# Министерство образования и науки Российской Федерации

## Федеральное государственное бюджетное образовательное учреждение высшего образования

## «Новосибирский государственный технический университет»

NSTU_Logo_blue

## Кафедра теоретической и прикладной информатики

### Лабораторная работа № 4 по дисциплине «Компьютерное моделирование»

|  |  |  |
| --- | --- | --- |
| сигма градиент синий1 | Факультет: | ПМИ |
| Группа: | ПМИ-61 |
| Студенты: | Ершов П.К., Мамонова Е.В., Цыденов З.Б. |
|  |  |
| Преподаватель: | Черникова О. С. Карманов В. С. |

Новосибирск

2020

1. **Цель работы**

Научиться строить модели системы управления цепями поставок.

1. **Ход работы**

Модель цепи поставок основана на известной деловой игре «BeerGame» разработанной группой профессоров Массачусетского Технологического Института (MIT). Задача цепи поставок состоит в том, чтобы произвести и поставить конечному потребителю пиво: фабрика производит, а другие три звена цепи поставок продвигают пиво, пока оно не достигает конечного потребителя в конце системы поставок.

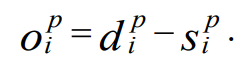
У каждого участника цепи есть метод оценки величины заказа, история заказов, текущий объём товара, товар, находящийся в пути (поставки). Сделанный кем-либо из участников заказ поступает на его склад через 1 период (в разработанной модели 1 неделя). Для упрощения задачи производитель изготовляет свою продукцию с такой же задержкой. Спрос на рынке является случайным (рассмотреть разные распределения).

За хранение товара на складе каждый участник цепи платит арендную плату. Расходы, связанные с содержанием сбытовых запасов – это упущенные доходы, которые могли быть получены, если бы деньги, вложенные в запасы, были использованы в других целях. Поскольку объём складских помещений предполагается постоянным, то расходы на хранение также не будут меняться. Кроме того, к потерям относятся излишки сбытовых запасов, имеющиеся сверх их минимально необходимого количества. Запасы, превышающие минимально необходимый уровень, не используются, что означает наличие упущенной выгоды, которая равна сумме прибыли, которая могла быть получена в случае размещения данных средств, например, на депозите в банке. Если на момент заказа у поставщика недостаточно товара, то он закупает его у конкурента по завышенной цене. Различие между собственной ценой и ценой конкурента является параметром задачи.

**Вариант:**

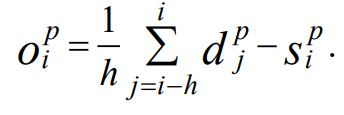
|  |  |  |
| --- | --- | --- |
| Номер варианта | Алгоритмы заказа | Тип структуры сети поставок |
| 2 | 1  2, h=3 | Линейная, 3 участника |

*Алгоритм № 1.*

Размер текущей закупки определяется заказом, который поступил последним. Это самая простая («наивная») стратегия, которая оптимальна при стабильном рынке: 

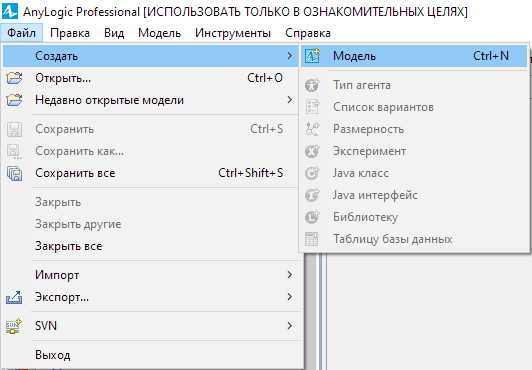
*Алгоритм № 2*.

Объём заказа определяется через скользящее арифметическое среднее порядка *h*:

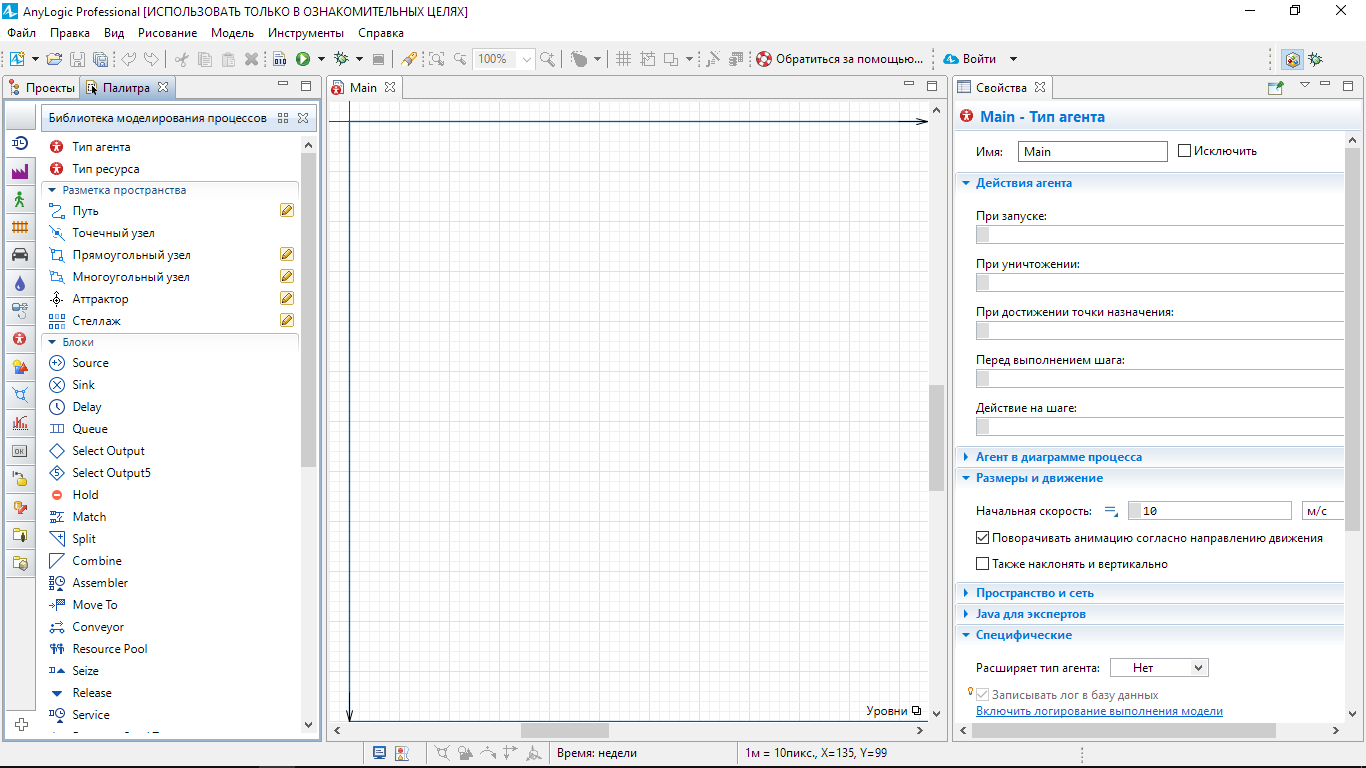


**Создание модели**

Для выполнения данной лабораторной работы требуется создать модель в инструменте имитационного моделирования AnyLogic. Назовем нашу модель Dead\_Game, т. к., хотя за основу взята BeerGame, мы против алкоголя, ведь алкоголь убивает. Назначим единицы модельного времени – недели.



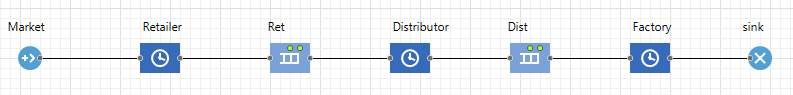
После создания модели откроется панель разработчика или рабочая область:



В левой части области находится панель Палитра, обеспечивающая легкую навигацию по элементам моделей.

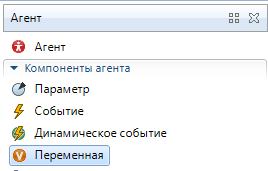
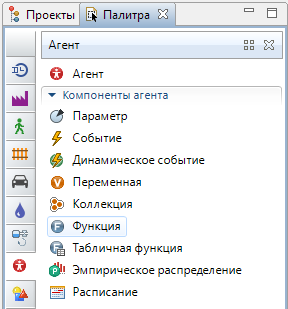
**Создание диаграммы процессов**

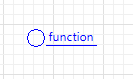
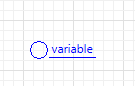
Наша модель представляет собой цепочку следующего вида:



|  |  |
| --- | --- |
| Наименование | Расшифровка |
| Market | Рынок |
| Retailer | Розничный торговец |
| Ret | Промежуточная очередь |
| Distributor | Дистрибьютор |
| Dist | Промежуточная очередь |
| Factory | Фабрика |
| Sink | Выход |

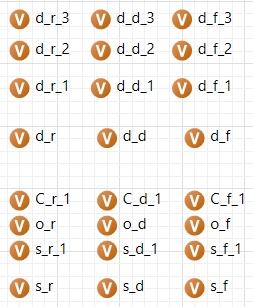
Для реализации данной модели понадобятся функции и переменные, которые находятся в палитре Агент:

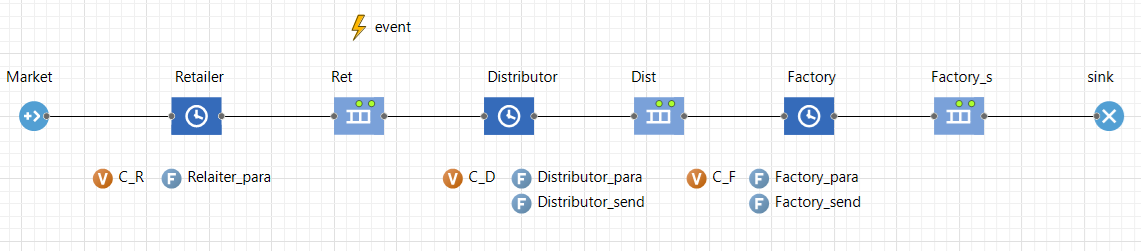
 

Согласно 2-ому алгоритму, объём заказа определяется через скользящее арифметическое среднее порядка *h.* Требуется для каждого участника цепи предусмотреть количество переменных, не меньшее *h = 3* (по варианту – это максимальное значение *h*):

Все переменные:



Преобразуем модель следующим образом:



Для каждого участника предусмотрим ряд функций, учитывая, что:

Необходимо оценить объём поставок на основе анализа истории закупок и текущего состояния запасов. В этом случае уравнение управления запасами выглядят следующим образом:

.

Смысл уравнения состоит в следующем: уровень запаса на конец периода *i* равен сумме уровня запаса на конец предшествующего периода (*i* - 1) и объема поступлений за вычетом спроса за период *i* [15].

*i* – номер периода (день, неделя, месяц);

– объём заказа производителя, дистрибьютора, оптового поставщика, розничного торговца, [шт];

– объём спроса для производителя, дистрибьютора, оптового поставщика, розничного торговца, [шт];

– объём остатков на складе у производителя, дистрибьютора, оптового поставщика, розничного торговца, [шт];

– объём поставок у производителя, дистрибьютора, оптового поставщика, розничного торговца, [шт];

 – финансовый результат (прибыль или убыток) у производителя, дистрибьютора, оптового поставщика, розничного торговца, [д.е.];

*l –* срок поставки, кол-во временных периодов;

*sdt\_o* – среднеквадратичное отклонение объёма заказов;

*sdt\_s* – среднеквадратичное отклонение объёма остатков;

*sdt\_C* – среднеквадратичное отклонение финансового результата;

 – цена производства, цена продажи у производителя, дистрибьютора, оптового поставщика, розничного торговца, [д.е.];

*t* – стоимость хранения одной единицы товара на складе, [д.е.];

*q* – наценка при перекупке, %.

Данные, определяющие значения входных показателей:

*h* – количество используемых для вычислений заказов;

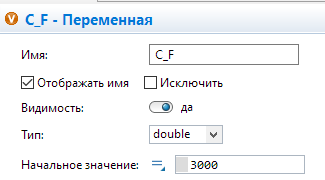
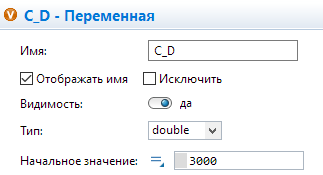
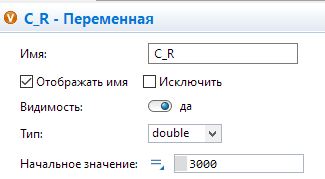
*k* – цена хранения одной единицы товара на складе, [д.е.].

Решением задачи будут оптимальные взаимодействия участников цепи поставок. Каждый из них на каждом временном промежутке выполняет следующие действия:

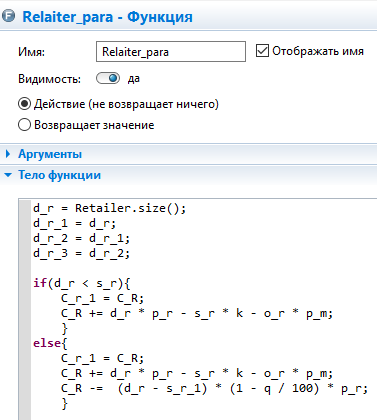
* принимает заказы от клиентов;
* получает товар от поставщиков;
* обновляет значения текущих показателей;
* отправляет товар клиентам дальше по цепи, посылает новый заказ поставщикам.

Выбор объемов заказов в каждом раунде является тем решением, которое принимают участники цепи.

Добавим в модель переменные прибыли, учитывая, что C\_R, C\_D, C\_F – прибыль каждого из участников соответственно.



**Функция для розничного торговца:**



Данная функция проверяет товар на складе и высчитывает прибыль розничного торговца, согласно формулам, указанным ниже. C\_r\_1 – прибыль на 1 итерацию в прошлое.

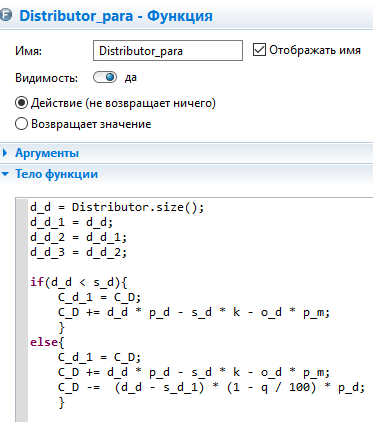
Каждую единицу времени (в рассматриваемой модели неделя) происходят поставки, продажи, закупки и производится выплата за хранение. Финансовый результат вычисляется по следующей формуле (на примере розничного торговца):

*.*

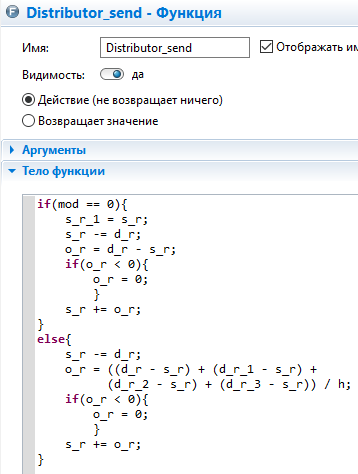
Если на складе не хватает товара, то штраф за перекупку определяется так:



**Функции для дистрибьютора:**

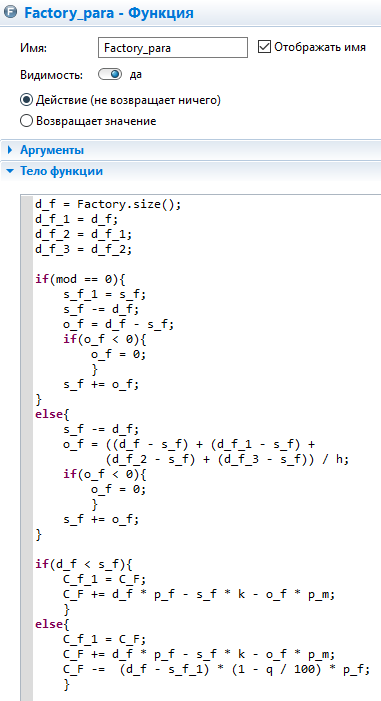


Данная функция проверяет товар на складе и высчитывает прибыль дистрибьютора, согласно формулам, аналогичным приведенным выше формулам для розничного торговца. C\_d\_1 – прибыль на 1 итерацию в прошлое. Она обрабатывается при поступлении заказа дистрибьютору.

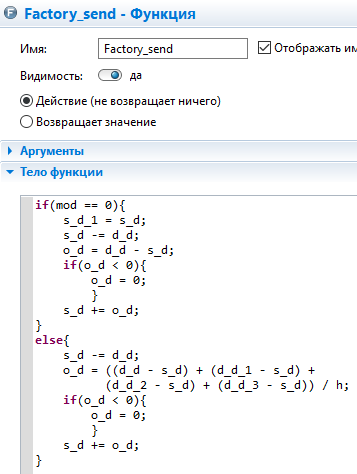


Данная функция пополняет склад ретейлера. Выполняется при выходе заказа из дистрибьютора.

**Функции для фабрики:**



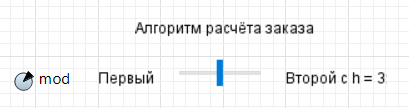
Данная функция рассчитывает прибыль фабрики, а также поступление на сырья на склад. Обрабатывается при входе заказа на фабрику.



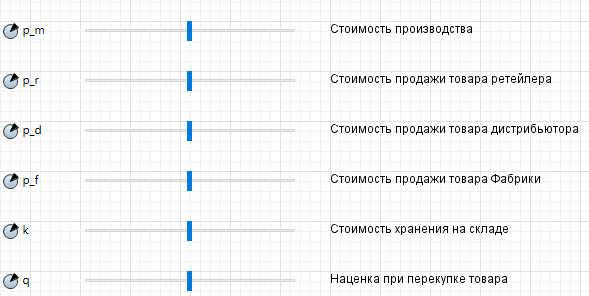
Данная функция пополняет склад дистрибьютора. Обрабатывается на выходе заказ из фабрики.

Обработка функций пополнения складов, т. е. отправки заказа заказчику выставлена на выход следующего звена в цепи поставок для того, чтобы имитировать время, необходимое на поставку заказа.

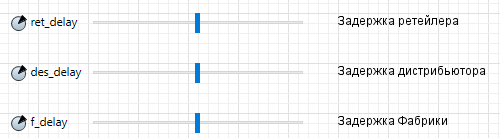
Mod – переменная режима, от которой зависит используемы алгоритм расчёта заказа (0 – первый; 1 – второй с h = 3).



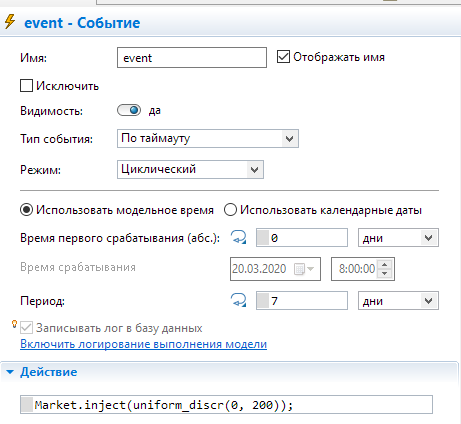
У заказа есть стоимость, которую тоже нужно регулировать. Установим соответствующие параметры для всех участников:



Также понадобятся параметры задержки:

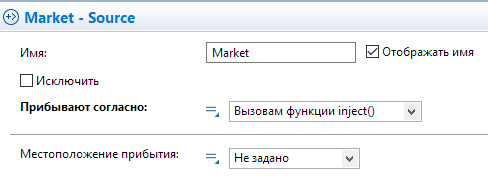


Событие event для Market (рынка) о прибытии заказа изменяется согласно стандартному дискретному распределению:

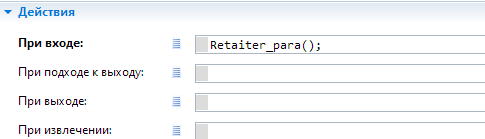


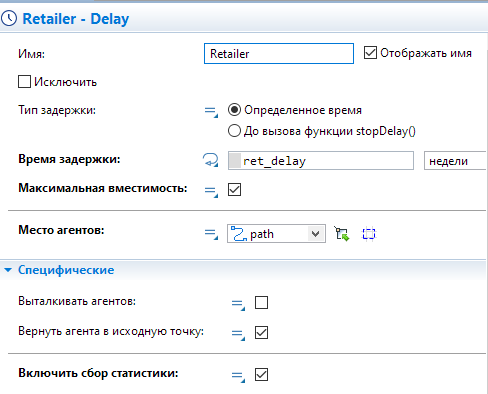
Свяжем параметры и участников.

**Рынок:**

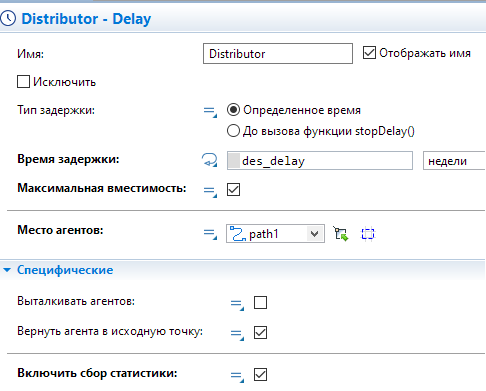


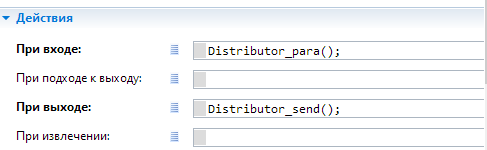
**Розничный торговец:**



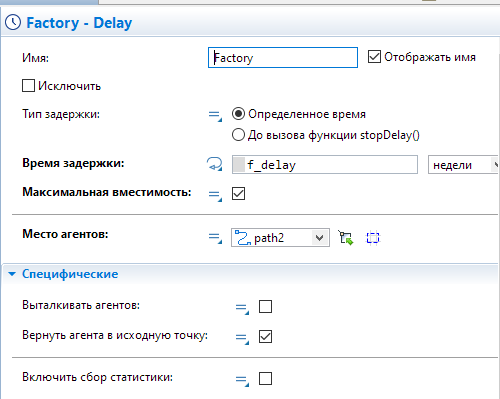


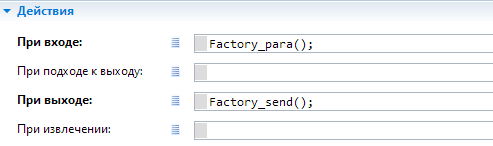
**Дистрибьютор:**





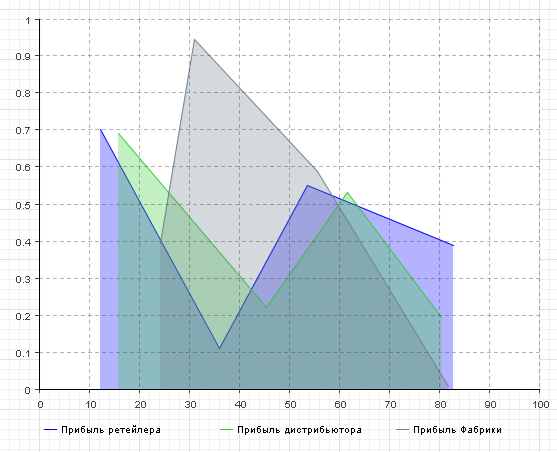
**Фабрика:**

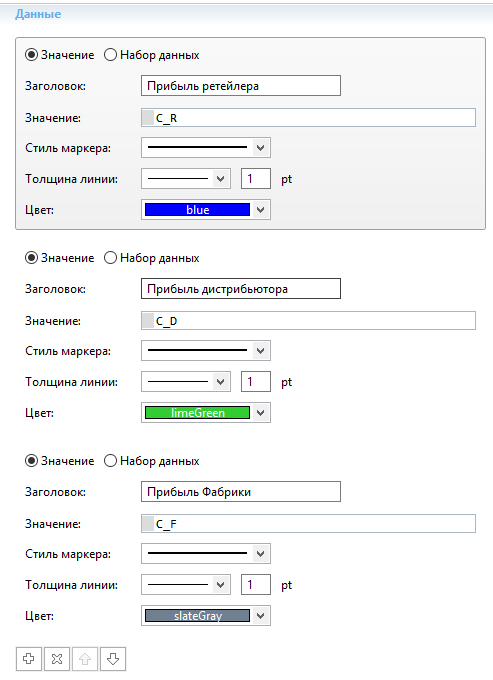




**График прибыли**

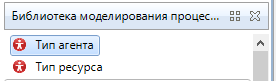
Добавим график для отображения прибыли ретейлера, дистрибьютора и фабрики:





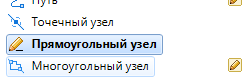
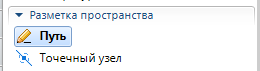
**Анимация**

Для более понятного отображения запросов выберем новый тип анимации через создание своего типа агента:

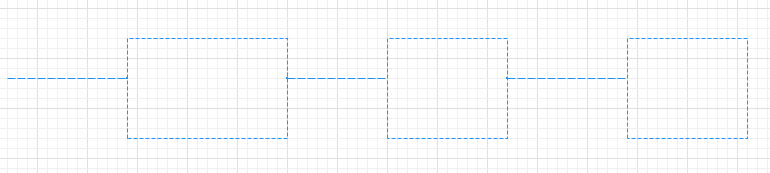


Отобразим товар в виде коробки:

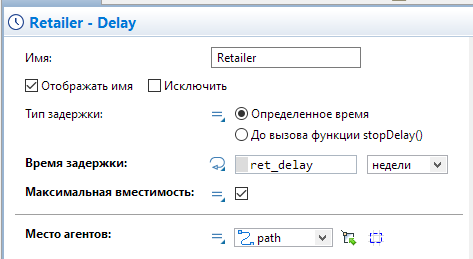
Создадим области отображения анимации:

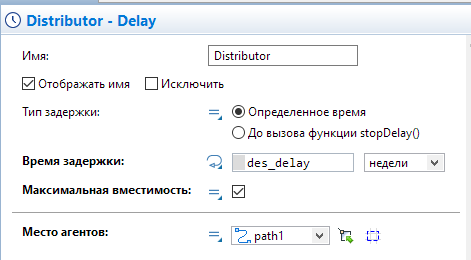
 

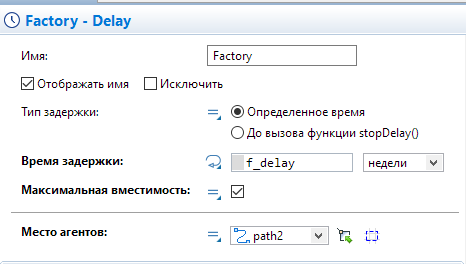
Расположим элементы следующим образом:



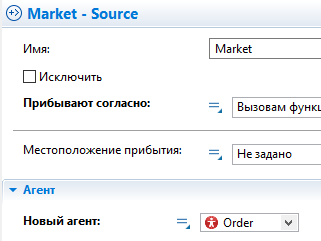
Свяжем участников и пути:

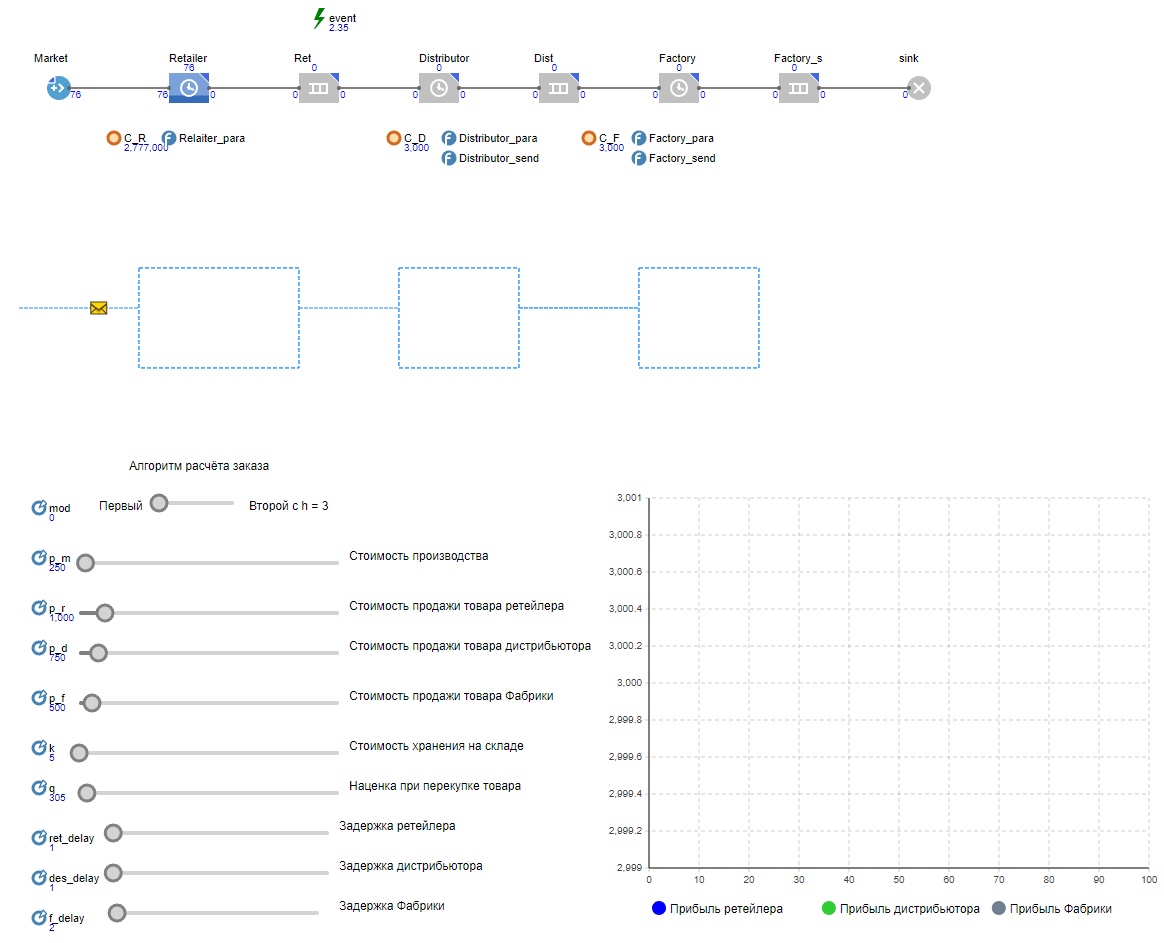






В source (рынке) укажем нового агента Order, чтобы на вход подавались запросы товара:



Проверка работоспособности: 

Проведём исследование прибыли в зависимости от различных параметров системы:

На начало исследования параметры следующие:  
Стоимость производства: 250 у.е.;

Стоимость продажи у ретейлера: 1000 у.е.;

Стоимость продажи у дистрибьютора: 750 у.е.;

Стоимость продажи у фабрики: 500 у.е.;

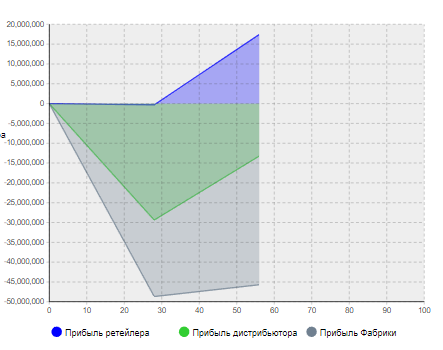
Цена при перекупке товара: 1200 у.е. (важно отметить, что этот параметр общий для всех и на начало мы используем цену, которая превышает цену продажи у ретейлера, в дальнейшем мы изменим этот параметр);

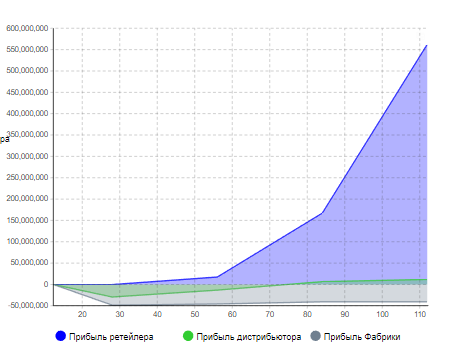
Стоимость хранения на складе: 200 у.е.;  
Задержки:

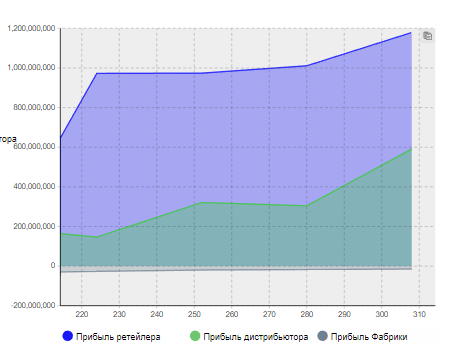
Ретейлер: 1 неделя,

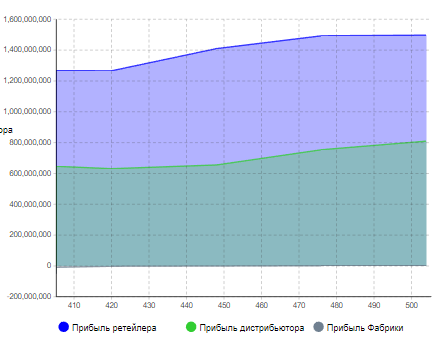
Дистрибьютор: 1 неделя,  
 Фабрика: 1 неделя.

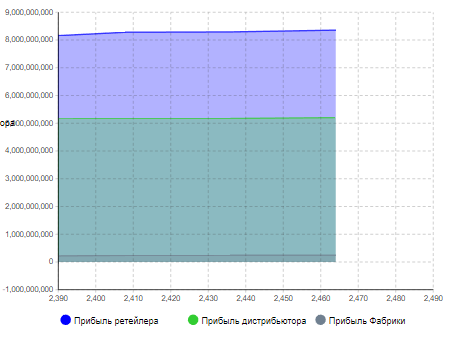
Все тесты проводятся для первого алгоритма.

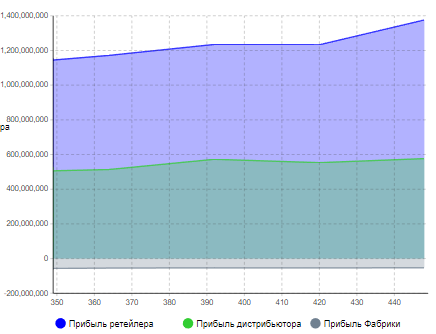
Как можно видеть по графику, сначала все участники терпят убытки: 

Однако, постепенно прибыль уходит в плюс. Наибольшую прибыль получает ретейлер, как конечное звено в цепи поставок, перед потребителем: 

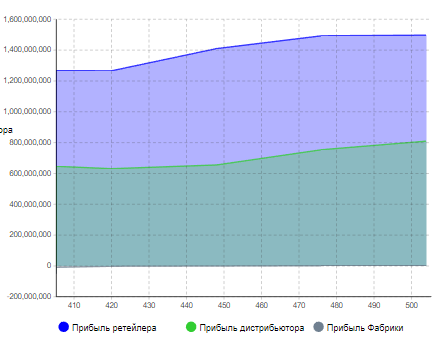
Медленнее всего растёт прибыль фабрики: 

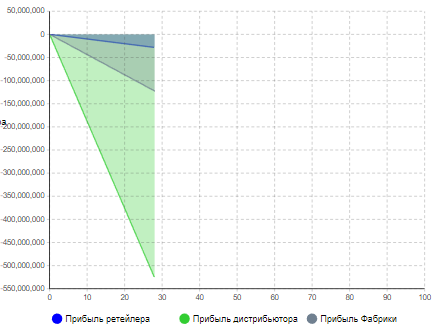
Как можно заметить, прибыль растёт скачкообразно: 

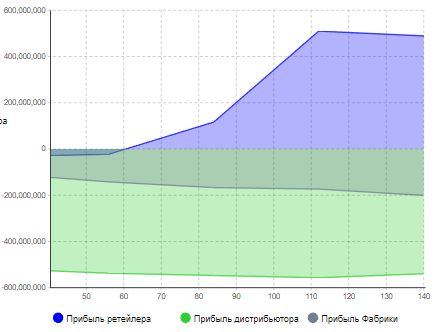
Со временем рост идёт более стабильно: 

Изменим стоимость хранения на складе, сделав её равной стоимости производства товара, т.е. 250 у.е.:  


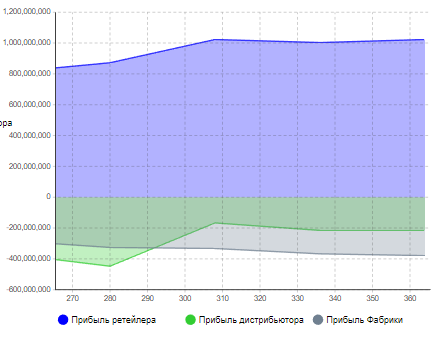
Можно заметить, что на рост прибыл подобное изменение не особенно повлияло:

График к 400-тым неделям при стоимости хранения в 200 у. е.: 

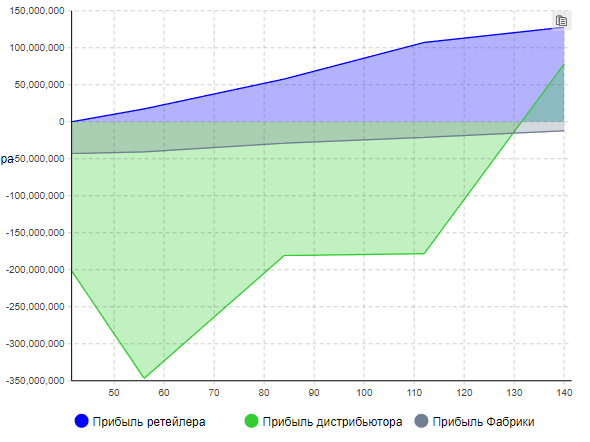
Прибыль начинает падать при приближении стоимости хранения товара к 500 у.е.: 



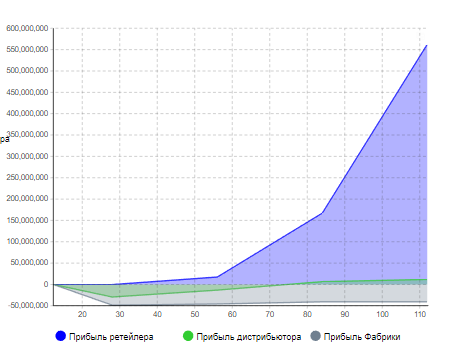
В основном от повышения стоимости хранения товара страдают фабрика и дистрибьютор, так как их цена продажи товара ниже стоимости продажи у ретейлера.



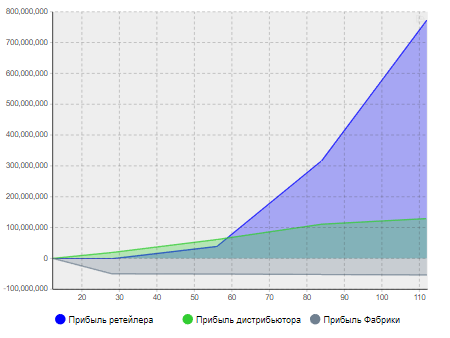
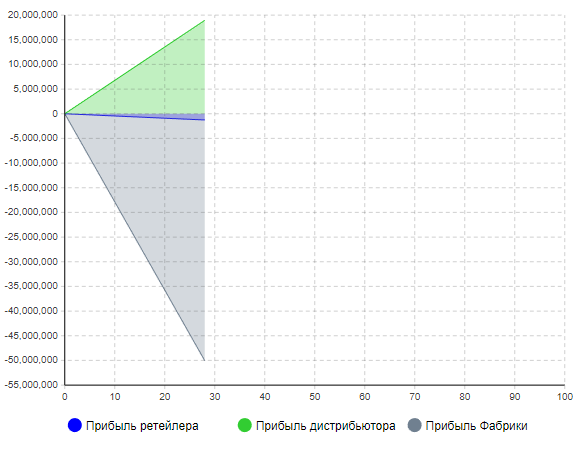
Увеличим задержку поставок у дистрибьютора до 2 недель, а у фабрики до 4:



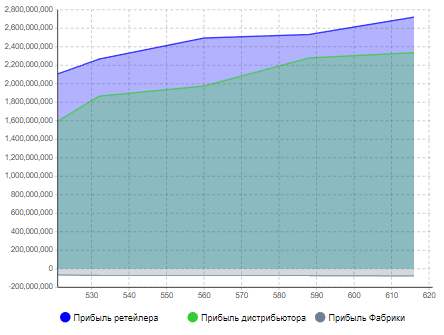
Как видно по графику, задержка в поставках ведёт к мгновенному изменению прибылей в меньшую сторону. Для сравнения ниже представлен график для задержки в одну неделю у всех членов цепи поставок:



Проверим как изменяется прибыль при использовании второго алгоритма:



Как можно заметить, прибыль при втором алгоритме растёт заметно быстрее.

Следовательно можно сделать вывод, что второй алгоритм предпочтительнее. Однако, можно, заметить, что из-за того, что наценка при перекупке на 700 у. е. больше стоимости продажи товара фабрикой, второй алгоритм приносит ей только убытки: 

Изходя из того, как меняется прибыль при изменении размера штрафа при перекупке, можно сделать вывод, что данный параметр, не должен привышать стоимости продажи более чем в два раза.

Кроме того, если стоимость производства составляет половину от стоимости продажи, окупить производство быстро не получится. На всех графиках видно, что фабрика дольше всех находиться в в минусе по прибыли. Для второго алгоритма окупаемости у фабрики нет вообще.

Изменим стоимость производства с 250 у. е. до 150 у. е.:

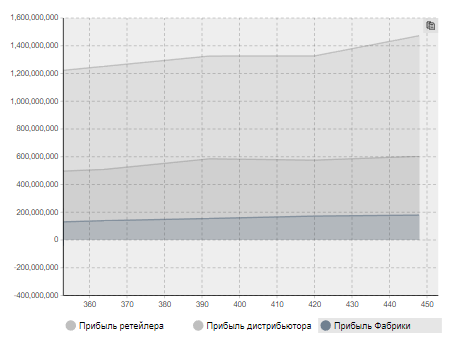
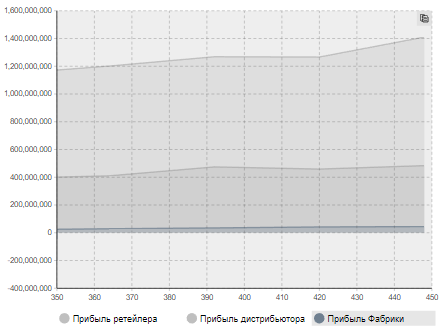
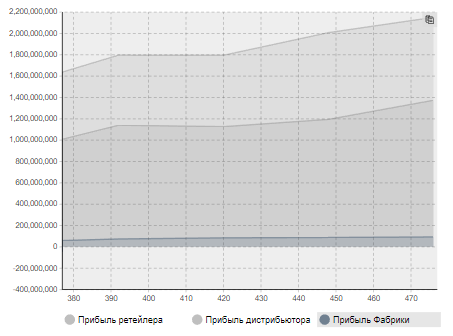
Как можно заметить, фабрика начала получать прибыль в гораздо больших объёмах:  


График для сравнения (стоимость производства составляет 250 у. е.):

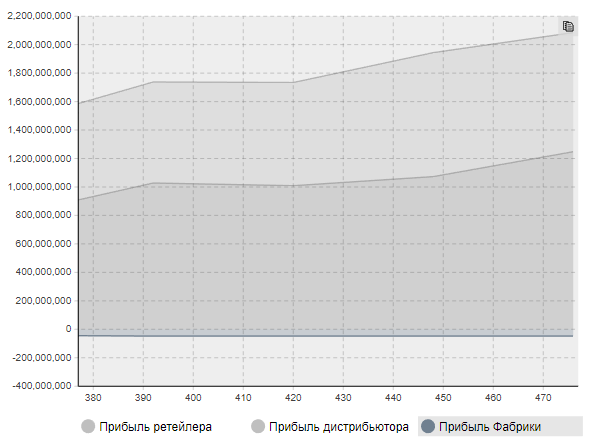


Схожие графики для второго алгоритма:

Стоимость производства 150 у. е.:



Стоимость производства 250 у. е.:



1. **Выводы**

В ходе проведённой работы нами была построена модель цепи поставок на основе BeerGame. Проведя исследование работы модели в зависимости от различных параметров системы, можно сделать вывод, что прибыль точнее и реалистичнее рассчитывается по второму алгоритму, так как он учитывает несколько прошлых состояний системы, а не только предыдущее.