ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

###### Факультет бизнеса и менеджмента

Апрелкова Юлия Игоревна

**Имитационная модель цепи поставок транспортной компании в условиях динамического изменения заказов**

Выпускная квалификационная работа - БАКАЛАВРСКАЯ РАБОТА

по направлению подготовки *бизнес-информатика*

образовательная программа «бизнес-информатика»

|  |  |
| --- | --- |
| Рецензент  д-р технических наук, профессор кафедры бизнес-аналитики  А. С. Акопов  Москва 2016 | Научный руководитель  преподаватель  А. Л. Бекларян |

**Аннотация**

В рамках данной выпускной квалификационной работы была реализована имитационная модель, описывающая основные бизнес-процессы транспортной компании. Как результат функционирования модели, можно получить ряд параметров таких как эффективность работы отдельных отделов, так и в принципе, оценить рентабельность деятельности компании, с финансовой стороны. Так же на основе данной модели были решены оптимизационные задачи, такие как: максимизация прибыли компании в зависимости от размера автопарка, количества складов транспортной компании.

**Ключевые слова:** имитационное моделирование, логистические цепочки, AnyLogic

Содержание

[Введение 4](#_Toc450243168)

[Глава 1. Теоретические основы моделирования деятельности транспортной компании 6](#_Toc450243169)

[1.1 Основы логистики 6](#_Toc450243170)

[1.2 Понятие, цели и область применения имитационного моделирования 8](#_Toc450243171)

[1.3 Подходы имитационного моделирования 9](#_Toc450243172)

[1.4 Описание деятельности транспортной компании. 11](#_Toc450243173)

[1.5 Описание существующих моделей и инструментов. 12](#_Toc450243174)

[Глава 2. Инструментальные средства моделирования деятельности транспортной компании 13](#_Toc450243175)

[2.1 Сравнительный анализ пакетов имитационного моделирования 13](#_Toc450243176)

[2.2 Постановка задачи 19](#_Toc450243177)

[2.3 Описание бизнес-процессов 22](#_Toc450243178)

[Глава 3. Программная реализация 27](#_Toc450243179)

[3.1 Условные обозначения модели 27](#_Toc450243180)

[3.2 Объекты модели 28](#_Toc450243181)

[3.3 Результаты работы модели 40](#_Toc450243182)

[Заключение 47](#_Toc450243183)

[Список используемой литературы 48](#_Toc450243184)

# **Введение**

Формирование имитационного моделирования началось еще в 50-х годах XX века на основе только появляющихся языков Fortran [32, с. 3] и Algol [29]. Вычислительные возможности компьютеров развивались, в результате, имитационное моделирование стало решать целый ряд задач помимо научного применения, для которого оно изначально создавалось. С появлением известного языка имитационного моделирования – GPSS [5, с. 8] в 60-х годах, начинается активное изучение систем массового обслуживания. А последующие усовершенствование компьютерных мощностей и возможностей, а также развитие объектно-ориентированных языков [13], на которых базируются практически все программные средства, реализующие имитационные модели, значительно ускорило совершенствование моделирования. Увеличилось быстродействие, изменился графический интерфейс программ: они стали понятными и доступными в пользовании.

Кроме того, значительно расширился набор задач, которые способны решить имитационные модели. Одним из важных направлений имитационного моделирования является решение проблем логистики. Так как сама логистика претерпела значительные изменения в XX веке: произошли существенные преобразования управления материальными потоками, начали внедряться и использоваться новые методы и технологи доставки товаров. Процесс управления и оптимизации логистических операций значительно усложнился в связи с развитием и совершенствованием современной логистики. Поэтому построение и оптимизация процессов логистической компании являются **актуальной темой для исследования**.

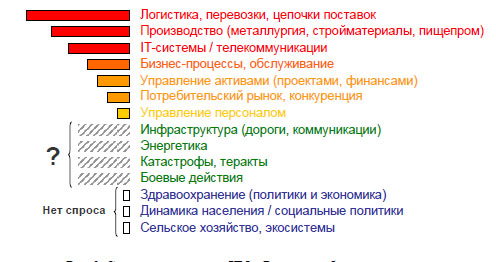


Рис.1 Структура спроса на имитационные модели в России по областям применения.

Согласно данным статьи Борщева А. В. [31, с. 12] логистика является предметной областью, в которой более других востребовано имитационное моделирование, что также подчеркивает актуальность текущей работы.

Транспортная компания является **объектом** данной работы.

**Предметом** данного исследования выступает имитационная модель деятельности транспортной компании.

**Цель**: Разработка имитационной модели деятельности транспортной компании, включающей в себя как оценку эффективности функционирования отделов компании, так и рентабельности бизнеса в целом.

Для достижения поставленных целей были выдвинуты следующие **задачи**:

* Изучить основы имитационного моделирования
* Выявить основные бизнес-процессы транспортной компании
* Провести сравнительный анализ программных средств разработки имитационных моделей
* Построить имитационную модель транспортной компании, провести калибровку и верификацию модели
* Провести ряд экспериментов и описать полученные результаты.

Для решения поставленных задач были применены следующие **методы и инструменты**: EPC [28, 137], BPMN [42] нотации на этапе проектировании деятельности транспортной компании. На следующем этапе - реализации модели в программном продукте AnyLogic, был использован метод системной динамики, процессный подход и агентное моделирование.

В первой главе приводиться понятие логистики, описываются теоретические основы имитационного моделирования, рассматриваются различные подходы данного метода.

Во второй главе рассматриваются существующие программные средства для построения имитационных моделей и обосновывается выбор наиболее подходящей из них. Кроме этого, формулируется постановка задачи, решение которой будет описано в следующей главе.

В третьей главе описывается сама модель и приводняться результаты ее функционирования.

В заключении работы подводятся итоги работы модели, характеризуются ее возможности.

# **Глава 1. Теоретические основы моделирования деятельности транспортной компании**

## **Основы логистики**

Логистика – наука о планировании, организации, управлении и контроле движения материальных и информационных потоков в пространстве и во времени от их первичного источника до конечного потребителя. [12, с. 7]

Хотя первые упоминания термина логистики можно отнести к 9 веку, тем не менее особенно активно как наука она начала развиваться во время второй мировой войны. Логистику применяли для решения стратегических задач, она объединяла в себя взаимодействие военной промышленности, снабжения, а также транспортировку для обеспечения армии всеми требуемыми ресурсами.

С течением времени данные знания начали применяться не только в военной области, но и стали решать спектр гражданских задач [12, с. 10], связанных с рациональным управлением материальных потоков.

Время не стоит на месте, логистика становится самостоятельной дисциплиной, появляется множество методов и подходов. К концу XX века выделяют следующие разделы логистики: снабженческую, производственную, сбытовую, транспортную, информационную и другие разновидности.



Рис.2. Общая структура логистической цепи.

В рамках деятельности транспортной компании наибольший интерес представляет транспортная логистика, которая связана с товаром, маршрутом, местом и экономическим аспектом. Данный тип логистики решает следующие задачи:

1. Доставить именно требуемое количество товара
2. Использовать наиболее оптимальный маршрут
3. Доставить именно в требуемую точку
4. И выполнить все вышеперечисленное с наименьшими издержками.

Кроме того, одним из важных вопросов, который встает перед перевозчиками, является выбор вида транспорта.

Существующие виды транспорта, используемые при транспортировке: автомобильный, воздушный, железнодорожный, морской, внутренний водный транспорт, трубопроводный.

В целом, логистику можно поделить на внутреннюю, которая происходит внутри предприятия и внешнюю, которую также называют сетями поставки.

В связи с тем, что структура логистической цепи становилась все более сложной, процесс управления этой цепью, также, как и оптимизация используемых ресурсов также становились все более нетривиальными задачами. Возникла необходимость разрабатывать модели данных цепей.

Модели логистических цепей можно разбить на два класса: графические и математические.

Первый из них отражает модели производственных и бизнес-процессов. Как правило, выполняются в нотациях IDEF0[9, c.100], BPWin [9, c. 113], UML [9, c. 196], ARIS [33, c.26].

Математические или количественные модели делятся на ряд подклассов:

* Статические описательные модели
* Статические оптимизационные модели
* Динамические имитационные модели

Математические методы помогают определить численные показатели деятельности транспортных сетей. Отличительной чертой имитационных моделей является тот факт, что они отражают динамику процессов, происходящих в реальных системах, в результате, показатели модели вычисляются не в виде констант, а в виде функций, зависящих от времени.

## **Понятие, цели и область применения имитационного моделирования**

В основе любой модели лежит какая-либо проблема, результатом изучения данной проблемы является модель. Если изначально стоит задача не только описать существующую деятельность, но и решить данную проблему, то модель начинают оптимизировать, в итоге, оптимальная модель должна стать основой для управления деятельностью, согласно ей, процессы и параметры должны быть изменены и внедрены.

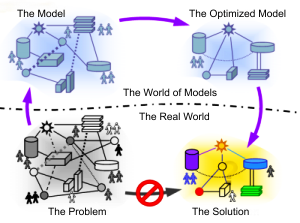


Рис.3 жизненный цикл модели.

Сутью имитационного моделирования является создание модели, которая детально описывает и воспроизводит деятельность самой системы.

К данному методу обращаются в случаях:

Когда аналитическая модель не может быть создана либо она будет не способна полностью отображать поведение системы. Например, при наличии параметра - времени, либо последовательностей или причинно-следственных связей;

Когда невозможно, либо очень дорого экспериментировать на реальных составляющих системы или на самой системе в целом.

Преимуществом имитационных моделей является возможность управления временем в модели: так время в модели системы с медленной изменчивостью можно ускорить, в то время как для систем, в которых в жизни процессы происходят очень быстро, время можно замедлить, чтобы детально проанализировать происходящие изменения в системе.

Построение имитационной модели включает в себя следующие этапы:

* Построение концептуальной модели;
* Внедрение модели по средствам пакета имитационного моделирования;
* Экспериментирование на рабочей модели, сбор результатов и анализ полученных данных.

Области применения имитационного моделирования:

* ИТ-инфраструктура
* Математическое моделирование исторических процессов
* Движение всех типов транспорта
* Логистика
* Производственный сегмент
* Управление проектами
* Рынок и конкуренция
* Бизнес процессы
* Экосистемы

На сегодняшний день область применения имитационных моделей очень широка. Данный метод прекрасно справляется с решения физических и математических задач, для которых он первоначально работал. Также очень широко моделирование используется для решения задач массового обслуживания, этим целям имитация служит с 60-х годов. А сегодня имитационное моделирование применяется практически во всех отраслях, где основу составляют процессы, которые можно описать математическим языком. Это и производственные, и логистические и всевозможные процессы, связанные с перемещение автомобильных, воздушных, морских, железнодорожных средств, а также людей. Также имитационное моделирование хорошо воссоздает демографические и экономические проблемы (выход на окупаемость, и т.д.)

## **Подходы имитационного моделирования**

Имитационное моделирование включает в себя следующие подходы:

* Системная динамика
* Дискретно-событийное моделирование
* Агентное моделирование

****

Рис.4 Виды имитационного моделирования

Системная динамика стала результатом исследований конца 1950-х годов Джея Форрестера. Данные исследование были нацелены на изучение зависимости внутренних факторов (организационной структуры) предприятия и эффективности деятельности компании. В основе данного метода лежит описание процессов, протекающих в системе в рамках дифференциальных уравнений, обращенных к форме Коши. В целом, метод необходим для изучения поведения сложных систем в динамике. Он использует высокий уровень абстракции, так как отдельные элементы рассматриваются исключительно в масштабах работы целой системы, особое внимание уделяется изучению связей между элементами. Рассматривая с позиции системной динамики, поведение системы представляется как множество взаимно влияющих задержек и обратных связей: положительных и отрицательных. Так как модель, построенная при помощи описанного метода, работает с количественными агрегатными показателями, то объекты, расположенные в одном накопители неотличимы - их индивидуальные характеристики опускаются. Поэтому работая с данным методом, необходимо учитывать только глобальные зависимости системы. Системная динамика- это верхнеуровневое представление системы.

Дискретно-событийное моделирование рассматривает процессы как последовательность событий, для этого используется транзакции и блоки-объекты. Еще в 1961 году Джефри Гордон воплотил идею дискретно-событийного моделирования в GPSS. Значительным достижением языка GPSS стала реализация такой методики как потоковое моделирование. Который нашел широкое использование и развитие в последующем в различных программных продуктах. Система представляется как продвигающийся по структурной диаграмме поток сущностей. Кроме того, поведение этих сущностей может быть обусловлено нахождением в очереди, конкуренцией с другими сущностями за захват ресурсов и покиданием системы. Моделирование с дискретными событиями наиболее развито и реализовано большим количеством программных средств, так как имеет огромную сферу применения. Производственные процессы наиболее точно описываются именно с помощью применению этого подхода. Данный тип моделирования используется при среднем или низком уровне абстракции: каждый объект рассматривается отдельно, однако большая часть детальных параметров опускается.

Агентное моделирование. В первую очередь, детально определяется каждый отдельный агент: задается ряд параметров его характеризующих. После чего ряд агентов помещается в глобальную среду. Таким образом, что поведение каждого отдельного агента влияет на поведение системы в целом. А сама модель является множеством взаимодействующих объектов.

Данный подход моделирования стал результатом исследований децентрализованных систем, проводимых в 90-е годы XX века. Данный метод использует направленность не сверху вниз, как предыдущие методы, а наоборот функционирование отдельных агентов определяет глобальные правила, существующие в системе. Задача создателя модели: изучить и проанализировать эти глобальные правила на основе предположений о поведении отдельных агентов и их взаимодействии.

Таким образом, выбор метода зависит от структуры системы, поведение которой планируется имитировать.

## **Описание деятельности транспортной компании.**

Транспортные компании занимаются доставкой грузов. Содержат, как правило, ряд отделов:

1. Маркетинга, занимающейся привлечением клиентов по средствам рекламы в интернете, баннерной рекламы и другими способами.
2. Логистический - занимается построением оптимальных маршрутов, расчётами времени движения и издержек, которые несет при компания, при выполнении заказов.
3. Юридический и бухгалтерский отделы. Занимаются общей поддержкой деятельности, путем составления корректных договоров с клиентами, и расчетами заработной платы персонала, оплатой налогов и прочим.

Клиенты обращаются в компанию с целью доставить заказ, содержащий товар в определенном количестве. В результате эффективной деятельности логистического отдела, определяются склады на которых присутствует достаточное количество товара, выбирается подходящее транспортное средство и строится оптимальный маршрут.

Кроме того, как правило клиентом вводится временное ограничение по времени доставки. Компании должны осуществлять доставку в срок, иначе они выплачивают штрафу клиенту.

Эффективная работа всех отделов, приводит к положительным отзывам, а также повторным обращениям клиентов.

Имитационная модель будет описывать деятельность подобной компании.

## **Описание существующих моделей и инструментов.**

В настоящее время реализовано большое количество моделей, описывающих и решающих логистические проблемы. Каждая имеет свои преимущества и недостатки, однако каждая модель решает определенную задачу. Сама задача сформулирована в постановке: пункт 2.2. Отличительной же особенностью модели является учет внутренней структуры компании: эффективность работы отделов, которая влияет на формирование заказа, кроме того, забегая вперед, хотелось бы отметить, что период прихода клиентов также является результатом функционирования отделов на базе системной динамике, описанной в пункте 3.2 для агента delivery. К тому же вся логика движения транспортного средства задана с помощью системной динамике, которая будет описана в пункте 3.2 для агента Truck. В целом же вся модель является агентной, так как каждая физическая сущность реализована в виде агента. Таким образом, модель представляет из себя симбиоз всех трех подходов имитационного моделирования.

Сравнивая же выбранное и обоснованное в пункте 2.1 программное решение с линейкой продуктов 1С:Предприятие 8. TMS Логистика или другими пакетами, стоит отметить, что AnyLogic, благодаря возможностям Java, становится уникальным продуктом, так как модель является кастомизируемым продуктом, отвечающим требованиям и условиям заказчика, а не пакетным решением.

**Выводы:** в данной главе, были разобраны понятие логистики, имитационного моделирования, описана деятельность транспортной компании, перечислены различные подходы реализации имитационных моделей, а также указана область применения данного метода.

# **Глава 2. Инструментальные средства моделирования деятельности транспортной компании**

## **2.1 Сравнительный анализ пакетов имитационного моделирования**

*IThink-*современный объектно-ориентированный пакет имитационного моделирования; обеспечивает вычислительную, графическую и информационную поддержку построения высокоуровневого анализа. Имеет широкие области применения: производство, бизнес, управление финансовыми потоками. Основным подходом, реализуемым данным пакетом является системная динамика. Поэтому основными составляющими являются станции, конвертер, потоки и соединители. Модель включает в себя разделение на уровни и иерархии. Фактически, модель представляет из себя набор подсистем, каждая из которых состоит из объектов и взаимодействий с определенной логикой.

Графические результаты функционирования модели, как правило, представляют из себя диаграммы разброса или временные диаграммы.

Данная среда моделирования способна решать целый ряд задач системного анализа:

- использование имитации процессов для получение качественных характеристик и статистических оценок;

- реализация моделей среднего уровня сложности как комбинация простых;

- разработка моделей широкого спектра систем;

- выявление зависимостей у составляющих, с целью выявить поведенческие характеристики объектов.

Требуемые технические характеристики: Microsoft Windows XP/ Vista/7/8

256 Мб оперативной памяти и 200 Мб свободного пространства на жестком диске.

В целом, данная программа отличается простотой в использовании, не требует специальных навыков в области программирования либо математических методов.

*Powersim* 

рис. 5 интеграция динамических моделей с системой управления предприятием

Powersim является пакетом, реализующим классический подход системной динамики. Может быть использован как автономное приложение, так и интегрирован с другими системами, как отражено на рисунке 5.

Powersim интегрируется с системой планирования/бюджетирования (SAP SEM-BPS, Cognos TM1 и прочими), эта система в свою очередь может быть интегрирована с системой, занимающейся сбором и консолидацией информации (Oracle, SAP BW, MS SQL Sever, и прочие). В самом низу иерархии находится ERP система, хранящие конкретные транзакционные данные. Однако интеграция может происходить не обязательно данным путем, возможно интеграция напрямую с нижним уровнем (SAP ERP).

Данная программа выделяется среди других средств, поддерживающих метод системной динамики, за счет возможностей реализации моделей для сложных организационных структур. Powersim достиг высокого уровня в развитие средств визуального программирования, также выделяется среди других возможностью интегрироваться с информационными системами (Oracle, SAP BW) Помимо этого в PowerSim реализован широкий спектр расширенных возможностей: оптимизация бизнес-процессов, распараллеливание вычислений, встроенные блоки анализов рисков, методы вероятностного моделирования и другие.

Модели в данной системе удобно строятся и воспринимаются, помогают в этом различные доступные способы представления: гистограммы, временные графики, таблицы. Данная программа позволяет строить модели как дискретных, так и непрерывных процессов.

Модели, построенные с помощью данного программного продукта, будут востребованы:

- При расчёте всевозможных показателей: эффективности, экономических, производственных и других;

**-** При создании моделей деятельности компаний;

- При оптимизации ключевых бизнес-процессов предприятия;

- При расчете и анализе рисков.

Успешная работа с данным пакетом не требует особенных компетенций со стороны пользователя.

*AnyLogic* современное средство имитационного моделирования, реализовавшее в себе различные подходы:

* Системную динамику
* Модели дискретных систем
* Дискретно-событийный подход
* Агентное моделирование.

А также имеет возможность комбинировать все вышеперечисленные методы. Кроме того, AnyLogic оснащён очень развитым графическим интерфейсом, в процессе настройки модели можно настроить 2D или 3D анимацию, благодаря чему результаты становятся более репрезентативными. Нельзя не отметить очень дружественный, понятный интерфейс программы, большое количество встроенных примеров и справочных материалов.

В связи с большим количеством реализованных подходов, область применения данной программы очень велика. Это и производственные процессы, и процессы, связанные с перемещением транспорта (железнодорожного, автомобильного, воздушного и морского), управление проектами и активами и прочие.

AnyLogic поддерживает модели разных уровней: от моделей, где важны параметра одного конкретного объекта до верхнеуровневых моделей, обеспечивающих поддержку стратегических решений. Помимо этого, данный продукт способен работать с непрерывными, дискретными событиями, а также их комбинацией в рамках одной модели. Поддерживает более 40 математических распределений и решает оптимизационный задачи, путем поиска значения параметров при которых достигается оптимальное значение целевой функции, при этом могут быть наложены ограничения на значение параметров.

Наиболее отличительной чертой AnyLogic является кроссплатформенность, так как в основе данного пакета находится объектно-ориентированный язык программирования Java. Благодаря чему данный продукт можно назвать универсальным, так как он может быть полезен пользователям Windows, Mac OS и Linux.

В целом, реализовать модель можно и не обращаясь к коду, однако для создания сложных, функциональных моделей может возникнуть необходимость обратиться к программированию на Java и использованию как стандартных библиотек, так и собственных разработок. Каждый объект модели представляет из себя Java-класс, пользователю предоставляются широкие возможности по переопределению методов базовых классов, добавлению собственных классов и прочим аспектам.

Приложение можно использовать как на одном компьютере, так и, построив Java-апплет, запускать модель из браузера.

Начиная с версии 7.2 в AnyLogic появилась возможность использовать встроенную базу данных, кроме того данную программу можно интегрировать с ERP и CRM системами и отдельными их модулями, а также с базами данных.

Возможности системы очень широки, благодаря прекрасной графической реализации и возможности разрабатывать модель с собственными классами и методами, однако, тем самым это и налагает и ограничения на пользователя, в основном, связанные со знанием языка программирования Java.

Требуемые технические характеристики:

-Microsoft Windows 7/8/10, Apple Mac OS X 10.7.3 или выше, SuSE Linux, x86-32, Ubuntu Linux 10.04 или выше;

-500 MB дискового пространства;

-Не менее 2 Gb оперативной памяти.

*GPSS World* является современной конкурентоспособной средой компьютерного моделирования. В основе заложен объектно-ориентированный язык GPSS. Данная программа оснащена мощным функционалом: способна выполнять статистический анализ результатов, решает оптимизационные задачи, может работать как с непрерывными, так и дискретными событиями. Интерфейс схож со всеми Windows-приложениями, также, как и возможности: использование нескольких окон, привычные команды.

Отличается от систем AnyLogic и Powersim отсутствием графической реализацией математических моделей. В связи с чем процесс отладки является более сложным и длительным.

Тем не менее данная программа обладает широкими возможностями в области имитационного моделирования:

- Способность работать в многозадачном режиме, что увеличивает эффективность работы в целом;

- Возможность проведение экспериментов с автоматическим анализом результатов;

- Широкий выбор встроенных распределений вероятности (больше 20)

- Использование динамического вызова функций из внешних файлов.

Основным направлением считаются системы массового обслуживания.

К главным объектам относятся: процесс моделирования, модель, текст и отчет.

В целом является мощным инструментом моделирования, однако, как было отмечено ранее, более трудоемким в освоении, чем аналогичные программы.

Далее будет приведена таблица, сравнивающая вышеперечисленные инструменты имитационного моделирования.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерии:** | **GPSS World** | **AnyLogic** | **PowerSim** | **IThink** |
| Потоковые диаграммы | **-** | **+** | **+** | **+** |
| Блок-схемы | **+** | **-** | **-** | **-** |
| Наличие встроенного аппарата многопараметрической оптимизации | **-** | **+** | **+** | **-** |
| Наличие объектно-ориентированного подхода | **+** | **+** | **-** | **+** |
| Кроссплатформенность моделей | **-** | **+** | **-** | **-** |
| Удобный экспорт и импорт данных из различных приложений | **-** | **+** | **+** | **-** |
| Case-средства | **-** | **-** | **-** | **+** |
| Поддержка анализа результатов | **+** | **+** | **-** | **+** |
| Возможность использования всех подходов в одной модели. | **-** | **+** | **-** | **-** |
| Модели системной динамики | **-** | **+** | **+** | **+** |
| Модели динамических систем | **+** | **+** | **-** | **-** |
| Дискретно-событийное моделирование | **-** | **+** | **-** | **+** |
| Агентное моделирование | **-** | **+** | **-** | **-** |
| Наличие демонстрационной версии | **+** | **+** | **+** | **+** |
| Наличие студенческой версии | **+** | **+** | платная | **-** |

Таблица 1. Сравнительный анализ программ.

Проведя обзор наиболее популярных инструментов имитационного моделирования, а также выполнив сравнительный анализ, был сделан вывод, что AnyLogic является более подходящим средством, так как поддерживает более широкий набор парадигм, в особенности важной парадигмой для данной задачи является агентное моделирование, так как в ходе построения модели будет использовано множество объектов моделирования, однако свойства части объектов невозможно опустить, поэтому нельзя ограничится методом системной динамики, кроме того AnyLogic имеет большое количество встроенных функций. Возможно решение оптимизационных задач. Модели, реализованные в данном продукте, выглядят более презентабельно, благодаря развитой графике. А возможность работать с Java, делает моделирование более гибким и доступным для реализации желаемого. Кроме того, нельзя не отметить, что AnyLogic предоставляет бесплатную версию своего продукта для образовательных целей, в которой доступно значительное количество функций.

## **2.2 Постановка задачи**

У людей есть необходимость перевозить товары в пределах России, в рамках данной модели используется автомобильный транспорт. Применительно к нашей ситуации: транспортная компания осуществляет перевозки по России (преимущественно центральная часть) морской транспорт автоматически не рассматривается, в силу географических особенностей нашей страны. Трубопроводный также не может быть выбран как основной в силу того, что номенклатура, перевозки которой осуществляет транспортные компании много шире, чем транспортируемые трубопроводным транспортом. Воздушные перевозки в центральной части России не выдерживают ценовую конкуренцию с другими видами транспорта. Внутренний водный транспорт, к огромному сожалению, мало используется в России, в связи с практически полным отсутствием водных транспортных средств, удовлетворяющим нормам безопасности. Новые водные судна производятся крайне редко и в малых количествах, хотя данный вид транспорта является одним из самых дешевых. Остается две категории автомобильная и железнодорожная, однако автомобильный транспорт является более мобильным и способен доставить груз именно в конкретную точку, требуемую заказчиком, что соответствует задаче транспортной логистики. Кроме того, с водителями легче коммуницировать, узнавать текущее местоположение и ожидаемое время доставки. Контролировать перемещение железнодорожного транспорта, особенно при наличии какого-либо сбоя в расписании, практически невозможно. При этом ж/д транспорт доставит груз лишь до ближайшей станции, а не до места заказчика, необходимо все равно дополнительно использовать автомобильный транспорт. Сравнить их экономические различия достаточно тяжело, так как железнодорожный транспорт более экономичен при больших расстояниях. Однако, учитывая указанные нами условия, при оптимальном построении маршрута и наличии нескольких складов расстояние должно быть минимальным насколько это возможно. Таким образом, в рамках данной задачи будет введено **ограничение** на использование только автомобильного типа транспорта.

В данной модели будут использованы следующие значения основных параметров:

Количество складов (переменная department\_quantity) – 6

Число грузовиков (переменная numberOfTrucks) – 7

Количество мест доставки (переменная delivery\_quantity) – 12

В качестве источника возможных позиций будет выступать база данных, которую можно обновлять.

Заказ будет сформирован из пар ключ-значение, где ключом вступает имя груза, а значением его количество.

Так как услуги перевозки востребованы, то на данном рынке существует некоторый спрос на сервис транспортной компании. Вводится оценка населения городов-заказчиков, и устанавливается, что в среднем в них живет около 100000, как показатель общего числа людей. Касательно спроса, он динамически изменяется, кроме того, могут наблюдаться сезонные изменения. По этой причине, спрос задается следующим уравнением: 1000+A\*Math.sin(PI/2\*T), где A- число, задаваемое равномерным распределением в промежутке [-40;40]. Однако это уравнение демонстрирует спрос на все, оказываемые логистические услуги в городе. Поэтому существует параметр, демонстрирующий вероятность обращения конкретно в данную фирму, его значение варьируется от 0,7 до 2 %, согласно нормальному распределению. Произведение данного параметра на спрос и генерирует число потенциальных клиентов, которые вероятно воспользуются услугами компании. Клиенты обращаются в компанию либо самостоятельно, проанализировав рынок услуг, либо в результате воздействия эффективной рекламы, являющейся результатом деятельности отдела маркетинга данной компании. Эффективность отдела маркетинга определяется отношением числа клиентов к числу потенциальных клиентов. После того как клиент обратился за консультацией с ним начинает работать сначала логистический отдел, который составляет оптимальный маршрут, делает оценку времени доставки, а потом и юридический отдел, ответственный за составление всей необходимой документации. Эффективности отделов являются динамическими значениями, зависящими от времени. Если клиента устраивает качество предоставляемых услуг, то он подписывает контракт. Подписавшие контракт клиенты могут повторно обратиться в транспортную компанию.

Каждый подписавший контракт клиент оформляет заказ. Для каждого заказа происходит подбор транспортного средства, выбирается ближайший грузовик, не доставляющий другой заказ в данный момент. Когда грузовик определен, начинается движение транспортного средства: он едет на склад, определенный заказом, при условии, что время выполнения заказа не превышено, забирает указанный товар, при этом число товаров на складе уменьшается на это количество. Если же время превышено, то клиенту выплачивается фиксированный штраф в размере 2500 рублей за недоставленный в срок заказ. После чего происходит процесс упаковки заказа, при этом опять же осуществляется проверка времени, если ограничение по времени превышено, то заказ должен быть расформирован, а грузовик отправится доставлять другой заказ. Если же лимит времени не был превышен, то грузовик отправляется в место доставки. После перехода грузовика в состояние движения к месту доставки проверка на лимит времени перестает действовать вышеописанным способом, то есть товар уже в любом случае будет доставлен, однако, если лимит времени был превышен, то на доставку будет действовать 50% скидка. Далее происходит процесс распаковки груза. После чего грузовик отправляется вновь на ближайший склад. С финансовой стороны, фиксируется время начала и окончания движения грузовика, за каждый час движения водителям выплачивается зарплата 200 рублей. Таким образом, расходы определяются формулой: expenses=WorkTime\*200; Кроме этого, размер расходов увеличивается на 2500 при каждом случае штрафа, описанных выше. А в том случае, когда грузовик уже движется к заказчику, а срок выполнения проходит, то параметр расход увеличивается на expenses=expenses+distance\*10; Таким образом, 50% возвращаются заказчику. Приход же определяется формулой income=income+distance\*20; и перечисляется в компанию в случае доставки заказа. Так как вышеперечисленные действия происходят для каждого отдельного грузовика, доставляющего заказ, то на верхнем уровне, отражающим общую прибыль должно происходить суммирование всего прихода и из него вычитаться все расходы, просуммированные по каждому отдельному грузовику, это переменные доходы и расходы. revenue=revenue+truck.get(i).income-truck.get(i).expenses Однако, нельзя забывать и про фиксированные затраты, которые идут на оплату складов. revenue=revenue-department.size()\*1000/24\*time(); Вышеописанную деятельность транспортной компании необходимо смоделировать.

## **2.3 Описание бизнес-процессов**

Основные описываемые бизнес-процессы:

1) Работа над заявкой

1. Каждый из процессов состоит из ряда действий. Работа над заявкой представлена на EPC-диаграмме. Рис.6
2. Более подробное описание ряда функций (1.Клиент обратился сам, 2.Поиск Клиента) представлено на дополнительных диаграммах.



Рис. 6 Диаграмма процесса работы над заявкой.



Рис.7 Диаграмма самостоятельного обращения Рис.8 Диаграмма поиска клиента

2) Процессы, связанные с доставкой заказа.

Процессы непосредственно, связанные с движением транспортного средства также делятся на ряд основных действий.

1. Движение к складу.
2. Комплектация заказов.
3. Транспортировка заказов потребителям.
4. Разгрузка.
5. Возврат транспортного средства.

Описание логики движения транспортного средства представлено ниже на рис. 9



Рис. 9. Движение транспортного средства.

**Выводы:** во второй главе была дана краткая характеристика наиболее популярным средствам имитационного моделирования, а также проведен сравнительный анализ данных инструментов. В результате был сделан выбор в пользу AnyLogic. Кроме того, была поставлена задача и были описаны основные процессы транспортной компании, которые реализованы в имитационной модели, описание которой будет представлено далее.

# **Глава 3. Программная реализация**

## **3.1 Условные обозначения модели**

|  |  |  |
| --- | --- | --- |
| Обозначение | Единица измерения | Расшифровка |
| На объекте main: | | |
| delivery\_quantity | Шт. | Количество мест доставки |
| department\_quantity | Шт. | Количество складов |
| numberOfTrucks | Шт. | Количество грузовиков |
| department\_location |  | Коллекция гис-точек, отвечающих за положение складов на карте |
| delivery\_location |  | Коллекция гис-точек, отвечающих за положение мест доставки на карте |
| reject | шт | Коллекция заказов, доставка которых не производится |
| success | шт | Коллекция заказов, доставка которых успешно выполнена |
| timeOut | шт | Коллекция заказов, доставка которых не выполнена в следствии превышения срока ожидания |
| На объекте delivery: | | |
| People |  | Накопитель, отражающий общее количество людей. |
| PotentialClients |  | Накопитель, отражающий количество потенциальных клиентов. |
| Clients |  | Накопитель, отражающий количество клиентов. |
| A | - | Случайная величина от -40 до 40 |
| T | ч | Модельное время |
| SignedClients | - | Накопитель, отражающий количество людей, подписавших контракт. |
| logEff | - | Эффективность логистического отдела |
| lawEff | - | Эффективность юридического отдела |
| marketEff | - | Эффективность отдела маркетинга |
| selfProb | - | Вероятность обращения в логистическую компанию самостоятельно, выраженная в долях |
| TotalClients |  | Накопитель, итоговое количество людей, сделавших заказ. |
| Demand |  | Параметр, отвечающий за спрос. |
| omega |  | Параметр, заданный нормальным распределением. |
| sumEff |  | Показатель, отражающий эффективность логистического и юридического отделов. |
| timeToNext | ч | Время до следующего заказа |
| На объекте Department: | | |
| product\_stock |  | Список продуктов и их количество доступное на складе. |
| На объекте Order: |  |  |
| order\_delivery |  | Хранит значение места куда будет доставлен заказ. |
| order\_department |  | Хранит значение места откуда заказ будет забран. |
| product\_needs |  | Содержание заказа. |
| maxTime | ч | Максимальное время ожидания заказа. |
| createTime | ч | Время создания заказа. |
| боstatus |  | Статус заказа. |
| truck |  | Грузовик, который будет осуществлять перевозку. |
| На объекте Product: | | |
| Name |  | Название продукта |

Таблица 2. Обозначения модели.

## **3.2 Объекты модели**

Объект Main.

Это агент верхнего уровня. На нем располагаются все другие агенты модели. Представляет из себя Гис-карту, которая иллюстрирует эксперимент. На карте отмечены гис-точки, которые являются местами доставки. Кроме того, на агент Main вынесены коллекции reject, success, TimeOut, которые хранят в себе количество отклоненных, успешно доставленных, и не доставленных в срок заказов. На основе этих данных строятся графики, иллюстрирующие изменение данных показателей в динамике.

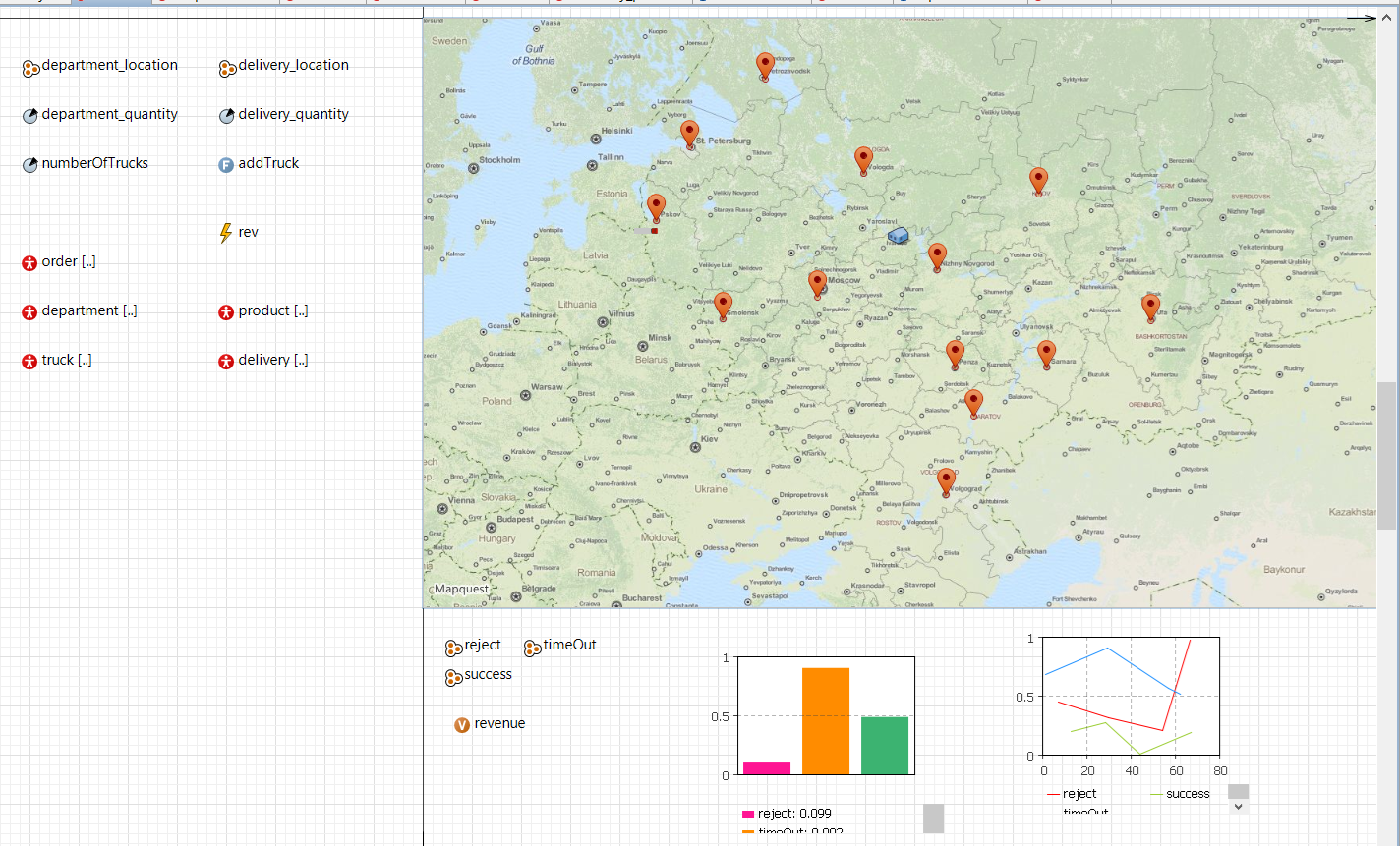


Рис. 10 агент main

Что касается маршрута, проделываемого автомобильным транспортом, то он строится, используя сервер AnyLogic. Так как речь идет о движении грузового автомобильного транспорта, то выбрана автомобильная сеть дорог.

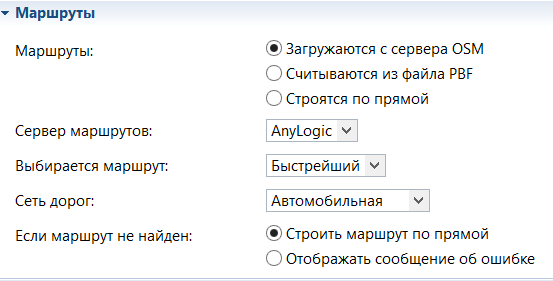


Рис. 11 Настройка построения маршрутов

При запуске модели выполняется следующий код:

**for (int i=0;i<department.size();i++) {**

**for (int j=0;j<product.size();j++) {**

**department.get(i).product\_stock.put(product.get(j).name, uniform\_discr(0, 200));**

**}**

**}**

В результате, при запуске модели каждый склад наполняется товарами, каждого продукта может иметься от 0 до 200 штук, число генерируется согласно равномерному дискретному распределению.

Агент Product

Имеет параметр name. Названия продуктов подгружаются из внешней базы данных. По указанным критериям.

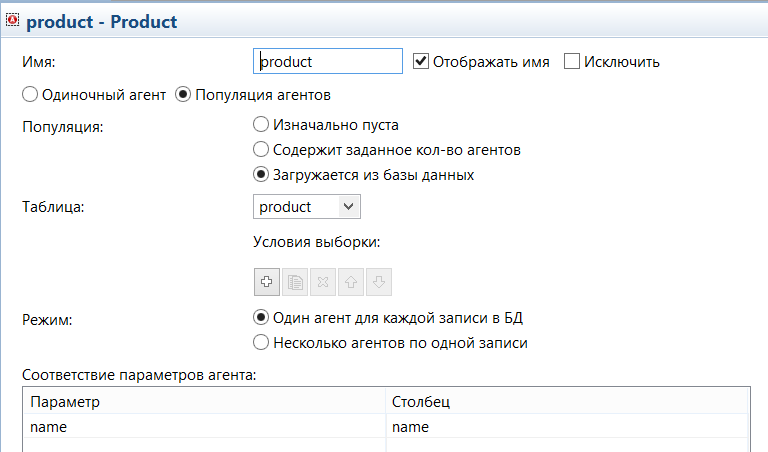


Рис. 12 Свойства агента product

Такой способ обращения к информации используются для обеспечения большей гибкости системы, так как в реальности товары могут изменяться. В данном случае базу данных представляет собой обычный Excel-файл, который можно редактировать, не открывая AnyLogic. Тем не менее, благодаря настройкам внутри имитационного пакета, значения, участвующие в модели, будут обновляться каждый раз перед ее запуском.

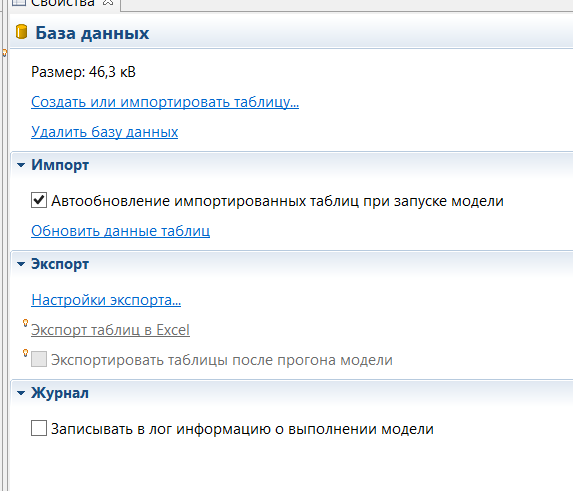


Рис.13 Использование базы данных

Объект Delivery

Данный агент является пунктом доставки. Именно здесь формируется заказ.

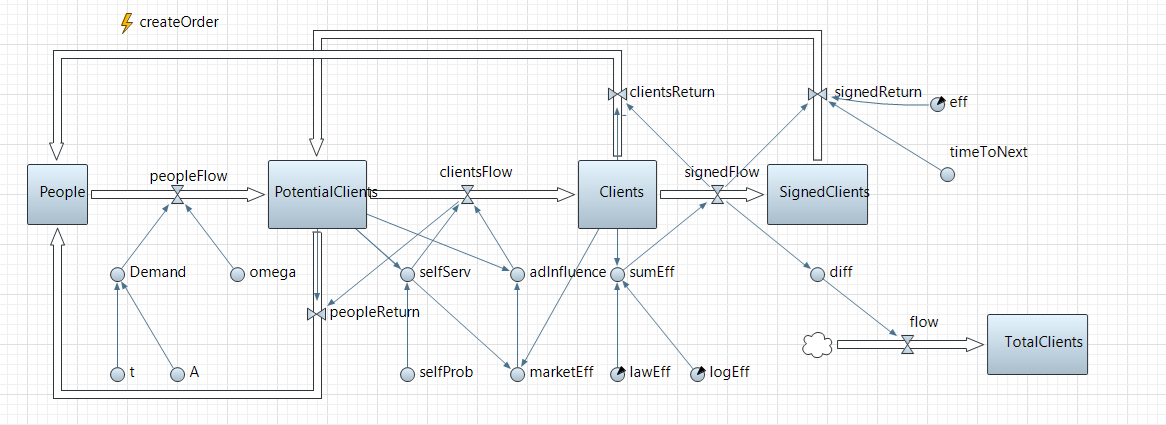


Рис. 14 Представление агента Delivery

Изначально рассматривается, что общее количество людей 100000. Однако далеко не все люди обратятся в компанию. Поэтому вводится динамическая переменная Demand, отвечающая за спрос. Спрос является не постоянной величиной, он носит сезонный характер и может колебаться. Поэтому вводится функция, описывающая его 1000+A\*Math.sin(PI/2\*t), где A- случайное число находится в промежутке [-40;40]. Однако это общий спрос на транспортные услуги всех компаний. По этой причине вводится переменная omega, представляющая вероятность обращения в данную компанию, в основе которой заложено нормальное распределение. Таким образом, что вероятность их обращения колеблется от 0,7 до 2%.

Произведение данной переменной на спрос формирует поток людей-peopleFlow, которые представляют из себя категорию потенциальных клиентов- PotentialClients. Далее начинается процесс формирования клиентов. Потенциальные клиенты могут обратится самостоятельно с вероятностью от 2,5 до 5%, заданную также нормальным распределением переменной selfProb. Произведение этой переменной на количество потенциальных клиентов задает переменную selfServ=PotentialClients\*selfProb. Кроме этого на потенциальных клиентов оказывает влияние маркетинг. Переменная marketEff вычисляется по формуле delay(Clients/PotentialClients,1,0.8). Которая означает, что переменная вычисляется как отношения числа клиентов к потенциальным клиентам с задержкой 1 модального времени, а в первый момент времени значение переменной равно 0,8. Динамическая переменная adInfluence= PotentialClients\*marketEff формирует объем людей, привлеченных рекламой.

Поток clientsFlow от накопителя PotentialClients к Clients формируется формулой (int)(adInfluence + selfServ). Что означает суммирование факторов самостоятельного обращения и воздействия маркетинга. Так как клиенты приходят или под влиянием первого, *или* под влиянием второго фактора. Далее на клиентов оказывают влияние логистический *и* юридический отделы, эффект рассчитывается по следующей формуле: sumEff =(Clients\*lawEff\*logEff) Данный показатель определяет поток signedFlow, который заполняет накопитель SignedClients.

От каждого накопителя уходят обратные потоки. Они (peopleReturn, clientsReturn) определяются разницей значения в накопители и выходящего потока.**peopleReturn=(int)(PotentialClients-clientsFlow);**

**clientsReturn=(int)(Clients-signedFlow).**

А возратный поток signedReturn определятся следующим образом: delay(eff\*signedFlow, timeToNext). Переменная timeToNext является промежутком перед повторным обращением клиента, подписавшего контракт и она имеет вид: normal(50, 60). Что говорит о том, что в среднем этот период составит 60 дней, однако возможно отклонение в 50 дней от указанного среднего. Параметр Eff определяется же отношением числа подписавших контракт клиентов к числу клиентов.

Окончательный накопитель TotalClients формируется за счет переменной diff, которая рассчитывается как разница накопителя SignedClients и значения этого же накопителя только в более ранний момент времени. Именно различие этого накопителя характеризует количество новых заказов.

Кроме того, переменная diff является условием для создания заказа. Как только она становится положительной отрабатывает событие createOrder.

В данном событии происходит следующее:

**int k = uniform\_discr(1, main.product.size());**

**Map<String, Integer> map = new LinkedHashMap<String, Integer>();**

**ArrayList<Integer> productCode = new ArrayList<Integer>();**

**ArrayList<Department> goodDepart = new ArrayList<Department>();**

**int t = uniform\_discr(30, 300);**

**for (int i=0; i<main.product.size()-1; i++) {**

**productCode.add(new Integer(i));**

**}**

**Collections.shuffle(productCode);**

**for (int i=0;i<k;i++) {**

**String nextProduct = main.product.get(productCode.get(i)).name;**

**map.put(nextProduct, uniform\_discr(1, 20));**

**}**

Происходит заполнение заказа позициями из возможных продуктов. Количество самих позиций выбирается рандомно. Затем каждому из выбранных продуктов формируется их количество от 1 до 20, с помощью равномерного дискретного распределения.

**boolean check = true;**

**for (Department d: main.department) {**

**for (Map.Entry e : map.entrySet()) {**

**String prod=(String)e.getKey();**

**int quant=(int)e.getValue();**

**if (d.product\_stock.get(prod)<quant) {**

**check = false;**

**break;**

**}**

**}**

**if (check) {**

**goodDepart.add(d);**

**}**

**}**

После этого происходит проверка складов на которых имеется достаточное количество товара. Формируется список таких складов.

**if (goodDepart.size()>0) {**

**Department depart = getNearestAgentByRoute(goodDepart);**

**for (Map.Entry e : map.entrySet()) {**

**String prod=(String)e.getKey();**

**int quant=(int)e.getValue();**

**depart.product\_stock.put(prod,depart.product\_stock.get(prod)-quant);**

**}**

Затем, имея список складов с необходимым числом товара, в нем начинается поиск ближайшего по расстоянию. И имеющееся количество товара на складе уменьшается на указанное в заказе.

**Order order = main.add\_order(this,depart,map,t,time(),1,null);**

**Truck tr = order.findTruck();**

**tr.orderList.add(order);**

Создается новый агент типа order с параметром delivery соответствующему данному месту (логика формирования заказа определенна именно в агенте, который является местом доставки). Параметр depart определился на предыдущем шаге как ближайший с нужным количеством товара. Кроме того, в агенте order запускается метод findTruck, который будет подробно описан позднее. И для найденного, таким образом, грузовика в его список заказов, который он должен доставить добавляется еще один.

**} else {**

**Order order = main.add\_order(this,null,map,t,time(),0,null);**

**main.reject.add(order);**

**}**

Если ж не нашлось складов с достаточным количеством товара, то заказ добавляется в список reject.

Агент Order

Для реализации логики этого агента была выбрана диаграмма действий.

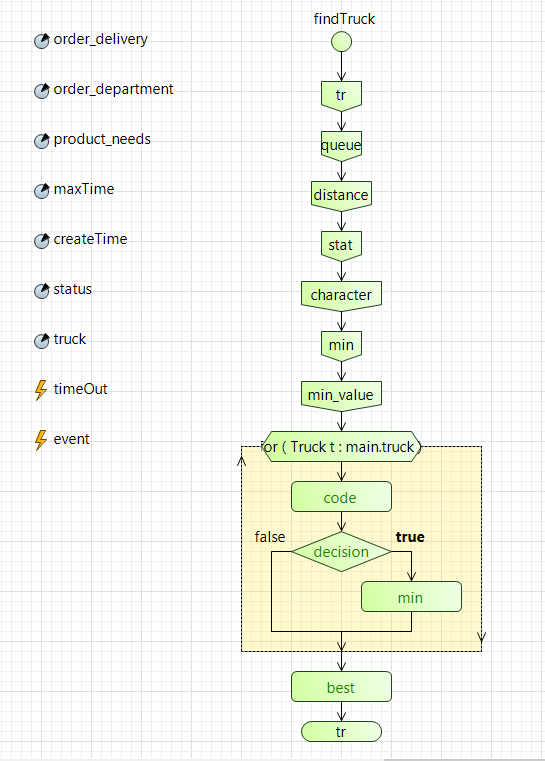


Рис. 15 Агент Order

Основной задачей, которую решает данная диаграмма является поиск подходящего транспортного средства.

Изначально описываются переменные, которые в последующем будут участвовать в цикле поиска. Первая переменная tr, которая и будет являться конечным возвращаемым значением, хранит в себе значение искомого экземпляра класса Truck. Следующая переменная queue является массивом типа int размерности main.truck.size() и хранит количество заказов, ожидающих доставки для каждого отдельного грузовика. distance массив типа double с размерностью также равному количеству грузовиков. Хранит в себе значения расстояний от каждого грузовика до склада. Переменная stat хранит значение статуса, определяемого в какой точки маршрута, находится грузовик. character, тип double[main.truck.size()], служит для присвоения каждому транспортному средству характеристики, по которой будет производиться сравнение. Переменная min имеет целочисленный тип и хранит порядковый номер грузовика, в результате прохождения цикла. Min\_value тип double хранит характеристику выбранного грузовика.

Цикл for проходит по всем элементам класса Truck.

Внутри цикла располагается элемент code, который имеет следующее содержание:

**queue[t.getIndex()]=t.orderList.size();**

Переменная queue приравнивается количеству заказов

**distance[t.getIndex()]=t.distanceByRoute(order\_department)/1000;**

Distance становится равна расстоянию до склада в км.

**if (t.inState(Truck.AtDepartment)||t.inState(Truck.Unpacking\_dep)) {**

**stat[t.getIndex()]=1;**

Если грузовик находится на складе, либо происходит распаковка заказа, в случае, когда после его упаковки было обнаружено, что заказ не будет доставлен вовремя, то статусу присваивается значение 1.

**} else if (t.inState(Truck.MovingToDepartment)) {**

**stat[t.getIndex()]=2;**

Если же грузовик после распаковки возвращается на склад, то ему приписывается статус 2.

**} else if(t.inState(Truck.Unpacking)) {**

**stat[t.getIndex()]=3;**

Статус становится равным 3, в случае распаковки заказа для покупателя.

**} else {**

**stat[t.getIndex()]=4;**

**}**

Во всех прочих расположениях грузовика: когда он движется от текущего склада, к нужному, где имеется товар, когда грузовик находится в состояние упаковки товара, когда он движется к месту доставки- статус равен 4.

**character[t.getIndex()] = distance[t.getIndex()] + queue[t.getIndex()] + stat[t.getIndex()];**

После этого грузовику присваивается характеристика равная сумме дистанции очереди и статусу заказа.

Далее выставляется условие: **character[t.getIndex()]<min\_value**

Сравнивающее характеристику с минимальным значением, если оно верно, то выполняются следующие действия:

**min=t.getIndex();**

**min\_value=character[t.getIndex()];**

Переменной min присваивается порядковый номер транспортного средства, а min\_value значение параметра character данного грузовика. А так как все это происходит в цикле, то будет выбран грузовик с наименьшим значением character. Именно ссылка на это транспортное средство будет передана в переменную tr, которая является возвращаемым значением. Таким образом, для каждого заказа будет подобрано наиболее оптимальный перевозчик.

Также в объекте заказ присутствует событие, которое контролирует заказы по времени исполнения, условие для срабатывания выглядит следующим образом: **(int)(time()-createTime)>=maxTime**, событие носит название timeOut и при срабатывании выполняет:

**main.timeOut.add(this);**

**set\_status(2);**

То есть, добавляет в коллекцию заказов, не удовлетворяющих временные рамки, на агенте main и меняет статус заказа.

Агент Truck

Логика данного агента отображена на диаграмме состояний.

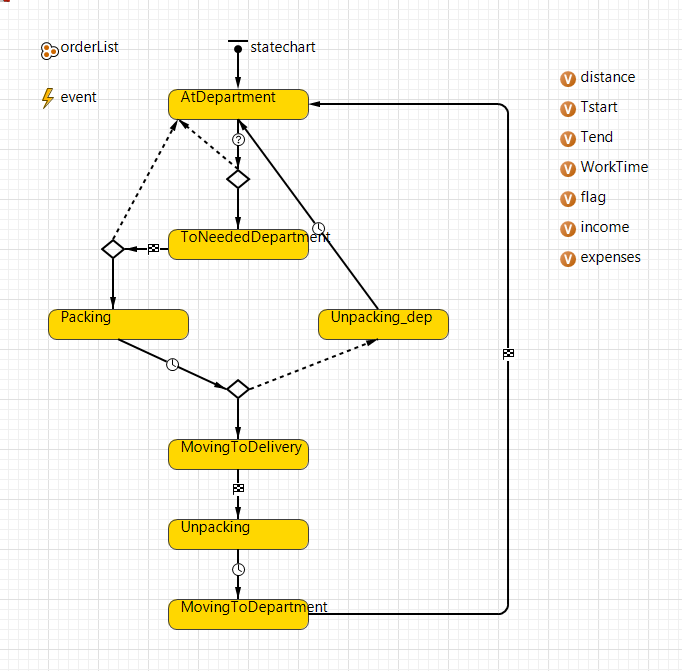


Рис.16 агент Truck

Процесс доставки состоит из последовательности состояний и связей между ними. Началом и концом движения грузовика является его пребывание на складе. Это состояние обозначено как AtDepartment. При выходе из этого состояния происходит следующее:

**Tstart=time();**

**if (flag==false)**

**{income=income+1000;**

**distance=0;}**

**flag=false;**

Фиксируется время начала движения. Осуществляется проверка переменной flag, которая отвечает за факт доставки. При первом проходе значении по умолчанию будет false. А это означает, что в наш доход мы получим 1000 - сумму не возвратного залога. Переменной, отвечающей за суммарное количество км потраченным на доставку и возвращение, изначально присваивается значения 0. Flag=false является сбросом значения булевой переменной.

Далее происходит проверка на условие orderList.size()>0, то есть в агенте заказа отработала описанная ранее логика, находящая подходящее транспортное средство, и этому грузовику был назначен заказ.

Если условие выполняется, то он переходит на следующий этап проверки: **orderList.get(0).status==1** Который означает, что статус ожидает исполнения, а не отменен по TimeOut, описанного выше для агента Order. (В этом случае статус был бы равен 2)

Если условие не выполняется, то происходит:

**for (Map.Entry e : orderList.get(0).product\_needs.entrySet()) {**

**String prod=(String)e.getKey();**

**int quant=(int)e.getValue();**

**orderList.get(0).order\_department.product\_stock.put(prod,orderList.get(0).order\_department.product\_stock.get(prod)+quant);**

**}**

**orderList.remove(0);**

**expenses=expenses+2500;**

А именно на склад с которого уже успел списаться товар, он возвращается, а сам список заказов очищается. В затраты добавляется размер штрафа 2500, выплачиваемый за недоставку груза. Грузовик же возвращается на исходный склад.

Если же условие выполнилось, то выполняется действие: **distance=this.distanceByRoute(orderList.get(0).order\_department)/1000;**

**moveTo(orderList.get(0).order\_department);**

Рассчитывается значение в км до нужного склада (может оказаться, что грузовик уже находится на нужном складе). И направляется на склад, ссылка на который находится внутри агента Order. Таким образом, транспортное средство оказывается в состоянии ToNeededDepartment. Если к тому времени как грузовик достиг нужного склада лимит времени будет исчерпан, то опять же происходит возврат списанного товара, а транспортное средство возвращается в исходную точку. В противном случае происходит переход в состояние Packing, отражающее упаковку товара, как следствие переход от этого состояния занимает время пропорциональное количеству товара. Далее происходит очередная проверка по тайм-ауту, однако грузовик не сразу возвращается на исходный склад, а сначала протекает процесс распаковки товара. Если же со сроками доставки все хорошо, то выполняются следующие действия: **distance=distance+this.distanceByRoute(orderList.get(0).order\_delivery)/1000;**

**moveTo(orderList.get(0).order\_delivery);**

То есть параметр distance к уже проделанному пути между складами добавляет расстояние от склада товара до места доставки, и происходит движению к заказчику. Таким образом, достигается состояние MovingToDelivery. При выходе из этого состояния переменной flag присваивается значение true, что свидетельствует о факте доставки. Если грузовик прибыл к месту доставки в момент, когда временное ограничение закончилось, процесс разгрузки и доставки будет продолжен, однако со штрафом для компании: if **(orderList.get(0).status==2)**

**expenses=expenses+distance\*10;** в затраты уходит 50% от тех денег, что будут получены за доставку. Происходит переход в состояние распаковки, по окончании этого действия переменная distance в которой происходит подобное вычисление **distance+this.distanceByRoute(getNearestAgent(main.department))/1000,** свидетельствующее о фиксировании в данной переменной еще и расстояние до склада.

По завершению распаковки происходит переход с затратой времени 0.0375\*orderList.get(0).product\_needs.values().stream().reduce(0, Integer::sum), это подтверждает зависимость времени распаковки от количества товара в заказе. Кроме того, в момент перехода, до наступления события MovingToDepartment происходит действие:

**main.success.add(orderList.get(0));**

**orderList.remove(0);**

**boolean bol = false;**

**int num=-1;**

**for (int i=0;i<orderList.size();i++) {**

**if (orderList.get(i).status==1) {**

**bol = true;**

**num=i;**

**break;**

**}**

**}**

В коллекцию success, отраженную на уровне main добавляется доставленный заказ. Из списка заказов, который должен доставить этот грузовик, удаляется первый элемент, в силу того, что он доставлен. Из списка оставшихся заказов происходит поиск тех, которые не просрочены по времени. Если хотя бы один заказ такой находится, то переменная num запоминает его порядковый номер из списка, а булевая переменная bol принимает значение true. Если такового заказа не будет, то bol сохранит значение false, а переменная num будет равна -1.

**if (bol==true) {**

**for (int i=0;i<num;i++) {**

**for (Map.Entry e : orderList.get(i).product\_needs.entrySet()) {**

**String prod=(String)e.getKey();**

**int quant=(int)e.getValue();**

**orderList.get(i).order\_department.product\_stock.put(prod, orderList.get(i).order\_department.product\_stock.get(prod)+quant);**

**}**

**}**

**for (int i=0;i<num;i++) {**

**orderList.remove(0);**

**}**

**moveTo(orderList.get(0).order\_department);**

Если заказ нашелся, то происходит возврат товаров на склады для просроченных заказов, удаления заказов до позиции текущего, и отправление на первоначальный склад.

**} else {**

**for (int i=0;i<orderList.size();i++) {**

**for (Map.Entry e : orderList.get(i).product\_needs.entrySet()) {**

**String prod=(String)e.getKey();**

**int quant=(int)e.getValue();**

**orderList.get(i).order\_department.product\_stock.put(prod, orderList.get(i).order\_department.product\_stock.get(prod)+quant);**

**}**

**}**

**orderList.clear();**

**moveTo(getNearestAgent(main.department));**

**}**

Если среди всех таких заказов не нашлось ни одного у которого время заказа еще не просрочено, то происходит возврат товаров на склады и удаления всего списка заказов и отправление на ближайший склад.

По прибытию на склад срабатывает логика:

**Tend=time();WorkTime=WorkTime+Tend-Tstart;**

**expenses=WorkTime\*200;**

**if (flag==true)**

**{income=income+distance\*20;**

**distance=0;}**

Согласно которой переменная Tend фиксирует окончание поездки, WorkTime рассчитывает время, затраченное на доставку грузов, происходит расчёт затрат expenses из ставки 200 р/ч за труд водителя, если груз был доставлен flag==true, то переменная доходов income увеличится из расчёта, что один километр движения стоит 20 рублей. И переменная расстояния distance обнулится. Остальные действия будут следовать логики описанной ранее, так как мы вернулись в исходное состояние. Говоря о финансах, стоит отметить, что переменная income отвечает за доходы, а expenses за расход. Может возникнуть ситуация, когда доходы превысят доходы. Это возможно в ситуации, когда ограничение по времени пройдет, а заказ не будет доставлен, в таком случае оплата водителям будет понесена, а покупателю будет выплачен штраф.

Также на верхнем уровне - на агенте main отражается переменная revenue, заполняемая событием rev

**revenue=0;rev2=0;**

**for (int i=0;i<truck.size();i++)**

**{revenue=revenue+truck.get(i).income-truck.get(i).expenses;**

**revenue=revenue-department.size()\*1000/24\*time();//** прибыль с учетом постоянных затрат на содержание склада

Расчет производится суммарный по всем грузовикам. Сначала вычитаются все переменные издержки, зависящие непосредственно от каждого заказа. А потом вычитается и расходы на содержание складов. Считается, что содержание склада в среднем обходится в 1000 рублей в день. Так как мы работаем с моделью, для которой модельным временем являются часы, а наше событие по расчету прибыли отрабатывает каждый час, то чтобы найти затраты в конкретный период времени, то необходимо 1000/24\*time(), а так как склад у нас ни один, то умножаем на их количество **department.size()**.

Также в ходе построение модели были реализованы оптимизационные задачи:

1. Максимизация объема прибыли при изменении количества складов
2. Максимизация прибыли при варьировании количества грузовиков
3. Максимизация прибыли при переменных значениях обоих вышеперечисленных параметров.

В первом эксперименте берется дискретный тип параметра department\_quantity со значениями от 1 до 6 и шагом 1. Максимизируется значение параметра revenue.

При второй оптимизации вариативные значения принимает параметр numberOfTrucks, в диапазоне от 1 до 20 с шагом 1.

В последнем же случае, оба параметра: число складов и число грузовиков становятся переменными значениями. И ищется максимальная достижимая прибыль при различных доступных комбинациях.

## **3.3 Результаты работы модели**

В данной модели использовано 6 агентов, 14 параметров, 10 переменных и 6 событий. Полученная модель отвечает всем требованиям транспортной логистики, перечисленным в главе 1, подразделе 1.1. Данная модель доставляет именно требуемый товар, для этого была введена проверка на складах достаточного количества товара, указанного в заказе. Она происходит на объекте Delivery в событии createOrder, описанном в подразделе 3.2. Транспортное средство движется по наиболее оптимальному маршруту, для этого, во-первых, выбирается подходящий грузовик, согласно логики описанной для объекта Order в подразделе 3.2 Во-вторых, в событии createOrder, упомянутом выше, также происходит выбор ближайшего склада с доступным количеством товара, а не произвольного, кроме того после доставки заказа, грузовик возвращается на ближайший склад, что указано в диаграмме событий агента Truck в подразделе 3.2. Из чего следует, что движение его будет совершаться по оптимальному маршруту. Кроме того, он будет доставлен именно в ту точку, откуда поступил заказ. Это условие достигается самой логикой агента Order, хранящего значение места доставки как параметр order\_delivery, и с помощью встроенного функционала AnyLogic, способного достаточно точно осуществлять движение по автомобильным дорогам.

В процессе выполнения работы значения параметров изначально устанавливались согласно общей информации касательно бизнес-процессов транспортной компании, после была выполнена калибровка, в ходе которой значения уточнялись, таким образом, чтобы результаты работы модели в целом соответствовали реальным показателям транспортной компании.

Агент Delivery, реализуемый с помощью системной динамики, содержит накопитель People=100000, что соответствует среднему числу жителей в выбранных городах, функция спроса имеет вид: 1000+A\*Math.sin(PI/2\*t), где a принимает значения от -40 до 40, согласно нормальному распределению. Ее вид свидетельствует о наличии незначительных колебаний спроса. Оценки эффективности логистического и юридического отделов равные 0,6462 и 0,7252 соответственно были выведены в результате калибровки модели, при значении данных показателей происходит генерация реалистичного количества заказов в день. Параметры department\_quantity, numberOfTrucks, deliver\_quantity соответствующие количеству складов, количеству грузовиков и количеству мест доставок были взяты произвольно, однако на основание именно этих параметров решаются оптимизационные задачи, приводящие к максимизации прибыли при изменении этих значений.

По мере выполнения ряда экспериментов было введено ограничение на время. Рассматриваются 48 часов деятельности компании. В ходе нескольких экспериментов было установлено, что в целом компания имеет положительную прибыль, которая, однако, может значительно колебаться в зависимости от дальности перевозок заказов. Имея 6 складов, 7 грузовиков и 12 потенциальных точек доставок, в целом за 2 дня компания успевает доставить от 4 до 7 заказов, тем не менее, кроме доставленных заказов возникают и неуспешные: reject – в следствии отсутствия нужных товаров на складе и timeOut в следствии превышения срока доставки. На основании этих данных собирается статистика и отражаются графики, иллюстрирующие количество невыполненных заказов в следствии как отсутствия товара, так и несоблюдения сроков и выполненных заказов.

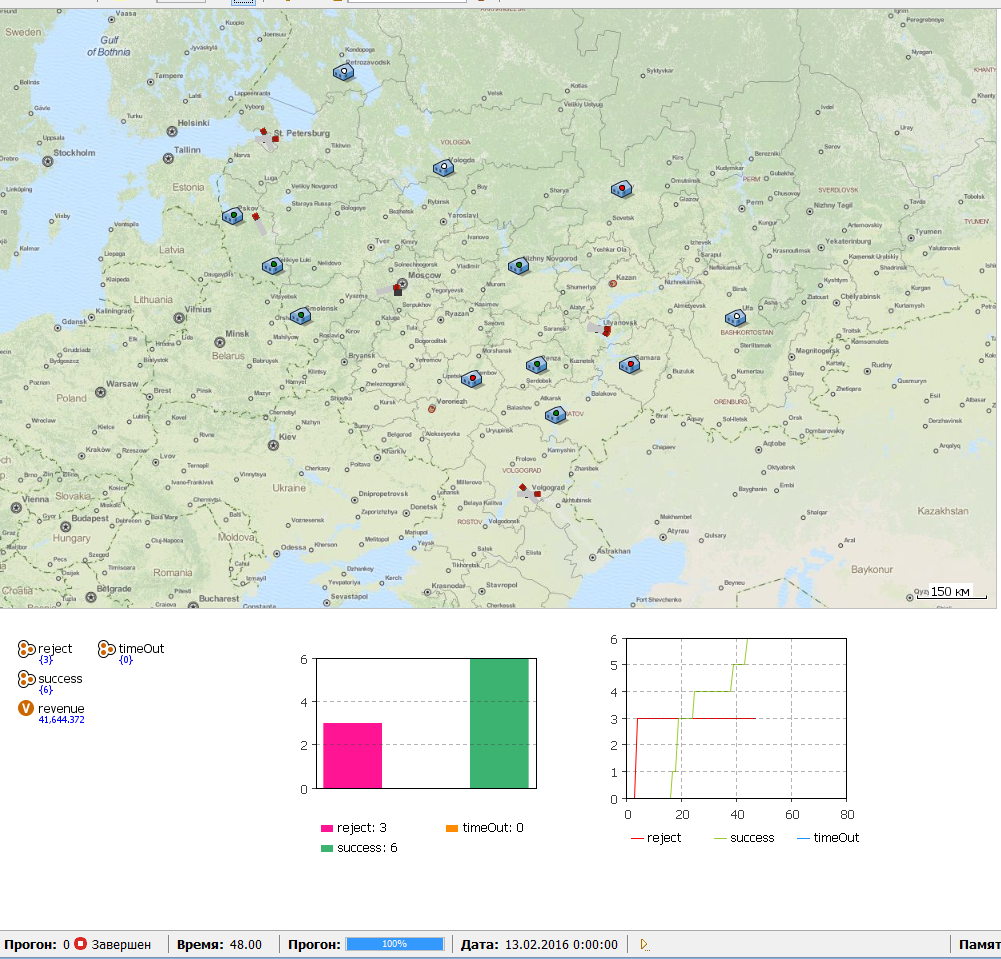


Рис. 17 Результаты эксперимента

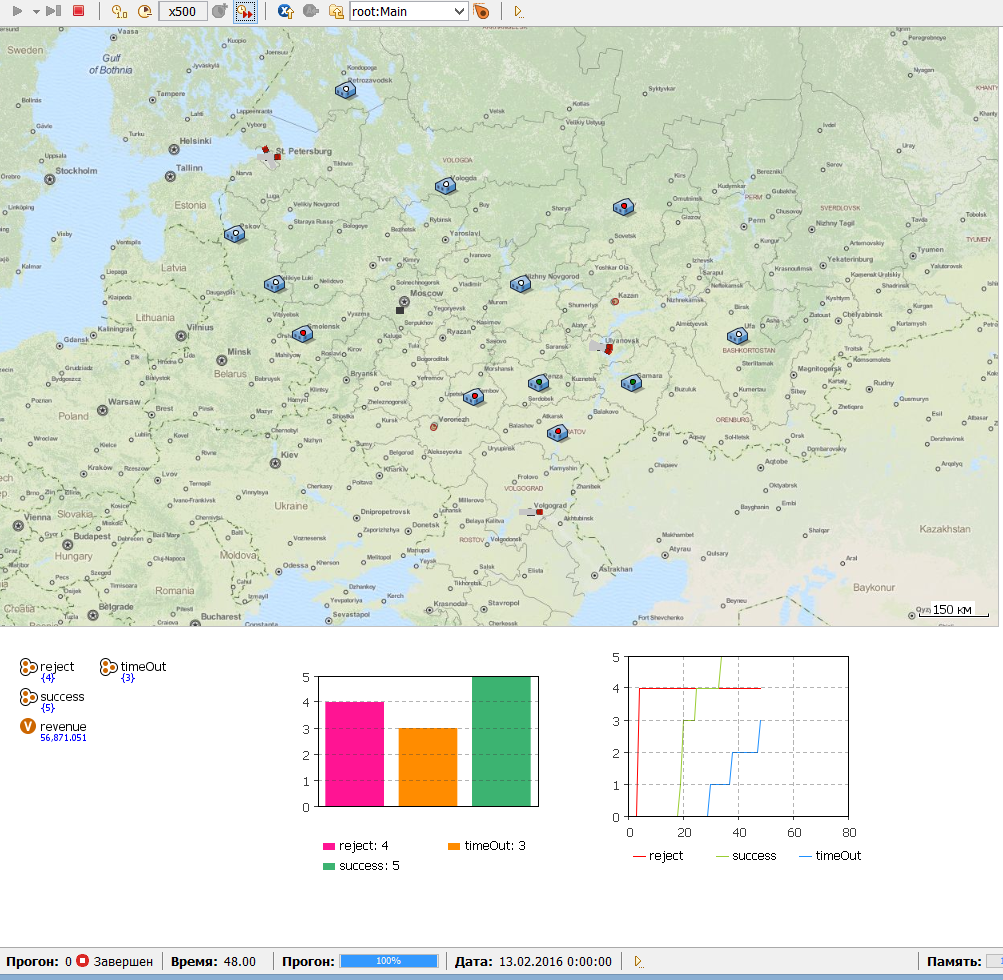


Рис. 18 Результаты эксперимента

Кроме этого, в процессе реализации модели были решены оптимизационные задачи.

Рассмотрим первую из них. Цель первого эксперимента: максимизировать значение переменной revenue, отвечающей за прибыль, при изменении числа складов. Стоит понимать, что каждый новый склад с одной стороны, в целом, сокращает время доставки товара, как следствие, увеличивает количество заказов, которые можно привести за час работы, что положительно сказывается на прибыли; с другой стороны, каждый новый склад стоит дополнительного содержания со стороны компании, что негативно отражается на прибыли. Так как мы генерируем не бесконечное количество заказов, то не имеет смысла рассматривать большое количество складов. Количество складов изменяется от 1 до 6. В результате эксперимента максимум прибыли составил: 46432, который достигается при 6 складах при неизменном числе грузовиков:7 и мест доставки:12.

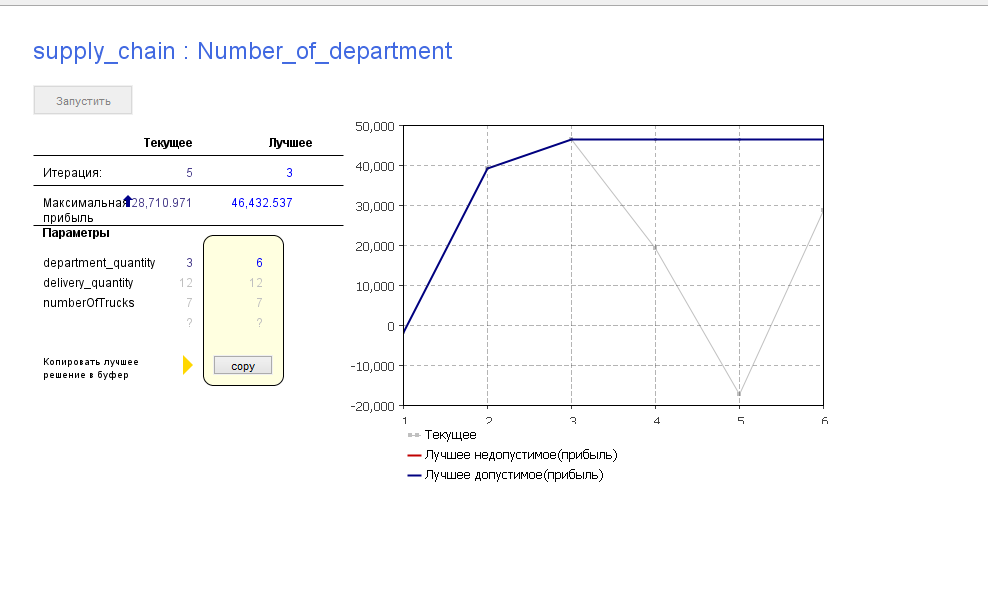


Рис. 19 Оптимизационная задача по максимизации прибыли при изменяемом числе складов

Вторая оптимизационная задача решает проблему максимизации прибыли при изменяемом размере автопарка. Число грузовиков варьируется от 1 до 20. Максимум прибыли 66389 рубля достигается при 10 грузовиках. Стоит отметить, что варьирование автопарка позволяет достичь большой прибыли, чем изменение количества складов в заданных приделах.

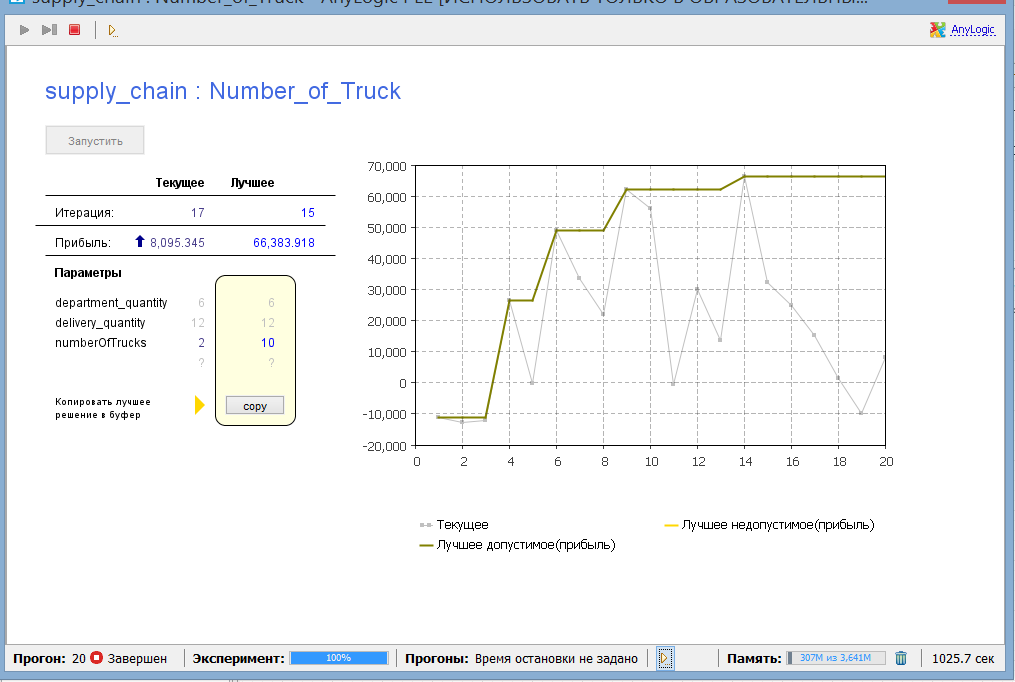


Рис. 20 Оптимизационная задача по максимизации прибыли при изменяемом размере автопарка

Последняя оптимизационная задача меняет два параметра: число складов и количество грузовиков с той же целью. Стоит заметить, что у этих показателей существует обратная зависимость в рамках реализации данной задачи. Это означает, что чем больше у нас становится складов, тем меньше нам нужно грузовиков, так как число заказов в целом не изменяется, так как оно зависит от количества мест доставки-delivery\_quantity, а этот параметр не изменен. Так же верно и то, что увеличение грузовиков ведет к уменьшению числа требуемых складов. Поэтому это наиболее интересная задача, однако при вариации нескольких параметров, число доступных сочетаний значительно возрастает. Всего при вариации складов от 1 до 6 и грузовиков от 1 до 20, число возможных комбинаций равно 120. В процессе работы эксперимента были рассмотрены все возможные комбинации, при 48 часах работы компании. Максимум прибыли: 68139 был достигнут при 2 складах и 11 грузовиках. Сравнивая полученный результат с первым оптимизационном экспериментом, стоит отметить, что прибыль выросла практически на 50% за счет значительного расширения автопарка. По сравнению же со вторым оптимизационном экспериментом прибыль выросла незначительно, так как возможность варьировать количество грузовиков уже обеспечила значительное изменение прибыли второму эксперименту. Однако, увеличение прибыли наблюдается. Достигается оно за счет сокращения числа складов, и как следствие расходов на их содержание, и увеличения автопарка, которое компенсирует уменьшение складов.

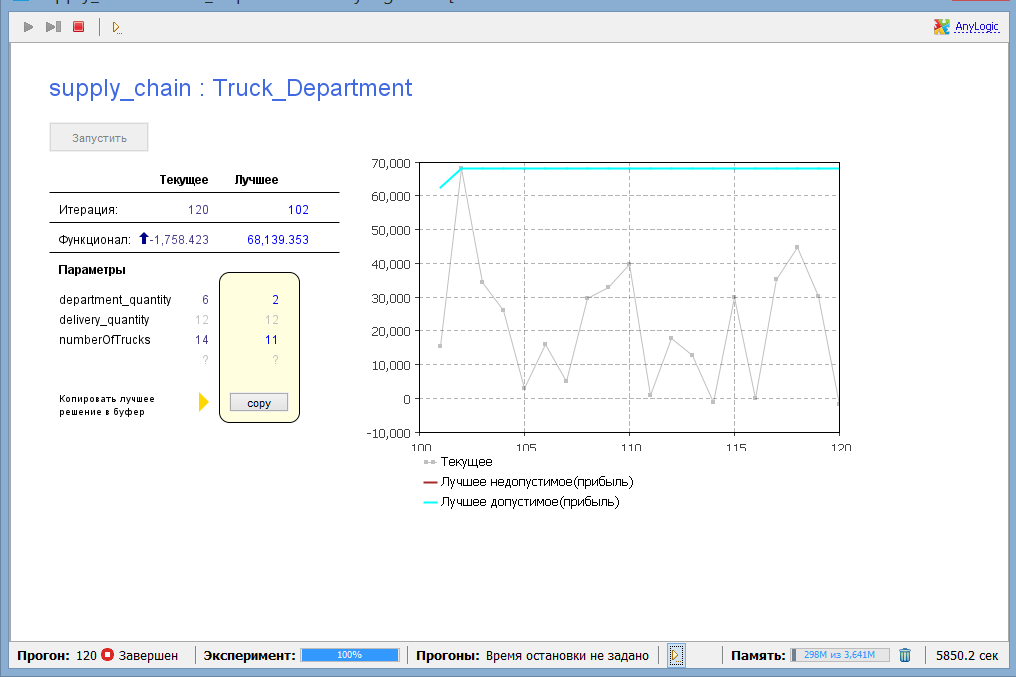


Рис. 21 Оптимизационная задача по максимизации прибыли при изменяемом числе складов и грузовиков

**Выводы:**

В данной главе были описаны агенты, использованные в модели, а также их элементы: параметры, события, переменные, диаграммы, элементы процесса, накопители. Была описана логика функционирования каждого отдельного элемента, также было представлено описание модели в целом- объяснение ее функционирования. В конце данной главы были приведены результаты функционирования модели и даны оценки основным показателям. А также были реализованы оптимизационные эксперименты, решающие проблему максимизации прибыли компании, и описаны результаты, полученные в рамках решения данных задач.

## **Заключение**

Результатом данной работы является функционирующая имитационная модель, включающая в себя симбиоз всех трех парадигм имитационного моделирования.

Среди ключевых особенностей модели можно отметить следующие:

* В основе генерации заказов учитывается работа отделов компании.
* Процесс повторного обращения клиентов является не случайным событием, а также зависит от эффективности работы подразделений.
* Транспортное средство подбирается с учетом построения оптимального маршрута.
* Осуществляются проверки на наличие товара на складе при формировании заказа.
* Собирается статистика по успешно выполненным, отклоненным в следствии отсутствия товара на складах и отклоненным по временным ограничениям заказам.
* Кроме того, на основе данной модели можно получить оценку финансовой деятельности компании.
* И решить оптимизационные задачи по максимизации прибыли в зависимости от количества имеющихся грузовиков и количества складов.

## **Список используемой литературы**

1. *Акопов А.С.* Имитационное моделирование. Учебник и практикум для академического бакалавриата. - М.: Издательство Юрайт, 2014. – 389 с.
2. *Аникин Б. А.* Логистика: Учебник-М.:ИНФРА-М, 2008.-368 с.
3. *Бауэрсокс Доналд Дж., Клосс Дейвид Дж.* Логистика: интегрированная цепь поставок. 2-е изд./[Пер. с англ. Н.Н. Барышниковой, Б.С. Пинскера]/Доналд Дж. Бауэрсокс., Дейвид Дж. Клосс. – М.: ЗАО «Олимп-Бизнес», 2008. – 640 с.
4. *Бережная Е.В., Бережной В.И.* Математические методы моделирования экономических систем: Учебное пособие. –М.: Финансы и статистика, 2005. -426 с.
5. *Боев В. Д.* Исследование адекватности GPSS World и AnyLogic при моделировании дискретно-событийных процессов. Монография— СПб.: ВАС, 2011. — 404 с.
6. *Боев В. Д.* Компьютерное моделирование: Пособие для практических занятий, кур-сового и дипломного проектирования в AnyLogic7:. — СПб.: ВАС, 2014. — 432 с.
7. *Бронов С.А.* Имитационное моделирование: учеб. Пособие –Красноярск: СФУ, 2007-82 с.
8. *Гинзбург А.И*. Экономический анализ: Предмет и методы. Моделирование ситуаций. Оценка управленческих решений: учебное пособие. –СПб.: Питер, 2003. -622 с.
9. *Грекул В.И., Денищенко Г.Н., Коровкина* *Н.Л.* Проектирование информационных систем: Учеб. пособие. - М.: Интернет-Ун-т Информ технологий, 2005. – 304 с.
10. *Замятина О. М.*Моделирование систем: Учебное пособие. – Томск: Изд-во ТПУ, 2009. – 204 с.
11. *Емельянов, А. А.* Имитационное моделирование экономических процессов: учеб. пособие / А. А. Емельянов, Е. А. Власова, Р. В. Дума ; под ред. А. А. Емельянова. –М.: Финансы и статистика, 2002. – 368 с.
12. *Еремеева Л.Э.* Логистика: учебно-методический комплекс / Л.Э. Еремеева. – Сыктывкар: СЛИ, 2008. – 96 с.
13. [*Кнут*](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%B4_%D0%9A%D0%BD%D1%83%D1%82) *Д.,*[*Грэхем*](https://ru.wikipedia.org/w/index.php?title=%D0%A0%D0%BE%D0%BD%D0%B0%D0%BB%D0%B4_%D0%93%D1%80%D1%8D%D1%85%D0%B5%D0%BC&action=edit&redlink=1) *Р.,* [*Паташник*](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B5%D0%BD_%D0%9F%D0%B0%D1%82%D0%B0%D1%88%D0%BD%D0%B8%D0%BA) *О.* [Конкретная математика. Основание информатики.](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) — М.: [Мир](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D1%80_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)); [Бином. Лаборатория знаний](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%B8%D0%BD%D0%BE%D0%BC._%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%B8%D1%8F_%D0%B7%D0%BD%D0%B0%D0%BD%D0%B8%D0%B9&action=edit&redlink=1), 2006. — 703 с.
14. *Кобелев Н. Б.* Основы имитационного моделирования сложных экономических систем: Учеб. пособие. – М.: Дело, 2003. – 336с
15. *Курочкин, Д. В.* Логистика: [транспортная, закупочная, производственная, распределительная, складирования, информационная]: курс лекций / Д. В. Курочкин. – Минск: ФУАинформ, 2012. – 268 с.
16. *Лоу А.М., Кельтон В.Д.* Имитационное моделирование 3-е издание: –СПб.: Питер, Издательская группа BHV, 2004- 847c.
17. Лукьянов В. С., Слесарев Г. В. Проектирование компьютерных сетей методами имитационного моделирования: Учеб. Пособие – Волгоград: ВолгГТУ, 2001. - 72с.
18. *Лычкина Н.Н.* Имитационное моделирование экономических процессов – М.:Инфра-М, 2012 - 164с.
19. *Миротин Л.Б. и др.* "Эффективность логистического управления" Учебник для вузов / Под общ. ред. д. т. н., проф. Л.Б. Миротина. — М.: Издательство «Экзамен», 2004. — 448 с.
20. *Миротин Л.Б., Некрасов А.Г.* "Логистика интегрированных цепочек поставок": Учебник / Л.Б. Миротин, А.Г.Некрасов. М.: Издательство «Экзамен», 2003. — 256 с.
21. *Миротин Л.Б., Ташбаев Ы.Э*. "Системный анализ в логистике". - М.: Изд-во "Экзамен", 2002 - 480 с.
22. *Мичасова О.В.* Имитационное моделирование экономических систем: учеб. Пособие. - Нижний Новгород: Нижегородский университет, 2014 - 186 с.
23. *Смирнова Е.А.* Управление цепями поставок: Учебное пособие. /Е.А. Смирнова. – СПб.: Изд-во СПбГУЭФ, 2009. – 120 с.
24. *Томашевский В., Жданова E.* Имитационное моделирование в среде GPSS. – М.:Бестселлер, 2003. – 416 c.
25. *Уотерс Д.* Логистика. Управление цепью поставок: Пер. с англ./Д. Уотерс. – М.: ЮНИТИ-ДАНА, 2003. – 503 с. – (Серия «Зарубежный учебник»)
26. *Фёдоров Л. С., Персианов В. А., Мухаметдинов И. Б*. Общий курс транспортной логистики: учебное пособие по дисциплине специализации специальности "Менеджмент организации" – М.: КноРус, 2011. – 309 с.
27. *Шикин Е.В.* Математические методы и модели в управлении. –М.: Финансы и статистика, 2002. -430 с.
28. *Bart-Jan Hommes* The Evaluation of Business Process Modeling Techniques// TU Delft. (2004). p.272.
29. Алгол. [Электронный ресурс]. URL: <https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D0%BB>(Дата обращения: 23.04.16).
30. Боев В.Д. Оценка результатов моделирования в GPSS World и AnyLogic [Электронный ресурс]. URL: <http://www.anylogic.ru/articles/otsenka-rezultatov-modelirovaniya-v-gpss-world-i-anylogic> (Дата обращения: 16.03.16).
31. Борщев А. В. Применение имитационного моделирования в России – состояние на 2007 г. [Электронный ресурс]. URL: <http://simulation.su/uploads/files/default/immod-2007-1-11-16.pdf> (Дата обращения: 25.04.16).
32. Горелик А.М. Средства поддержки параллельных вычислений в стандартах языка Фортран // Препринты ИПМ им. М.В.Келдыша. 2012. № 68. 17 с. URL: <http://library.keldysh.ru/preprint.asp?id=2012-68>(Дата обращения: 01.05.16).
33. Каменнова М. Моделирование бизнеса. Методология ARIS. [Электронный ресурс]. URL: [http://flightcollege.com.ua/library/D1%8F%20ARIS.pdf](http://flightcollege.com.ua/library/6%20%D0%9E%D0%91%D0%A9%D0%95%D0%A1%D0%A2%D0%92%D0%95%D0%9D%D0%9D%D0%AB%D0%95%20%D0%9D%D0%90%D0%A3%D0%9A%D0%98/65%20%D0%AD%D0%9A%D0%9E%D0%9D%D0%9E%D0%9C%D0%98%D0%9A%D0%90/65.050%20%D0%9C%D0%95%D0%9D%D0%95%D0%94%D0%96%D0%9C%D0%95%D0%9D%D0%A2/%D0%9A%D0%B0%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B2%D0%B0%20%D0%9C%20-%20%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%B1%D0%B8%D0%B7%D0%BD%D0%B5%D1%81%D0%B0.%20%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F%20ARIS.pdf) (Дата обращения: 16.03.16).
34. Коровин А.М. Анализ подходов и программного обеспечения для имитационного моделирования социальных и экономических систем. [Электронный ресурс]. URL: <http://simulation.su/uploads/files/default/2012-korovin.pdf> (Дата обращения: 16.03.16).
35. Логистика: теория и методология. [Электронный ресурс]. URL: <http://www.topspeed.ru/page_18.php> (Дата обращения: 16.03.16).
36. Лычкина Н.Н. Современные технологии имитационного моделирования и их применение в современных бизнес-системах [Электронный ресурс]. URL:<http://www.anylogic.ru/upload/iblock/049/0498c3885e7d7b5dc8ac3dd4f261bca0.pdf> (Дата обращения: 16.03.16).
37. Практика построения логистических систем. [Электронный ресурс]. URL: <http://logistika.pp.ru/> (Дата обращения: 16.03.16).
38. Сидоренко В.Н. Имитационное моделирование в науке и бизнесе: подходы, инструменты, применение. [Электронный ресурс]. URL: <http://simulation.su/uploads/files/default/2009-sidorenko-krasnoselsky.pdf> (Дата обращения: 16.03.16).
39. Скородумов П.В. Имитационное моделирование экономических систем: программные средства и направления их совершенствования [Электронный ресурс]. URL: <http://simulation.su/uploads/files/default/2015-skorodumov-1.pdf> (Дата обращения: 16.03.16).
40. Терминологический словарь логиста. [Электронный ресурс]. URL: <http://www.slovalogista.ru/> (Дата обращения: 16.03.16).
41. Транспорт и логистика в России. «Что такое логистическая цепь?». [Электронный ресурс]. URL: <http://avitrack.net> (Дата обращения: 16.03.16).
42. BPMN-нотация [Электронный ресурс]. URL: <http://www.bpmn.org/>(Дата обращения: 20.04.16).