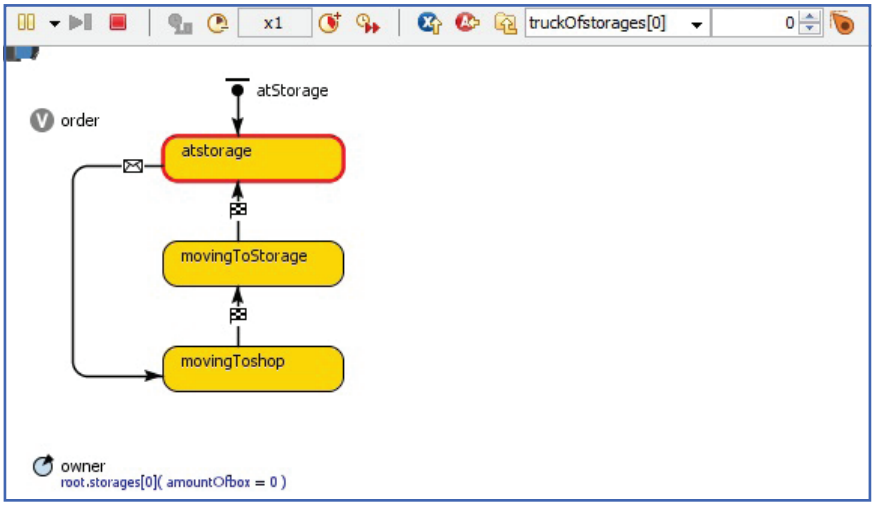

Рис. 3.40. Проверка работы функции setStorage, привязывающей грузовики к складам

Шаг 3. Добавление функции поиска свободного грузовика на складе в агент Склад

В агенте Склад нужно добавить функцию поиска свободного грузовика. По сути своей эта функция аналогична функции поиска свободного грузовика на заводе, поэтому повторите действия шага 4 предыдущего этапа. Конечный код функции приведен на рис. 3.41.

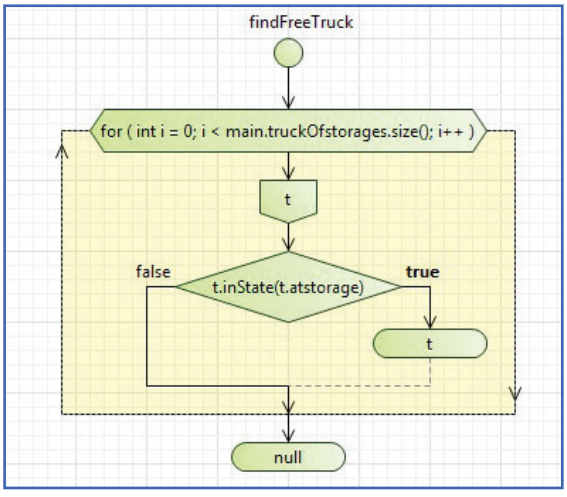

Рис. 3.41. Функция поиска свободного грузовика на складе

Шаг 4. Задание поведения агента Магазин

Агент Магазин продает коробки мороженого и заказывает новые на склады. Агент имеет запасы коробок. Для хранения количества коробок в запасе создайте параметр amountBox целочисленного типа (рис. 3.42).

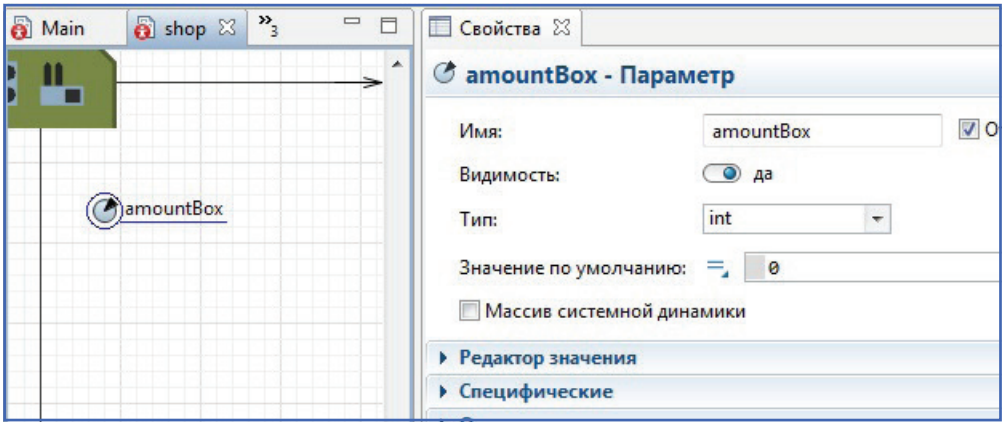

Рис. 3.42. Параметр агента Магазин

Для формирования заказов на склад создайте функцию requestShop, которая в качестве параметра принимает количество коробок в заказе, но сама ничего не возвращает. Ее код приведен на рис. 3.43.

```
if(amountBox < 5)
{
order order = new order(amount,null, this);
truckOfstorage t =null;
storage s=this.owner;
t= s.findFreeTruck();
if(t != null )
{
    t.order = order;
    send(order,t);
    this.amountBox += amount;
    s.amountOfbox -= amount;
}

}
```

Рис. 3.43. Функция составления и отправки заказа агента Магазин

В целом эта функция аналогична функции request. Только здесь в переменную *s* сохраняется склад-хозяин магазина и учитывается изменение количества коробок как в магазине, так и на складе.

Функция requestShop запускается событием OrderBox, которое запускается с заданной интенсивностью (рис. 3.44).

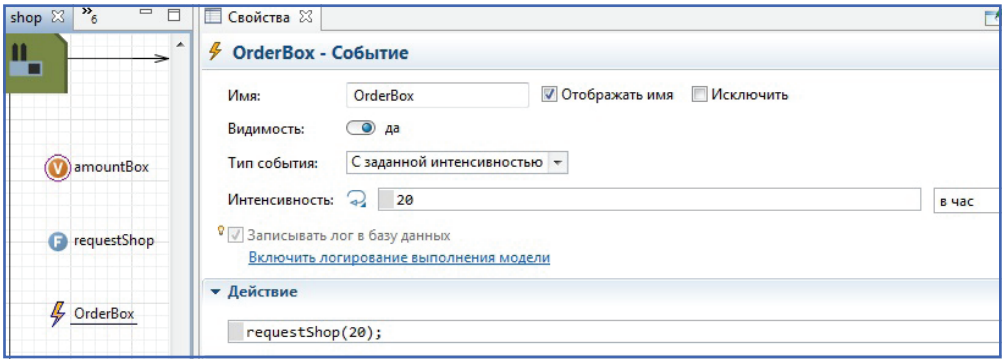

Рис. 3.44. Запуск функции формирования заказов магазина

Для моделирования продаж создайте событие sold, которое с заданной интенсивностью будет уменьшать коробки в запасах магазина (рис. 3.45).

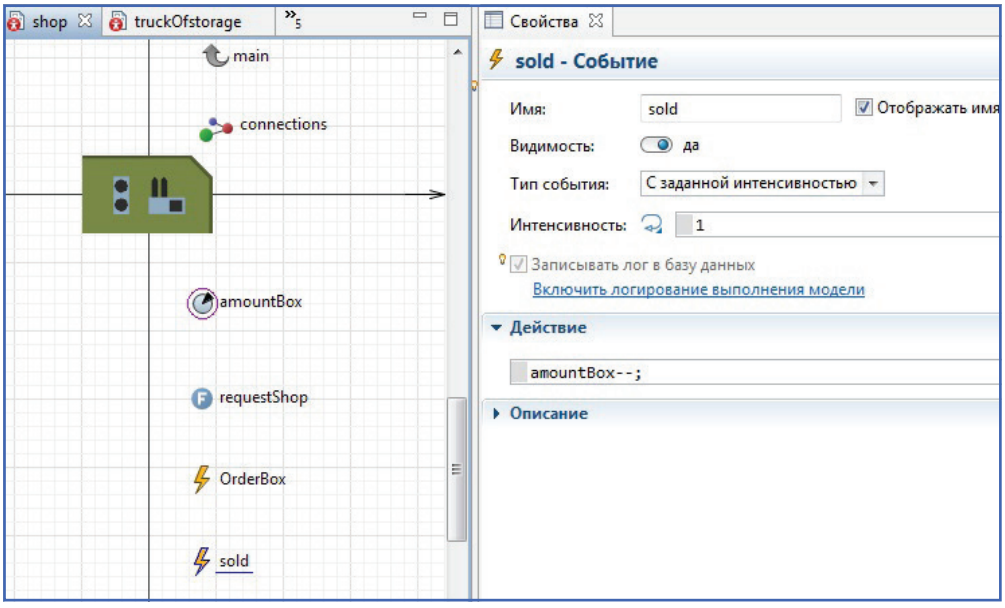

Рис. 3.45. Событие, имитирующее продажи

Шаг 5. Запуск модели

Запустите модель и проверьте работоспособность всех ее агентов (рис. 3.46).

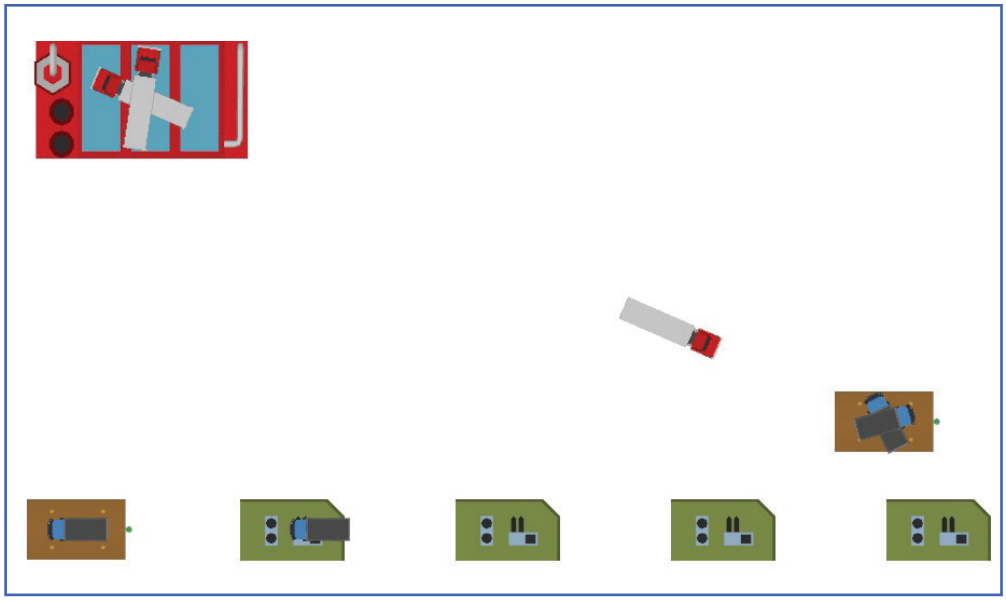

Рис. 3.46. Работа модели

Самостоятельно

1. Добавьте еще один склад с двумя магазинами.
2. Добавьте сбор статистики по грузовикам на складах и на заводе.
3. По результатам прогона модели со статистикой, оптимизируйте количество грузовиков.
4. Творческий проект. Создайте агентную модель транспортной системы.

Лабораторная работа № 4
Пешеходное моделирование. Модель магазина

Задача

Промоделировать движение и обслуживание покупателей в магазине самообслуживания. Магазин имеет несколько разделов. В магазине один вход и он же выход. Покупатели рассчитываются в пяти кассах на выходе из магазина. Схема магазина приведена на рис. 4.1.

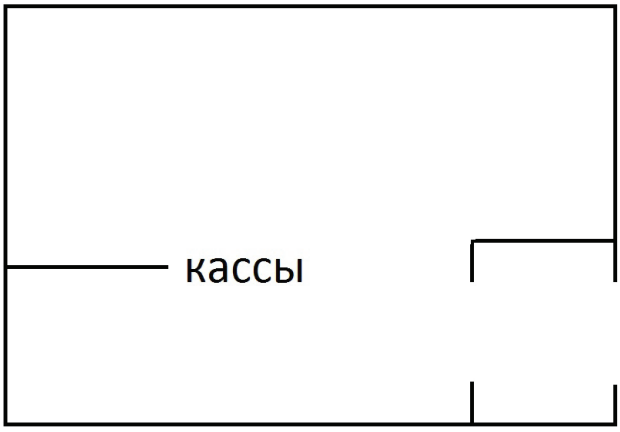

Рис. 4.1. План магазина

Решение

Этап 1. Разметка пространства модели

Шаг 1. Создание модели

Создайте новую модель, выберите единицы модельного времени и дайте ей имя (рис. 4.2).

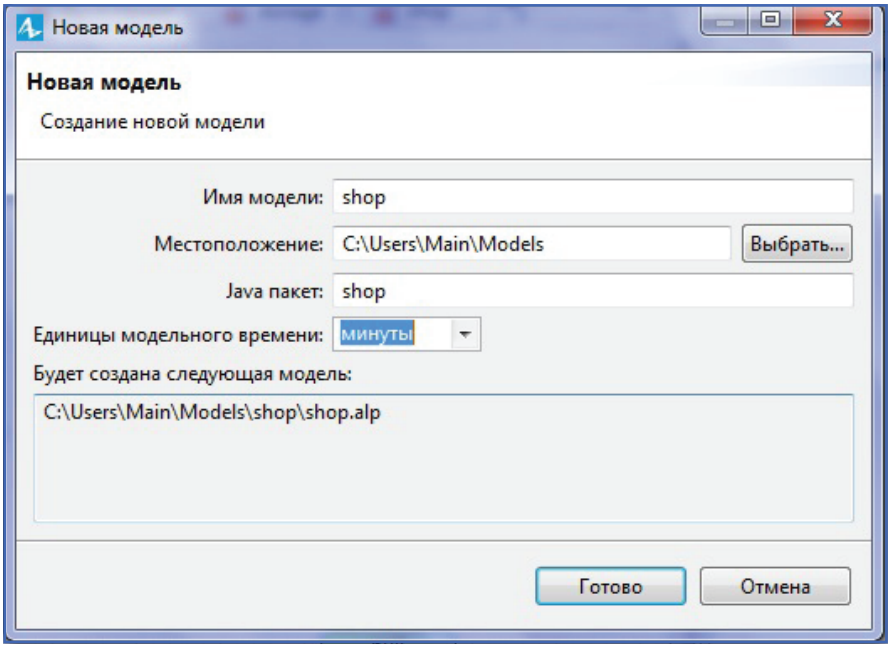

Рис. 4.2. Создание пешеходной модели

В любом графическом редакторе создайте план магазина, подобный приведенному на рис. 4.1.

Шаг 2. Размещение плана магазина в модели

Перейдите на вкладку Презентация и перетащите инструмент Изображение. В открывшемся окне проводника выберите файл с планом магазина. В результате должно получиться, как на рис. 4.3.

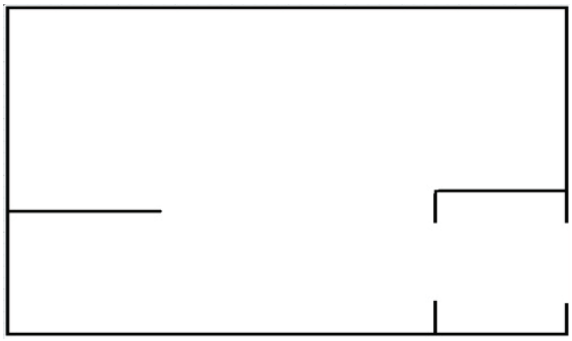

Рис. 4.3. План магазина в модели

Шаг 3. Создание стен магазина в модели

Для создания пешеходных моделей используется пешеходная библиотека, расположенная на отдельной вкладке (рис. 4.4). Перейдите в нее. В ней есть два раздела: разметка пространства и блоки. Создание пешеходной модели начинается с разметки пространства модели.

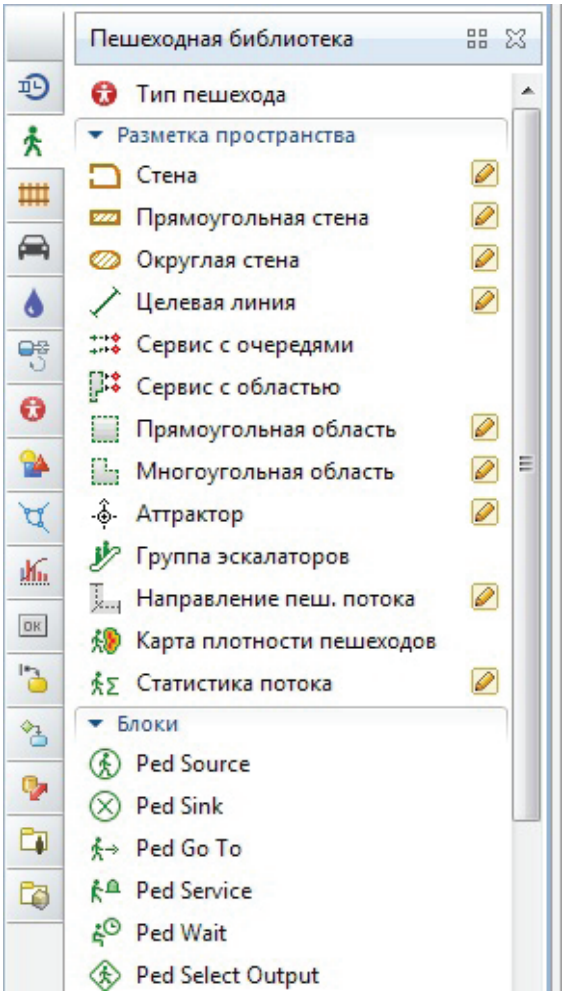

Рис. 4.4. Пешеходная библиотека

Перейдите в раздел Разметка пространства и выберите инструмент Стена, дважды щелкнув на нем. Обрисуйте периметр плана магазина. Результат работы приведен на рис. 4.5.

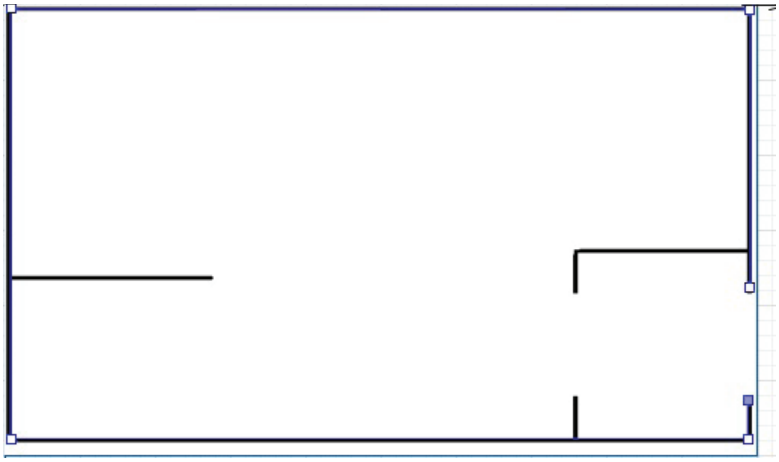

Рис. 4.5. Создание внешних стен магазина

Для создания внутренних стен выберите инструмент Прямоугольная стена, дважды щелкнув на нем (рис. 4.6). Нарисуйте прямоугольники вокруг внутренних стен.

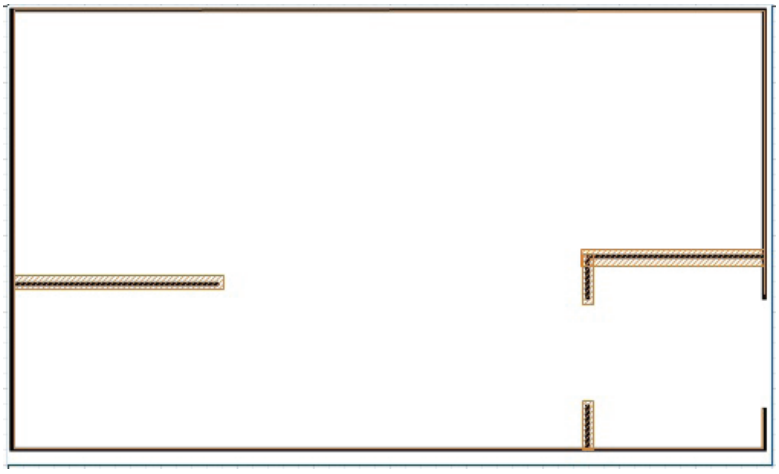

Рис. 4.6. Создание внутренних стен магазина

Шаг 4. Моделирование линии входа покупателей

В пешеходных моделях нужно указывать место появления пешеходов в модели, причем оно должно быть в области обрисованной части модели. Для создания линии входа используется инструмент Целевая

линия. Этот же инструмент может быть использован и для задания конечной цели движения пешеходов. Выберите этот инструмент, дважды щелкнув на нем. Нарисуйте линию входа у входа в магазин (рис. 4.7). В свойствах задайте ее имя EnterLine.

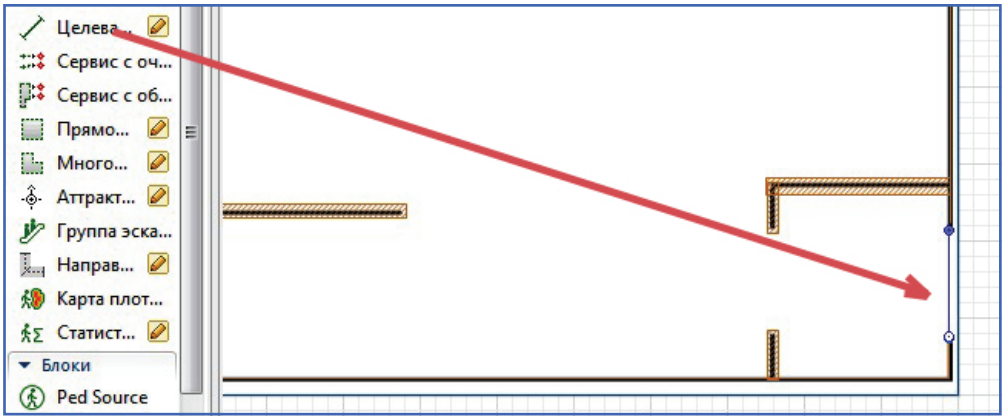

Рис. 4.7. Создание линии входа в магазин

Шаг 5. Моделирование разделов магазина

Для моделирования разделов магазина, которые посещают покупатели, используем инструмент Прямоугольная область (рис. 4.8). При создании Прямоугольной области автоматически в ней создаются аттракторы. Аттракторы — это специальные элементы пешеходной модели, которые «притягивают» пешеходов к себе. Их использование в модели делает возможным управление движением пешеходов.

Выберите инструмент Прямоугольная область и нарисуйте первый раздел магазина в правом верхнем углу плана магазина. В нем сразу будут размещены аттракторы. Нажмите на кнопку Аттракторы в свойствах инструмента Прямоугольная область и задайте их количество. После создания Прямоугольной области щелкните на аттракторе и переместите его. Расставьте аттракторы в места скопления товаров.

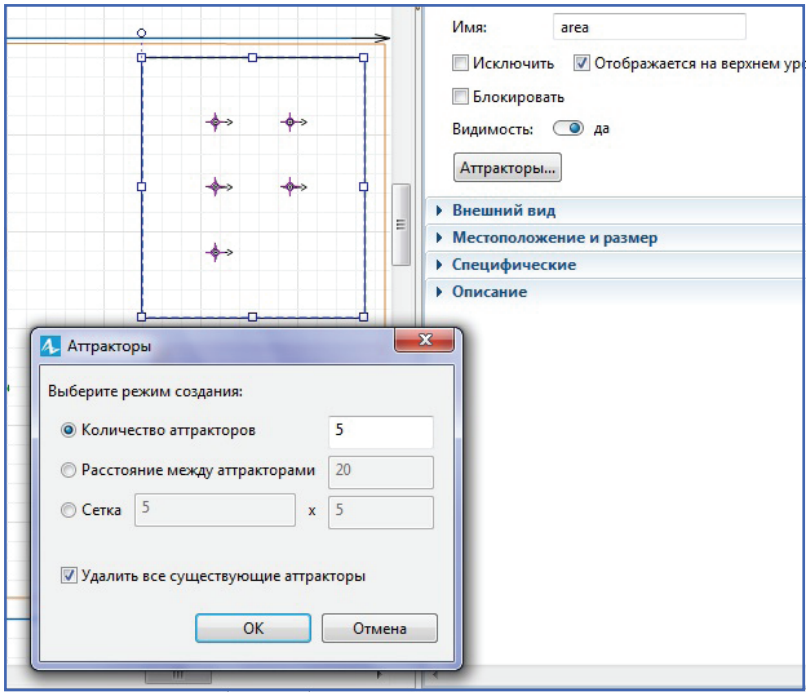

Рис. 4.8. Моделирование разделов магазина

Создайте 4 области для всех разделов магазина. Задайте и расставьте в них аттракторы. В конечном итоге должно получиться, как на рис. 4.9.

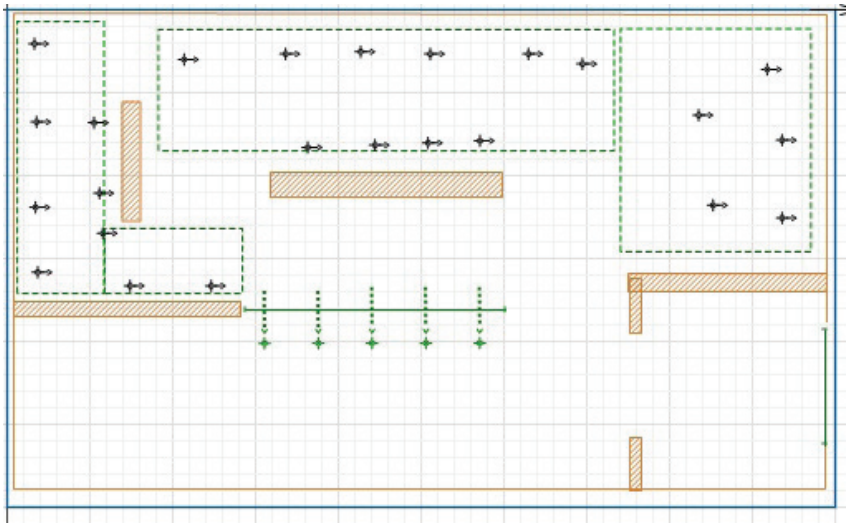

Рис. 4.9. Все разделы магазина

Шаг 6. Моделирование положения касс

Для задания положения касс используется инструмент Сервис с очередями (рис. 4.10). Перетащите его на рабочее поле модели в область касс. В свойствах инструмента задайте количество касс 5 и количество очередей. В магазине к каждой кассе идет своя очередь, поэтому в пункте Обслуживать пешеходов из выберите вариант Ближайшей очереди.

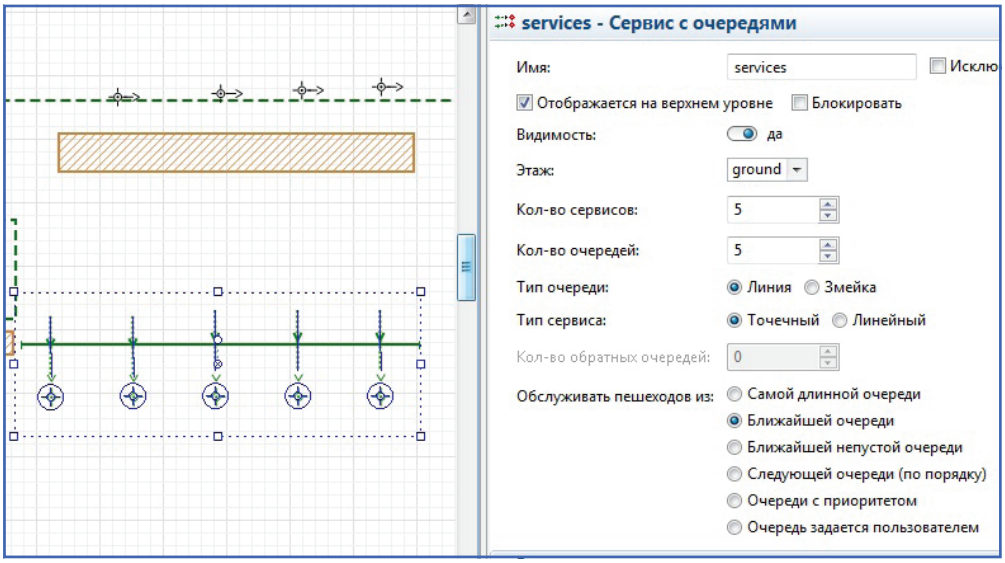

Рис. 4.10. Моделирование касс

Этап 2. Задание логики модели

Шаг 1. Моделирование появления покупателей

Логика модели создается с помощью блоков из раздела Блоки. Блок, моделирующий появление пешеходов в модели, называется PedSource (рис. 4.11). Перетащите его на свободное от плана магазина место в модели. В свойствах блока задайте интенсивность появления пешеходов 500 человек в час. В разделе свойств укажите место появления пешеходов в модели — EnterLine.

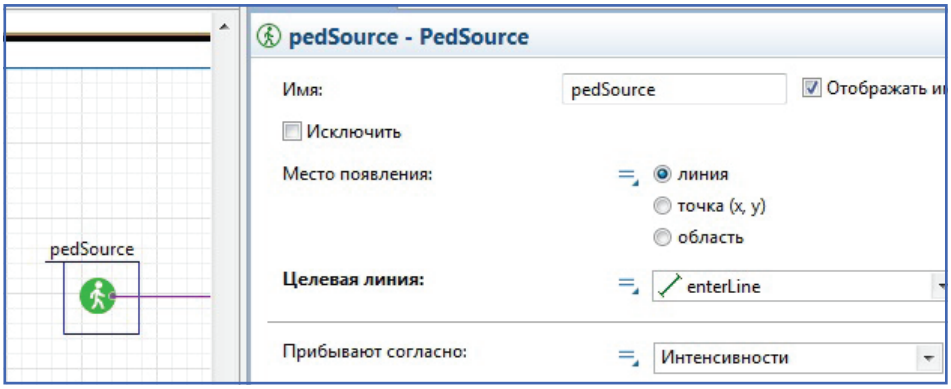

Рис. 4.11. Моделирование прихода покупателей

Шаг 2. Моделирование движения покупателей по магазину

Покупатели могут пойти в любой из 4 разделов магазина почти с равной вероятностью. Для реализации этого движения используем разделение потока покупателей после их входа в магазин на 4 потока (рис. 4.12). Разделение потока пешеходов осуществляет блок PedSelectOutput. Перетащите его в модель и соедините его вход с выходом блока PedSource. В свойствах блока задайте вероятности разделения потока в долях от единицы.

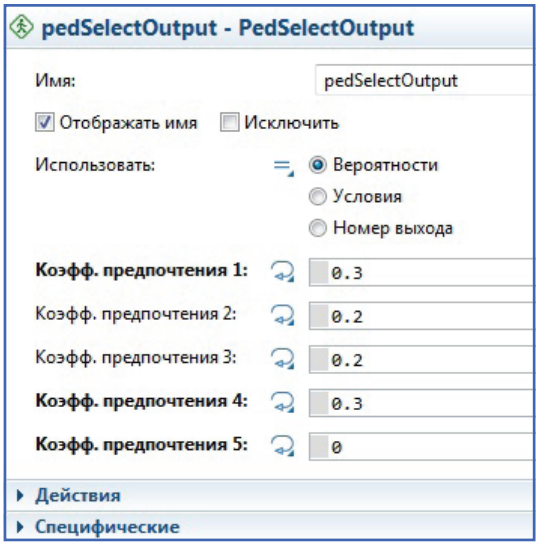

Рис. 4.12. Моделирование разделения потока покупателей

Движение пешеходов в модели задается блоком pedGoTo. Перетащите 4 таких блока на рабочее поле модели и соедините их входы с выходами блока PedSelectOutput (рис. 4.13).

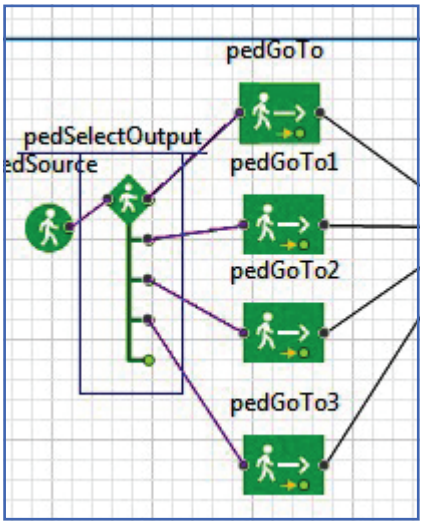

Рис. 4.13. Моделирование движения покупателей

В свойствах блоков pedGoTo укажите направление движения покупателей, направив каждый поток в свой раздел магазина (рис. 4.14).

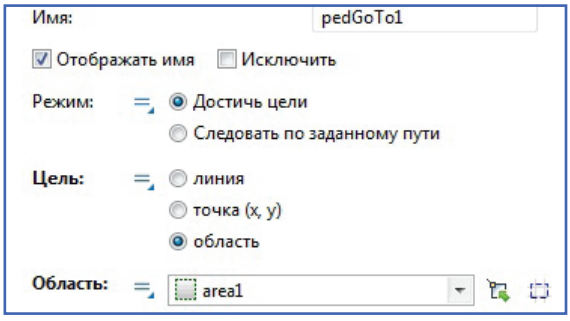

Рис. 4.14. Свойства блока pedGoTo

Такой вариант моделирует поведение очень целенаправленных покупателей, идущих строго в нужный им отдел. Реально покупатели обычно проходят по всем отделам. Поэтому направим их с выхода первого блока pedGoTo на вход второго блока pedGoTo и т. д. В конечном итоге должно получиться, как на рис. 4.15.

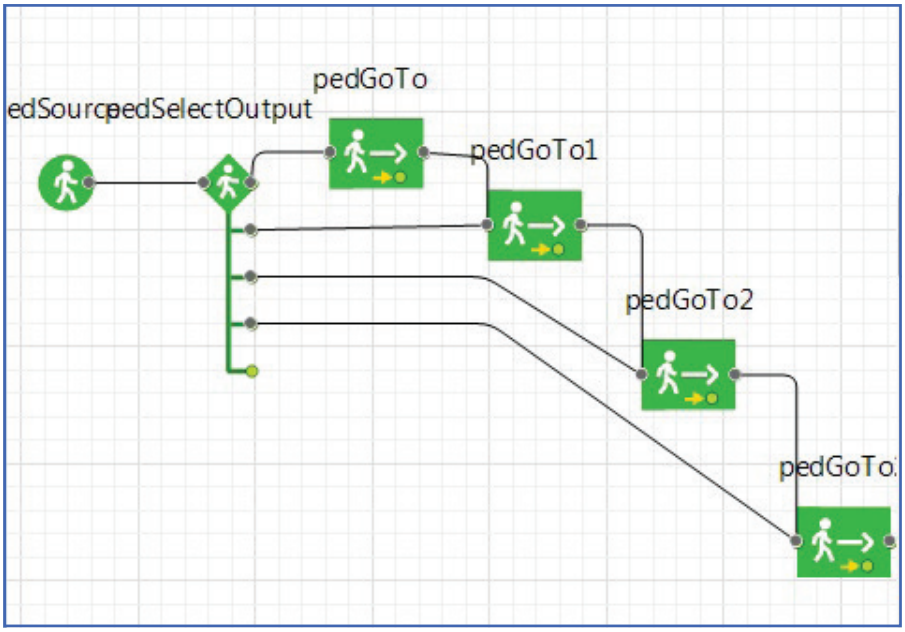

Рис. 4.15. Моделирование движения покупателей по всем разделам магазина

Шаг 3. Моделирование обслуживания покупателей

Блок PedService моделирует обслуживание пешеходов в пешеходных моделях (рис. 4.16). Перетащите этот блок на рабочее поле модели и соедините его вход с выходом последнего в цепочке блока pedGoTo. В свойствах блока pedService укажите место и время обслуживания. В пункте свойств Выбирается очередь выберите вариант Самая короткая очередь (рис. 4.16).

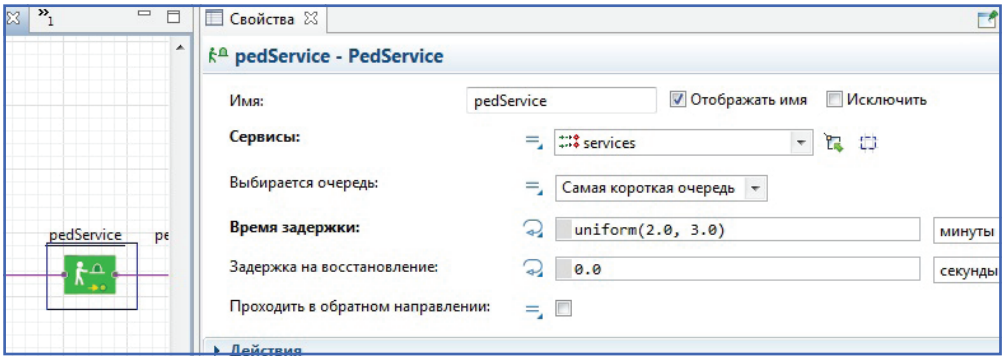

Рис. 4.16. Моделирование обслуживания покупателей

Шаг 4. Моделирование ухода покупателей из магазина

Для моделирования ухода пешеходов из модели используется блок pedSink. Перетащите его в модель и соедините его вход с выходом блока pedService. В конечном варианте логическая модель должна быть, как на рис. 4.17.

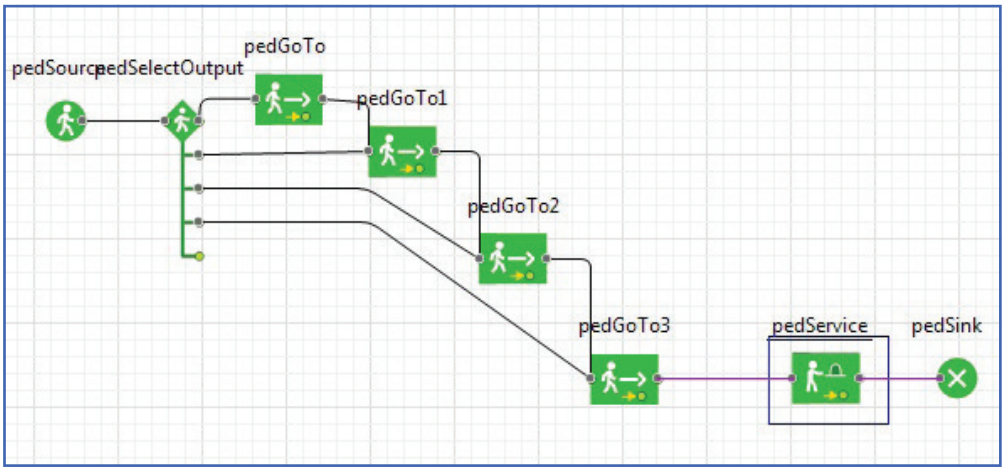

Рис. 4.17. Логика работы модели

Шаг 5. Проверка работоспособности модели

Запустите модель и понаблюдайте за движением пешеходов, которые будут изображаться точками.

Этап 3. Анимация модели

Шаг 1. Создание агента Покупатель

Для анимации пешехода чем-то более информативным, чем точки, создадим нового агента в модели — Покупатель. Перетащите блок Тип пешехода на рабочее поле модели. Запустится Мастер создания нового типа пешехода. На первом шаге Мастера задайте имя нового типа — Customer (рис. 4.18).

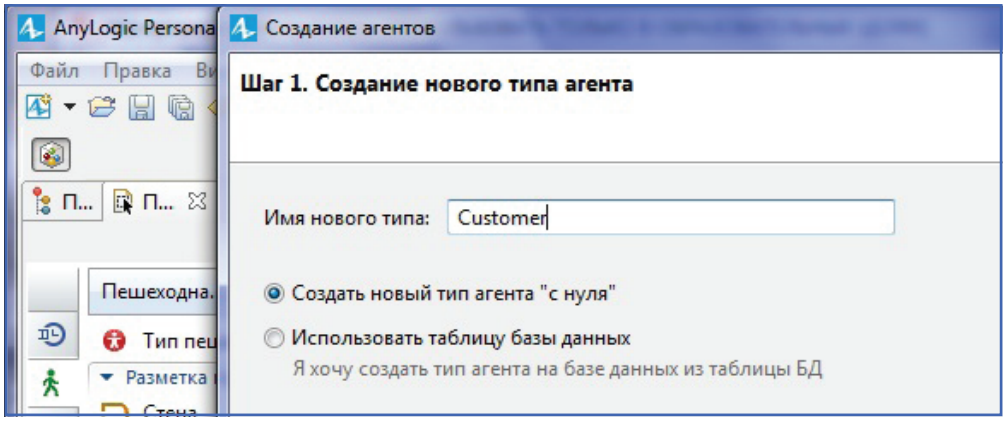

Рис. 4.18. Первый шаг Мастера по созданию нового типа пешехода

На втором шаге мастера выберите анимацию агента Покупатель (рис. 4.19).

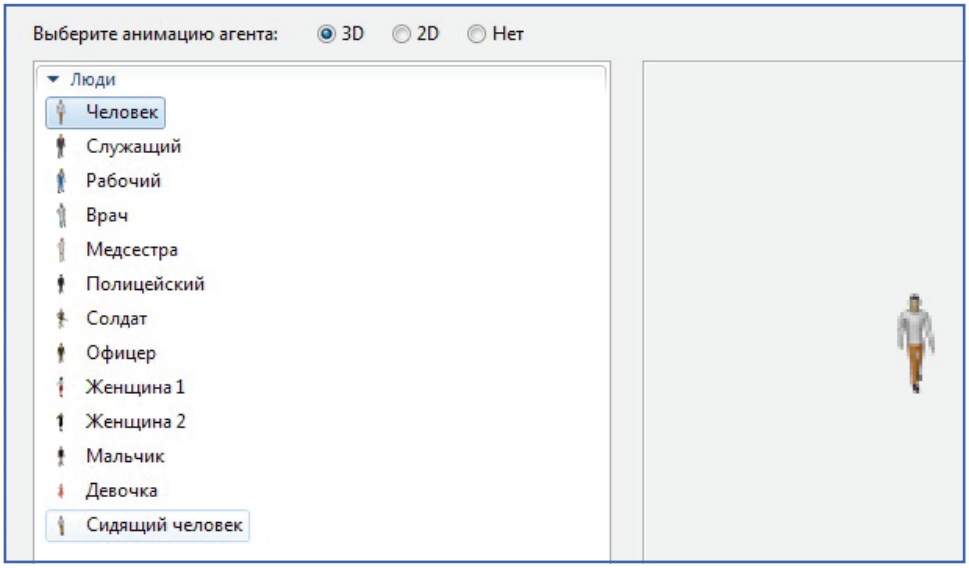

Рис. 4.19. Второй шаг Мастера по созданию нового типа пешехода

Шаг 2. Привязка агента Покупатель к модели

Перейдите в свойства объекта pedSource и задайте тип агента на его выходе (рис. 4.20).

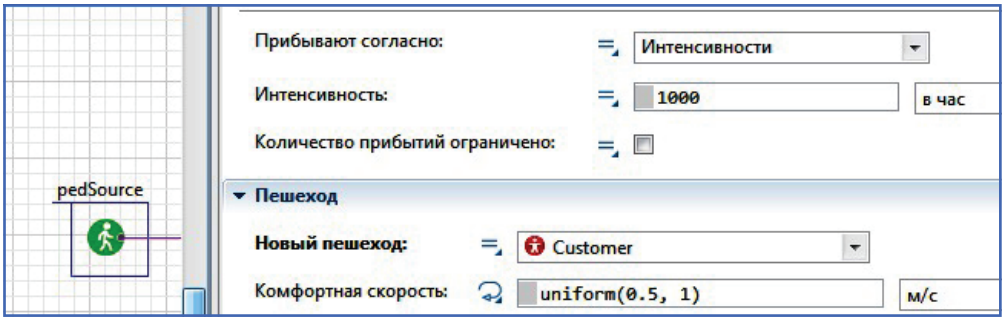

Рис. 4.20. Привязка нового типа пешехода к объекту pedSource

Шаг 3. Задание окна 3D-просмотра в модели

Перейдите на вкладку Презентация и перетащите объект 3D-окно на свободное место в модели (рис. 4.21).

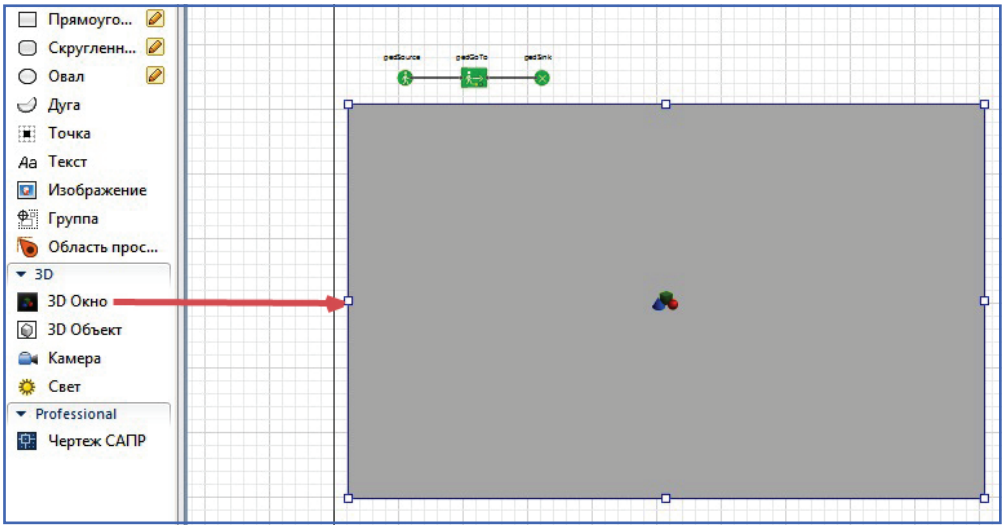

Рис. 4.21. Задание 3D-окна

Шаг 4. Добавление сбора статистики в модели

Для отображения интенсивности потока покупателей и места скопления людей в модели используется инструмент Карта плотности пешеходов из раздела Разметка пространства в пешеходной библиотеке. Перетащите этот инструмент на свободное место в модели (рис. 4.22).

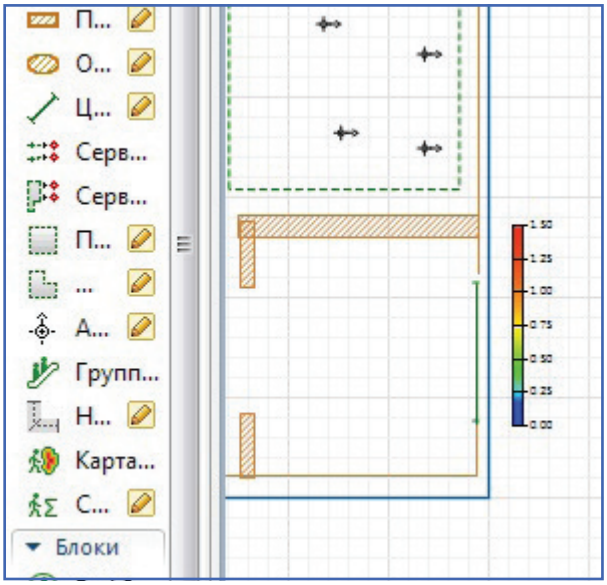

Рис. 4.22. Задание карты плотности потока пешеходов

Шаг 5. Запуск модели

Запустите модель (рис. 4.23). Понаблюдайте за движением и накоплением покупателей в магазине.

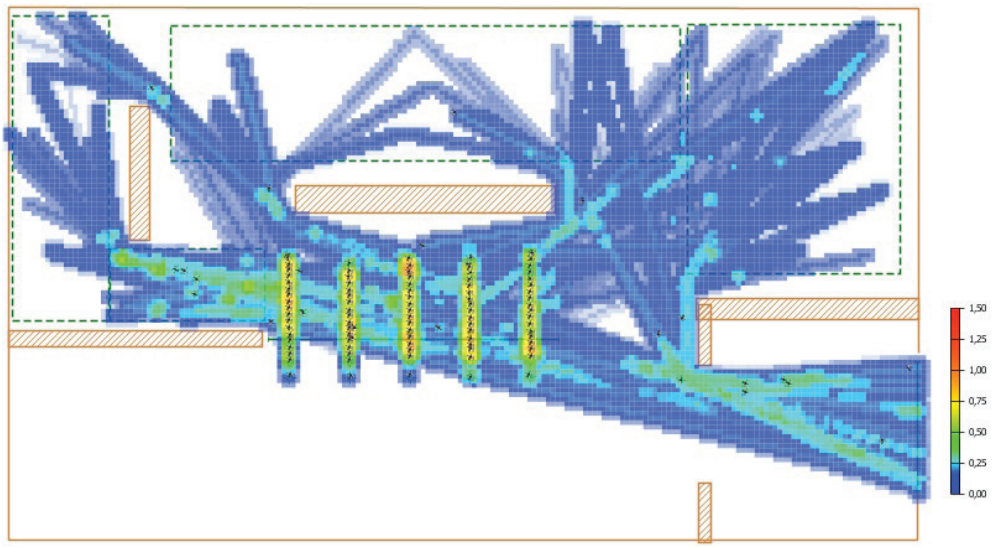

Рис. 4.23. Работа модели в 2D-варианте

Перейдите в окно 3D-просмотра (рис. 4.24).

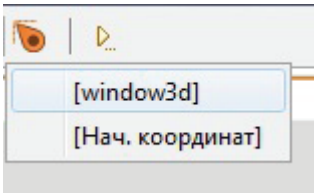

Рис. 4.24. Переход в 3D-вариант

Понаблюдайте за объемным вариантом модели (рис. 4.25).

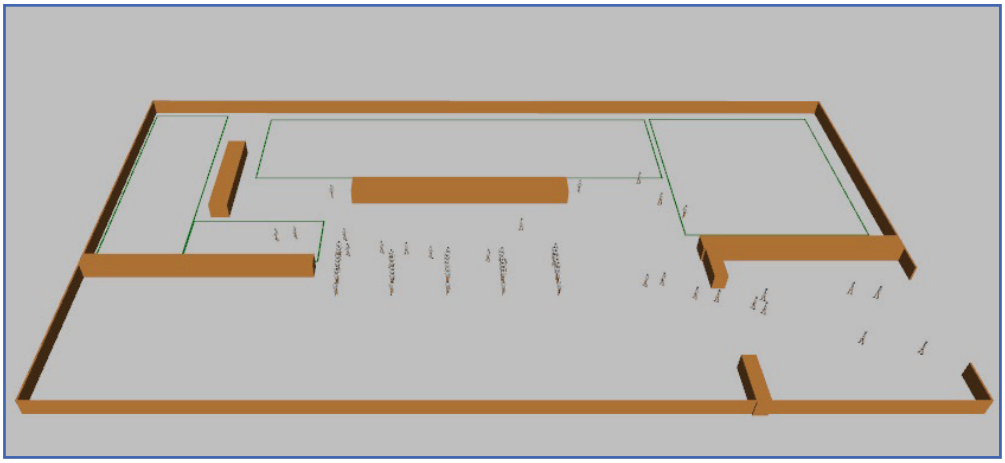

Рис. 4.25. Работа модели в 3D-варианте

Самостоятельно

1. Выявите узкие места модели и предложите их решение.
2. Творческий проект. Создайте пешеходную модель места массового обслуживания.

Содержание

Учебное электронное сетевое издание

Часть 2

Лимановская Оксана Викторовна

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ
В ANYLOGIC 7

Редактор И. В. Коршунова
Верстка О. П. Игнатьевой
Корректор Е. Е. Афанасьева

Подписано в печать 02.03.2017. Формат 70×100/16.
Гарнитура Newton. Уч.-изд. л. 4,75.

Издательство Уральского университета
Редакционно-издательский отдел ИПЦ УрФУ
620049, Екатеринбург, ул. С. Ковалевской, 5
Тел.: 8(343)375-48-25, 375-46-85, 374-19-41
E-mail: rio@urfu.ru


ДИЗАЙН УНИВЕРСИТЕТСКОГО КНИГОИЗДАНИЯ
УНИВЕРСИТЕТСКАЯ КНИГА
2015


9 785799 619961