ПОСТРОЕНИЕ МОДЕЛИ ПРОИЗВОДСТВЕННОГО ПРОЦЕССА СРЕДСТВАМИ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ И ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

В.А. Котляров, А.А. Дебелый, В.А. Лощенко, Н.Р. Спиричева (Екатеринбург)

Имитационное моделирование – это процесс построения и изучения математических моделей. Имитационное моделирование незаменимо там, где необходимо выяснить, как поведет себя тот или иной моделируемый процесс в зависимости от входных данных. В современном мире имитационное моделирование имеет широкий спектр применения. Оно используется в медицине, экономике, при проектировании жилых зданий и сложных систем.[1] Так, в частности, при необходимости рассчитать прибыльность производства можно прибегнуть к методам моделирования. В этом случае необходимо выделить факторы и численные величины, влияющие на производство, а также знать производительность и возможные дефекты задействованных в производстве приборов и оборудования.[2] Так, для процесса моделирования необходимо выделение различных моделей, представляющих различные объекты из материального мира и имеющие определенные количественные признаки (например, производительность станка Н деталей в минуту или затраты Н единиц ресурса на производство готового изделия). В ходе моделирования должны быть представлены последовательные этапы производства, на каждом из которых происходит определенный процесс с изменением количественных величин, которые содержит в себе модель.

На сегодняшний день моделирование – один из наиболее удобных способов расчета экономики производственного процесса. Благодаря ему можно анализировать и прогнозировать издержки и доходы производства, принимать решения о вводе в эксплуатацию того или иного оборудования.[3]

При разработке модели производства необходимо детально учитывать все его стадии, при необходимости учитывая возможные погрешности.

Описание задачи

Для примера построения имитационной модели производства была взята следующая задача.

Контейнеры с керамическими изделиями поступают в цех обжига каждые 8 минут. Каждый контейнер содержит партию из 100 изделий, которые требуют одинакового времени обжига. Время обжига – равномерно распределенная величина в интервале 20 +- 8.В цехе находится печь. в которую одновременно загружают три контейнера. Время обжига соответствует наибольшему из времен, необходимых для обжига изделий из этих трех контейнеров. Прибыль от обжига каждого изделия составляет 5 единиц стоимости. Один час работы печи требует 20 единиц стоимости (учитывается только «чистое» время работы печи).

Необходимо построить модель и сравнить экономическую эффективность следующих дисциплин обслуживания:

A. Контейнеры загружаются в печь по три по принципу FIFO. Для поддержки функционирования очереди необходимо 11 единиц стоимости в час.

B. Контейнеры разделяются на две очереди: очередь с большим временем обжига и.очередь с меньшим временем обжига изделий в печи. В печь загружаются во три контейнера из каждой очереди выбор осуществляется по принципу FIFO. Для поддержки этих двух очередей необходимо k1*11 единиц стоимости.

C. Контейнеры разделяются на три очереди: с «большим», средним» и «меньшим»временем обжига изделий в печи. В печь загружаются по три контейнера из каждой очереди, выбор осуществляется по принципу FIFO. Для поддержки функционирования этих трех очередей необходимо k2*11 единиц стоимости.

Моделирование необходимо проводить в течении 24 часов. Оценить интервалы значений k1 и k2 при которых дисциплины B и C становятся невыгодными.

Систему можно разделить на цех и печь:
1 Цех – управляет генерацией заказов, работой печи, считает стоимость очереди (в каждый такт времени).
2 Печь – обжаривает контейнеры.
Цех
Цех занимается управлением генерации заявок, работой печи и подсчетом стоимости поддержания очередей за каждый такт времени. Такт времени – 1 минута. Генерация заявок происходит каждые 8 минут.
Печь
Печь умеет обжаривать контейнеры. Печь имеет следующие состояния:
1) Ожидание загрузки контейнеров (печь свободна)
2) Обработка контейнеров (печь загружена, работает)
3) Готовность заказа (печь отработала, ждет разгрузки)
На рис. 1 можно посмотреть все состояния печи:

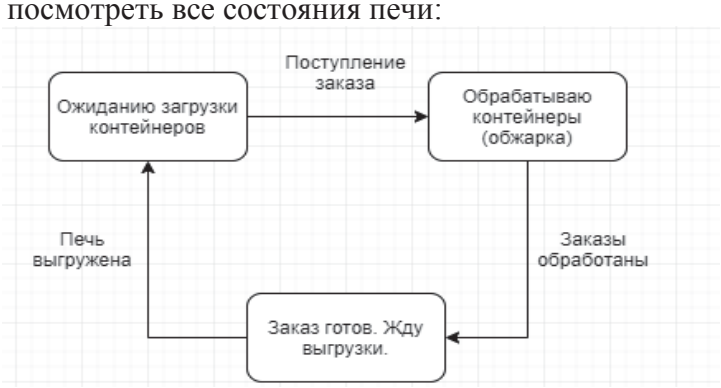

Рис. 1. Переходы состояний печи

Построение модели с использованием языка C#

C# – объектно-ориентированный язык программирования, т.е. позволяющий создавать классы объектов и их экземпляры.[4] Данная особенность языка существенно упрощает моделирование процессов т.к. под каждый элемент системы заводится свой класс – модель с определенным набором характеристик.

Ниже (рис. 2) представлен метод, который содержит основную логику работы цеха. Метод возвращает результат моделирования – класс MoneyResult, который содержит поля с прибылью и расходом.

```
public MoneyResult Start()
{
    while (Time < modelingTime)
    {
        GenerateOrders();

        ProcessOven();

        Time++;

        ProcessQueueCost();
    }

    return moneyResult;
}
```

Рис. 2. Метод Start. Основная логика работы Цеха.

Метод генерации заявок представлен на рис. 3. Генерация заявок происходит каждые 8 минут. Создается класс Order, который содержит свойство Time, которому присваивается

случайное значение в интервале от 12 до 28. Далее созданная заявка передается в обработчик очередей (об обработчике очередей будет рассказано позже). Если обработчик может предоставить 3 заявки, то данный метод поместит их в очередь на обжарку.

```
private void GenerateOrders()
{
    if (Time >= NextGenerateOrderTime)
    {
        var order = Order.CreateOrder(rnd);
        ordQueue.AddOrder(order);

        foreach (var ord in ordQueue.GetOrders())
        {
            ordersQueue.Enqueue(ord);
        }

        NextGenerateOrderTime += Constants.CreateOrderTime;
    }
}
```

Рис. 3. Метод генерации заявок

Обработка печи состоит из четырех процедур:

- Если печь работает (обжаривает), то происходит вызов метода Process() у печи, который увеличивает время с момента начала обжарки.
- Процедура подсчета стоимости работы печи. Печь предоставляет количество отработанных часов, которое умножается на стоимость работы печи за час.
- Процедура загрузки печи. Если печь свободна, то в нее при доступности загружаются новые контейнеры.

Процедура обработки результата работы печи. Когда печь закончила обрабатывать очередную партию контейнеров, то с помощью этого метода можно получить количество обработанных контейнеров для подсчета прибыли, и также можно “выгрузить печь” – вернуть ее в состояние “жду новых заказов”.

Метод обработки печи представлен на рис. 4.

```
private void ProcessOven()
{
    if (oven.Status == OvenStatus.Cooking)
    {
        oven.Process();
    }

    ProcessOvenCost();

    LoadOvenWithOrders();

    ProcessOvenResult();
}
```

Рис. 4. Метод обработки печи

Метод обработки стоимости очереди (рис. 5) каждые 60 минут увеличивает статью расходов на значение k * 11.

```
private void ProcessQueueCost()
{
    if (Time != 0 && Time % 60 == 0)
    {
        moneyResult.Charge += coefficient * Constants.QueueCostPerHour;
    }
}
```

Рис. 5. Метод обработки стоимости очереди

Обработчик очереди управляет очередью из заявок. Обработчик очереди умеет создавать n очередей, помещать в них заявки в зависимости от времени заявки. Каждой из n очередей соответствует свой промежуток времени, который получается путем деления промежутка времени обработки контейнера на n.

Пример промежутков времени для очередей:

1) Задача A (1 очередь) – [12;28]
2) Задача B (2 очереди) – [12;20);[20;28]
3) Задача C (3 очереди) – [12;17);[17;22);[22;28]

Цех передает обработчику сгенерированную заявку. Обработчик кладет ее в соответствующую очередь. Как только в одной из очередей наберется 3 заявки, при запросе Цеха, обработчик очереди отдает Цеху эти 3 заявки (рис. 6).

```
public IEnumerable<Order> GetOrders()
{
    // Получаем очередь, имеющую Count = 3 элементов из всех очередей
    var queue = Orders.Values.FirstOrDefault(q => q.Count >= Count);

    if (queue == null)
    {
        yield break;
    }

    // Возвращаем 3 элемента из очереди Цеху
    for (var i = 0; i < Count; i++)
    {
        yield return queue.Dequeue();
    }
}
```

Рис. 6. Метод поставки заявок в Цех. (Метод вызывается из Цеха)

В результате работы, программа провела следующие эксперименты:

- Работа цеха с одной очередью заявок (задача A)
- Работа цеха с двумя очередями заявок (задача B)
- Работа цеха с тремя очередями заявок (задача C)

Во втором и третьем эксперименте происходил подбор коэффициентов k1 и k2, при которых соответствующая задача становится невыгодной.

Коэффициент ki увеличивает стоимость обслуживания очереди в соответствующей задаче (эксперименте).

В ходе данной реализации программы моделирования была выявлена зависимость, что *Задача C* – цех с тремя очередями заявок, становится невыгодной при большем коэффициенте. Для обоих коэффициентов (k1, k2) указаны значения на одной оси, потому что значения коэффициентов увеличиваются на единицу со значения 1 до значения, пока задача станет невыгодной .(рис.7).

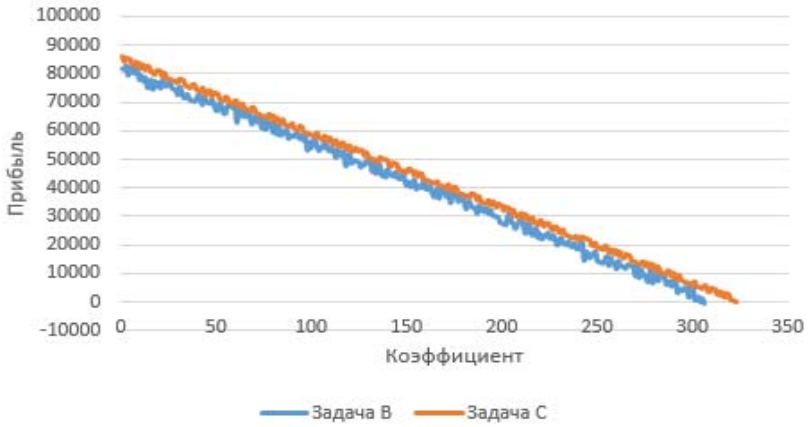

Рис.7. Зависимость прибыли от коэффициента

Таким образом, благодаря особенностям языка C# получилось построить достаточно гибкую модель, параметры которой при необходимости можно быстро менять, а также добавлять дополнительные обработчики.

Среда моделирования AnyLogic

AnyLogic – программное обеспечение для имитационного моделирования с графическим интерфейсом и поддержкой языка программирования Java для разработки моделей. AnyLogic содержит большое количество различных библиотек моделирования процессов, возможностей их объединения и представления.[5]

В среде AnyLogic имитационная модель производства была создана с использованием библиотеки производственных систем и библиотеки агентного моделирования и состоит из нескольких элементов:

- Непрерывный конвейер, по которому перемещаются контейнеры с максимально возможной скоростью (для исключения влияния скорости перемещения на результаты эксперимента);
- Станции обработки очереди и печи;
- Дискретно-событийная модель производства, описывающая этапы обработки заявки (создание заявки, постановка в очередь, обработка в печи);
- Элемент управления (слайдер), позволяющий задавать коэффициенты k1 и k2 для решения задач B и C;
- Переменные, содержащие различные статистические значения;
- Графические и текстовые элементы презентации статистических данных.

На рисунке 8 представлена типовая модель производства, в которую для решения каждой из трёх задач вносятся изменения в количество очередей и условия распределения по ним.

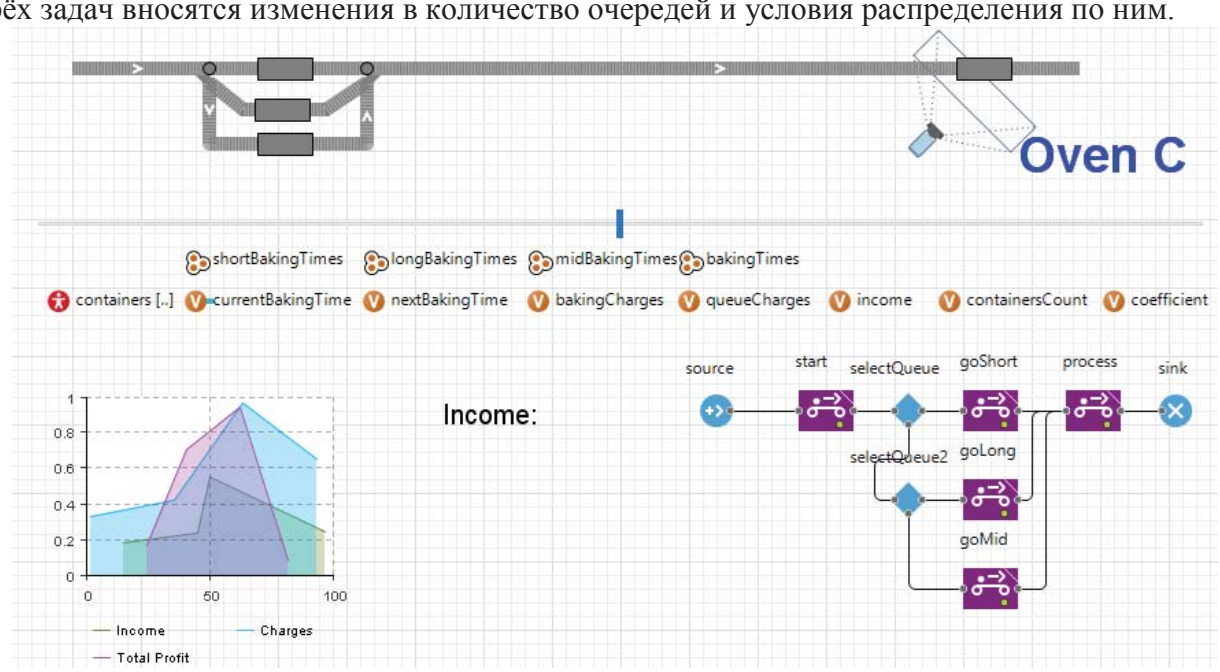

Рис. 8. Модель производства в среде AnyLogic

Таким образом, дискретно-событийная модель описывает создание заявок, перемещения контейнеров по конвейеру и условия распределения контейнеров по очередям в зависимости от времени их обжига, а основная логика их обработки написана на языке Java и выполняется при проходе контейнеров через станции обработки [6]. Станция обработки очереди группирует контейнеры в группы по 3, сохраняет максимальное время обжига среди этих трёх контейнеров в коллекцию и передаёт их в очередь на обжиг. Станция обработки печи также получает по 3 контейнера, получает из коллекции время их обжига и подсчитывает временные и материальные затраты на их обжиг. На рис. 9 и 10 представлена логика работы станций обработки очереди и печи соответственно.

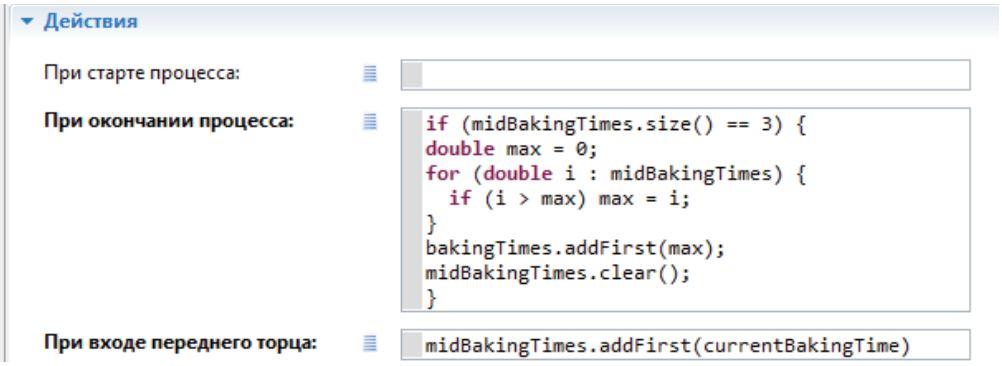

Рис. 9. Логика работы станции обработки очереди

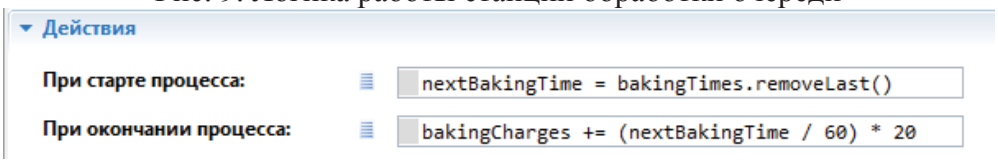

Рис. 10. Логика работы станции обработки печи

В результате проведения экспериментов были получены следующие результаты:

1) Работа цеха с одной очередью заявок (задача A)

На рис. 11 представлен результат проведения эксперимента для задачи A.

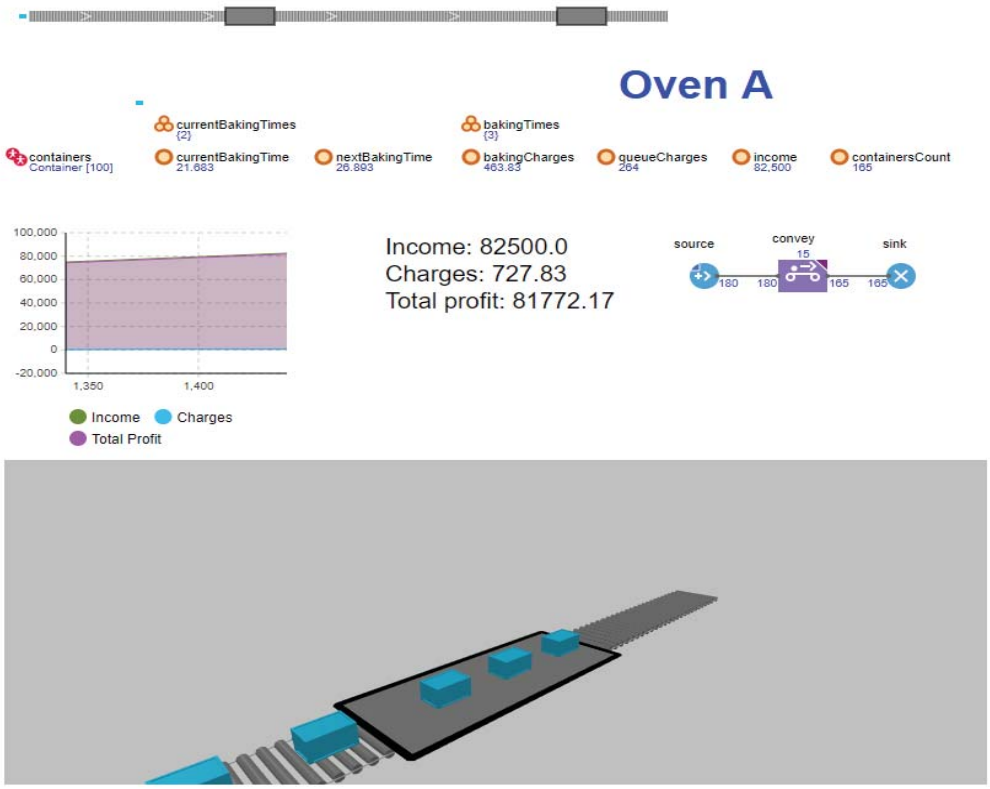

Рис. 11. Результат проведения эксперимента по условиям задачи A

Во втором и третьем эксперименте происходил подбор коэффициентов k1 и k2, при которых соответствующая задача становится невыгодной. Коэффициент ki увеличивает стоимость обслуживания очереди в соответствующей задаче (эксперименте). В ходе проведения экспериментов была выявлена зависимость, что *Задача C* – цех с тремя очередями заявок, становится невыгодной при большем коэффициенте.

2) Работа цеха с несколькими очередями заявок (задача B)

После прогона эксперимента был произведён подбор максимального коэффициента путём задания его с помощью слайдера, изменяющего соответствующую переменную модели.

Максимальный коэффициент, при котором задача остаётся выгодной с двумя очередями заявок – 305, а при коэффициенте 306 уже становится убыточной.

При введении в модель трех очередей заявок, максимальный коэффициент, при котором задача остаётся выгодной – 323, а при коэффициенте 323 уже становится убыточной.

Сравнение результатов

В ходе работы были получены результаты моделирования при помощи программы на языке C# и с использованием среды AnyLogic. Воспроизводимость последовательностей случайных чисел была обеспечена заданием одинакового зерна генератора случайных чисел в обоих моделях. При этом были получены примерно одинаковые результаты: в обоих случаях задача B становилась убыточной при коэффициенте k1 = 306, а задача C становилась убыточной при коэффициенте k2 = 323.

Заключение

Решив задачу имитационного моделирования и рассмотрев два пути решения можно сделать вывод, что использование объектно-ориентированных языков существенно ускоряет моделирование, делает модель более гибкой и расширяемой. При этом программа AnyLogic предоставляет большие возможности для визуализации, позволяет создавать модели “из коробки”, имеет большое кол-во встроенных функциональных возможностей и в целом больше подходит, если есть необходимость создать модель большого производственного процесса. Таким образом, AnyLogic лучше подходит для масштабных, типовых задач, в то время как написание собственного решения на каком-либо языке программирования позволяет сделать модель максимально гибкой и расширяемой, но может занять много времени в случае, если речь идет о достаточно большой производственной системе или необходимости визуализации.

Литература

1. Духанов, А. В., Медведева О. Н. Имитационное моделирование сложных систем: курс лекций—Владимир: Изд-во Владим. гос. ун-та, 2010.— 115 с
2. Димов Э.М., Маслов О. Н., Трошин Ю. В., Халимов Р.Р. Динамика разработки имитационной модели бизнес-процесса. // Инфокоммуникационные технологии. Т. 11, № 1, 2013.—С. 63–64
3. Макаров В.Л., Бахтизин А.Р., Сушко Е.Д. Агент-ориентированные модели как инструмент апробации управленческих решений // Управленческое консультирование. 2016. № 12. С. 16–25.
4. Троелсен Эндрю, Джепкинс Филипп, Язык программирования C# 7 и платформы .NET и .NETCore. Изд-во Вильямс, 2018г, с. 53-54
5. «AnyLogic» - программа имитационного моделирования. [Электронный ресурс] – Режим доступа: https://www.anylogic.ru (Дата обращения: 19.03.2007).
6. Medvedev S.N., Aksyonov K.A., Kruglov V.N. Comparative analysis of order allocation methods and intelligent systems for effective download of production capacities of manufacturing enterprise. Journal of Physics: Conference Series. Volume 1210, Issue 1, 4 May 2019, Number of paper 012094. 12th International Scientific and Technical Conference on Applied Mechanics and Systems Dynamics, AMSD 2018; Omsk; Russian Federation; 13-15 November 2018; Code 147986. **DOI:** 10.1088/1742-6596/1210/1/012094