|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_РТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ИУ-5\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_Моделирование\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_технического обслуживания\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_офисов банка­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студентка \_\_РТ5-71Б\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Попова Д. А.\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Студент \_\_\_\_РТ5-71Б\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Забурунов Л. В.\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Чёрненький М. В.\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2021*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине \_\_\_\_Имитационное Моделирование Дискретных Процессов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студенты группы РТ5-71Б:

Попова Дарья Алексеевна

Забурунов Леонид Вячеславович

Тема курсовой работы: Моделирование технического обслуживания офисов банка

Направленность КР (учебная, исследовательская, практическая, производственная, др.) \_учебная\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_\_\_\_\_\_\_кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения работы: 25% к 3 нед., 50% к 9 нед., 75% к 12 нед., 100% к 15 нед.

***Задание:*** Разработать имитационную модель технического обслуживания печатного офисного оборудования офисов банка.

***Оформление курсовой работы:***

Расчетно-пояснительная записка на \_\_\_ листах формата А4.

…

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дата выдачи задания** 14 сентября 2021 г.

**Руководитель курсовой работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Чёрненький М. В.**

(Подпись, дата) (И.О.Фамилия)

**Студентка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Попова Д. А.\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Забурунов Л. В.\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

**Оглавление**

[Введение 4](#_Toc90763636)

[1. Постановка задачи 5](#_Toc90763637)

[2. Структурная схема процесса 6](#_Toc90763638)

[3. Структурная схема модели 7](#_Toc90763639)

[4. Обобщённая схема моделирующего алгоритма 9](#_Toc90763640)

[5. Выбор технических средств 10](#_Toc90763641)

[6. Текст имитирующей модель программы 11](#_Toc90763642)

[7. Модель #1 15](#_Toc90763643)

[7.1 Постановка задачи 15](#_Toc90763644)

[7.2 Результаты экспериментов 16](#_Toc90763645)

[7.3 Анализ результатов моделирования 20](#_Toc90763646)

[8. Модель #2 21](#_Toc90763647)

[8.1 Постановка задачи 21](#_Toc90763648)

[8.2 Результаты экспериментов 22](#_Toc90763649)

[8.3 Анализ результатов моделирования 25](#_Toc90763650)

[9. Модель #3 26](#_Toc90763651)

[9.1 Постановка задачи 26](#_Toc90763652)

[9.2 Результаты экспериментов 27](#_Toc90763653)

[9.3 Анализ результатов моделирования 30](#_Toc90763654)

[Заключение 32](#_Toc90763655)

# Введение

[Водичка-вода…]

# Постановка задачи

Внутренняя служба центральных офисов Банка оказывает услуги по замене расходных материалов (картриджей) в печатном офисном оборудовании. Оператор-диспетчер принимает заявки с тремя уровнями приоритета, тратит 3 минутына обработку поступающего звонка или заявки в электронной системе; каждая заявка должна получить статус «принята» в течение 15 минут с момента её получения, иначе она получает статус «отклонена». Допускается накопление в очереди до 10 заявок, начиная с 11-ой каждая заявка также получит статус отклонённой. Требования к времени исполнения заявки:

1. Для высокого приоритета – не более 30 минут;
2. Для среднего приоритета – не более 2 часов;
3. Для низкого приоритета – не более 5 часов.

Также есть некоторое количество технических администраторов, производящих замену расходного материала. На обработку заявки и замену картриджа тратится время, *определяемое по некоторому вероятностному закону*.

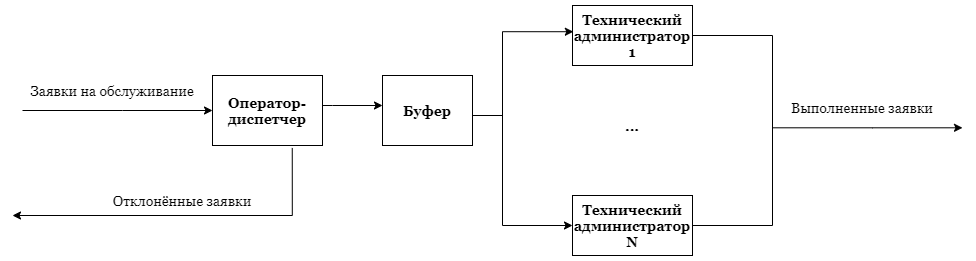
*Оператор передаёт администратору заявку в случае, если он свободен, в противном случае связывается с другим администратором.*

Также происходит еженедельная инвентаризация складов, в рамках которой производится пополнение склада до заданного количества картриджей.

Задача – смоделировать обслуживание печатного оборудования в течение месяца и подсчитать следующее:

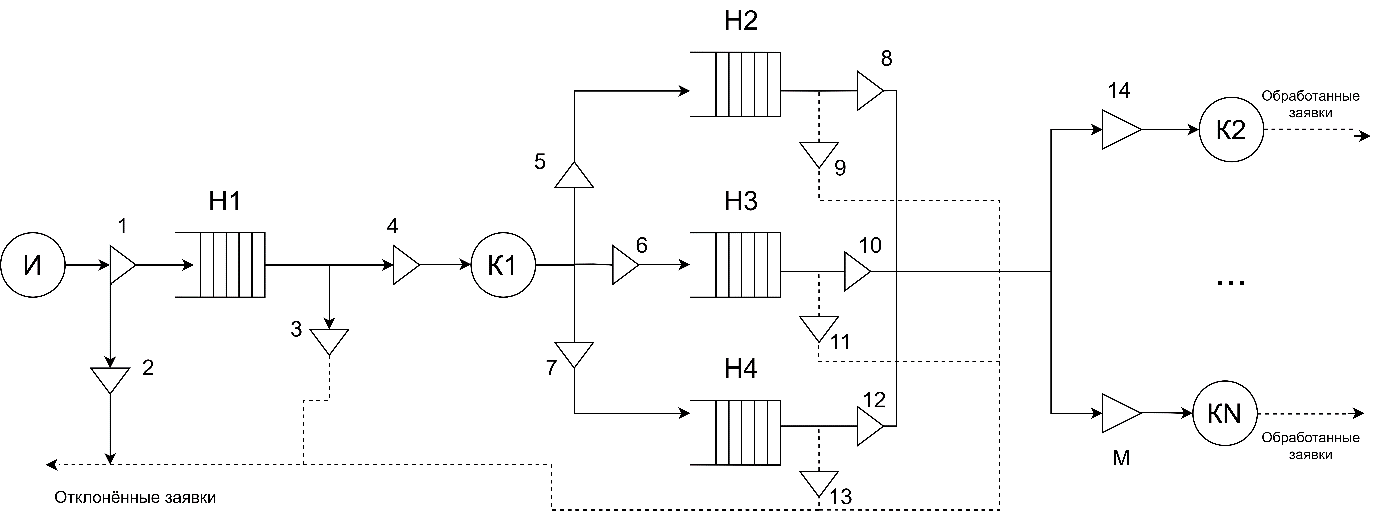
* Частоту поломок;
* Коэффициенты занятости оператора;
* Коэффициенты занятости технических администраторов.

# Структурная схема процесса



# Структурная схема модели

Представим структурную схему модели в символике *Q*-схемы (для непрерывно-стохастических моделей).



*И* – источник, из которого поступают заявки.

*Н1* – накопитель (очередь из заявок к диспетчеру-оператору).

*К1* – канал-диспетчер, обрабатывающий заявки.

*Н2*, *H3*, *H4* – очереди заявок на обслуживание к техническому администратору 1…N 1-го, 2-го и 3 уровня приоритета соответственно.

*K2*…*KN* – технический администратор под номером 2…N.

Источник *И* имитирует приходящие заявки на обслуживание. Если накопитель *Н1* заполнен (количество заявок в нём равно 10) и с источника поступает ещё одна заявка, то клапан 1 закрывается, открывается клапан 2 и заявка отклоняется. Иначе клапан 1 открывается и в накопитель в очередь к диспетчеру поступают заявки. Накопитель *Н1* заполняется до тех пор, пока количество заявок в нём не станет равным 10.

Если заявка пробыла в очереди *Н1* дольше 15 минут, то открывается ключ 3 и она также становится отклонённой.

Если заявка пробыла в очереди меньше 15 минут и оператор (диспетчер) свободен, то открывается ключ 4 и заявка попадает на обработку в К1.

После обработки в канале *К1* заявки распределяются следующим образом:

* если приоритет заявки равен 1, то открывается ключ 5 и заявка попадает в очередь с высоким приоритетом *Н2*;
* если приоритет равен 2, то открывается ключ 6 и заявка попадает в очередь со средним приоритетом *Н3*;
* если приоритет равен 3, то открывается ключ 7 и заявка попадает в очередь с низким приоритетом *Н4.*

Накопители *Н2*, *Н3*, *Н4* работают следующим образом:

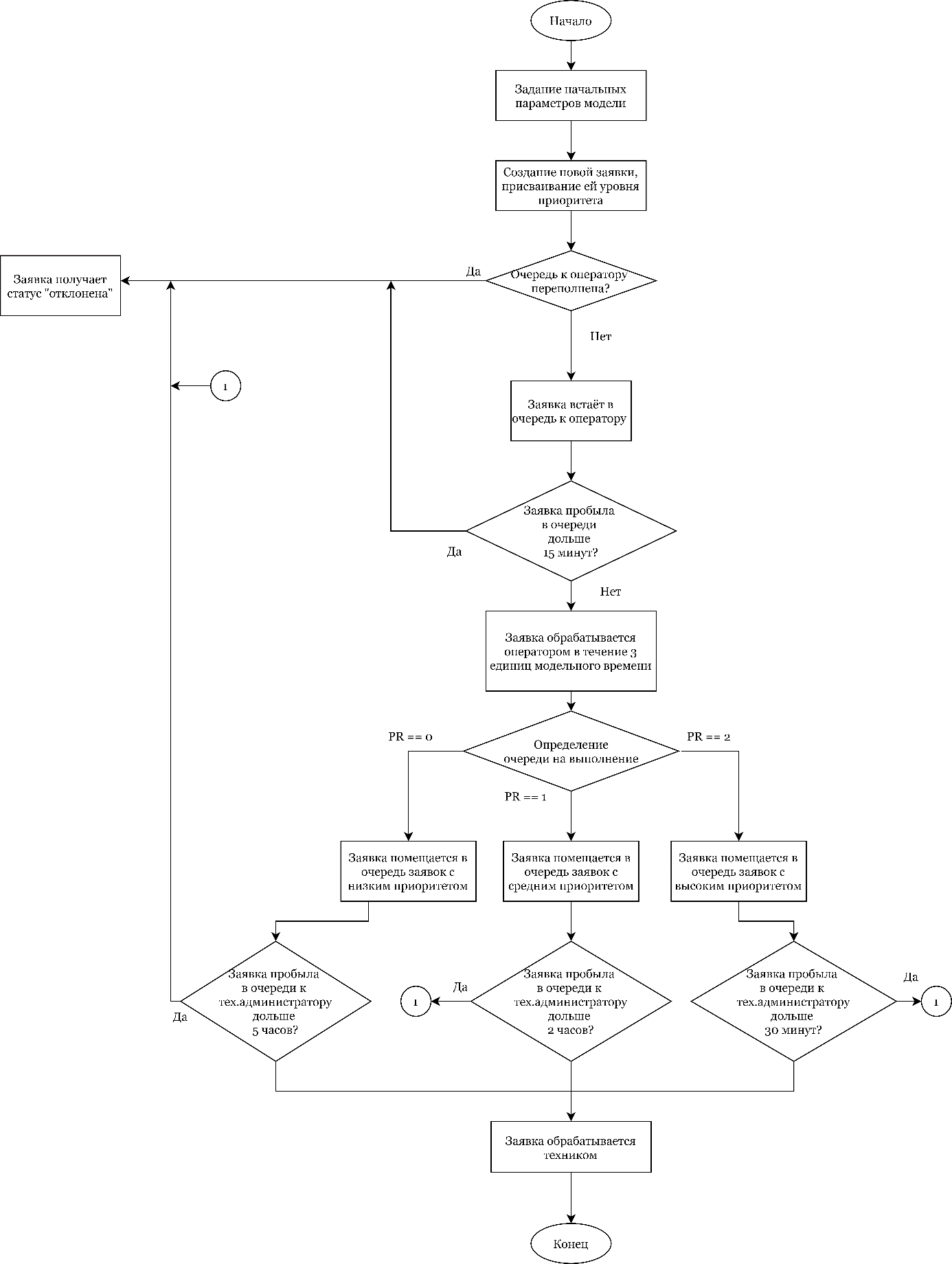
* если заявка пробыла в очереди *Н2*дольше 30 минут, то открывается ключ 9 и заявка считается отклонённой, иначе, если один из администраторов *К2*…*КN* освобождается, открывается ключ 8;
* если заявка пробыла в очереди *Н3* дольше 2 часов, то открывается ключ 11 и заявка считается отклонённой, иначе, если один из администраторов *К2*…*КN* освобождается, открывается ключ 10;
* если заявка пробыла в очереди *Н4* дольше 5 часов, то открывается ключ 13 и заявка считается отклонённой иначе, если один из администраторов *К2*…*КN* освобождается, открывается ключ 12.

Если администраторы (каналы) *К2*…*КN* свободны, то ключи 14…*M* открываются и заявки отправляются с вероятностью на обработку в каналы *К2*…*КN*.

Если канал *К2* занят, то ключ 14 закрывается и открывается ключ 15, если канал *К3* занят, то открывается ключ 16 и так далее. Если все каналы *К2*…*КN* заняты, то заявка находится в накопителе *Н2*, *Н3* или *Н4* до тех пор, пока один из каналов не освободится и ключ не откроется.

# Обобщённая схема моделирующего алгоритма

Единица модельного времени – 1 минута.



# Выбор технических средств

Имитационная модель реализована на языке *GPSS* в среде *GPSS* *World* и симулирует техническое обслуживание офисов банка.

Функция *PRIOR* вычисляет приоритет заявки с помощью датчика случайных чисел *RN1*. Наименьший приоритет будет присвоен с вероятностью 0,5, средний – с вероятностью 0,3, самый высокий – 0,2:

*PRIOR FUNCTION RN1,D3*

*0.5,0/0.8,1/1,2*

Многоканальные устройства:

* *Operator* (диспетчер)
* *Techs* (технические администраторы)
* *OperatorQueue* (очередь заявок к оператору)
* *HighPriorInquiries*, *MedPriorInquiries*, *LowPriorInquiries* – очереди из заявок с высоким, средним и низким приоритетом соответственно.

Дополнительные переменные, используемые в программе (в основном для сбора статистики)

* *QueueWaitingTime* (время ожидания в очереди)

# Текст имитирующей модель программы

Также стоит отметить, что некоторые числовые параметры будут изменяться в процессе исследования построенной модели (например, вместимость многоканального устройства).

*PRIOR FUNCTION RN1,D3*

*0.5,0/0.8,1/1,2*

*\* функция присваивает заявке наименьший приоритет (0) с вер-тью 0,5*

*\* 1 - с вер-тью 0,3, и самый высокий приоритет (2) - с вер-тью 0,2*

*Operator STORAGE 1*

*Techs STORAGE 10*

*OperatorQueue STORAGE 10*

*HighPriorInquiries STORAGE 100*

*MedPriorInquiries STORAGE 100*

*LowPriorInquiries STORAGE 100*

*\*--------------УНИЧТОЖАЕМ ЗАЯВКУ-------------*

*Terminate\_Transaction TERMINATE 1*

*\*----------------СОЗДАЁМ ЗАЯВКИ----------------*

*GENERATE 1,,1,,FN$PRIOR; Заявки генерируются раз в 1 минуту с определяемым ф-ей приоритетом (0, 1 или 2)*

*\*----ПРОВЕРЯЕМ ОЧЕРЕДЬ К ОПЕРАТОРУ НА ПЕРЕПОЛНЕНИЕ----*

*Check\_OpQueueOccupancy GATE SNF OperatorQueue,Terminate\_Transaction; Если очередь на обработку оператора уже заполнена, то мы сразу выходим*

*\*-------ПРОВЕРЯЕМ СОСТОЯНИЯ ОЧЕРЕДИ---------*

*Enter\_OpQueue ENTER OperatorQueue*

*QUEUE OperatorQueue\_Stats*

*MARK QueueWaitingTime; Запоминаем время в момент занятия очереди, чтобы потом вычислить длительность ожидания*

*GATE SNF Operator,Enter\_OperatorWaitLine; Если оператор занят, то встаём в очередь*

*\*------- ОПЕРАТОР ОБРАБАТЫВАЕТ ЗАЯВКУ-------*

*Leave\_OpQueue DEPART OperatorQueue\_Stats*

*LEAVE OperatorQueue*

*ASSIGN QueueWaitingTime,MP$QueueWaitingTime ; Итоговое время ожидания = "Текущее модельное время" - "Запомненное модельное время"*

*Check\_WaitTimeExpired TEST LE P$QueueWaitingTime,15,Terminate\_Transaction ; Если заявка пробыла в очереди дольше 15 минут, то выкидываем её*

*ENTER Operator*

*ADVANCE 3*

*LEAVE Operator*

*UNLINK OperatorWaitLine,Leave\_OpQueue,1; Вызываем к оператору следующую заявку*

*TRANSFER ,Test\_HighPriorityQueue*

*\*------------ОЖИДАЕМ ОПЕРАТОРА---------------*

*Enter\_OperatorWaitLine LINK OperatorWaitLine,FIFO*

*\*---ОПРЕДЕЛЯЕМ ОЧЕРЕДЬ НА ВЫПОЛНЕНИЕ---*

*\* Если в названии переменной или метки присутствует 0, 1 или 2, то этот объект относится к транзакту с соответствующим приоритетом!*

*Test\_HighPriorityQueue TEST E PR,2,Test\_MedPriorityQueue; Если приоритет равен двум, то встаём в очередь с высоким приоритетом*

*GATE SNF HighPriorInquiries,Terminate\_Transaction; Если очередь заявок с PR=2 заполнена, то сразу выходим*

*TRANSFER ,Enter\_HighPriorityQueue ; Переходим в очередь заявок с PR=2*

*Test\_MedPriorityQueue TEST E PR,1,Test\_LowPriorityQueue; Если приоритет равен единице, то встаём в очередь со средним приоритетом*

*GATE SNF MedPriorInquiries,Terminate\_Transaction; Если очередь заявок с PR=1 заполнена, то сразу выходим*

*TRANSFER ,Enter\_MediumPriorityQueue ; Переходим в очередь заявок с PR=1*

*Test\_LowPriorityQueue TEST E PR,0,Terminate\_Transaction; Если приоритет равен нулю, то встаём в очередь с низким приоритетом (все остальные случаи не рассматриваются)*

*GATE SNF LowPriorInquiries,Terminate\_Transaction; Если очередь заявок с PR=0, заполнена, то сразу выходим*

*TRANSFER ,Enter\_LowPriorityQueue ; Переходим в очередь заявок с PR=0*

*\*----ВЫПОЛНЯЕМ ЗАЯВКУ С ПРИОРИТЕТОМ 2----*

*Enter\_HighPriorityQueue ENTER HighPriorInquiries*

*QUEUE HighPrInquiries\_Stats*

*MARK TimeSpentInQueue; Отмечаем, в какое время мы зашли в очередь*

*GATE SNF Techs,WaitForTech; Если нет свободных техников, то заносим транзакт в список ожидания*

*\*------------ТЕХНИК ВЫПОЛНЯЕТ ЗАЯВКУ------------*

*Execute\_HighPriorityInquiry DEPART HighPrInquiries\_Stats*

*LEAVE HighPriorInquiries*

*ASSIGN TimeSpentInQueue,MP$TimeSpentInQueue; Считаем, сколько времени транзакт провёл в очереди*

*Check\_HighPriorityTimeExpired TEST LE P$TimeSpentInQueue,30,TerminateTransaction; Если заявка c PR=2 пробыла в очереди дольше 30 минут, то выкидываем её*

*ENTER Techs*

*ADVANCE 20,2*

*LEAVE Techs*

*UNLINK TechWaitLine,ExecuteAnotherRequest,1; Посылаем следующий транзакт на определение ветви исполнения*

*TRANSFER ,Terminate\_Transaction; Заявка выполнена, уничтожаем транзакт*

*\*----ВЫПОЛНЕНИЕ ЗАЯВКИ С ПРИОРИТЕТОМ 1----*

*Enter\_MediumPriorityQueue ENTER MedPriorInquiries*

*QUEUE MedPrInquiries\_Stats*

*MARK TimeSpentInQueue; Отмечаем, в какое время мы зашли в очередь*

*GATE SNF Techs,WaitForTech; Если нет свободных техников, то заносим транзакт в список ожидания*

*\*------------ТЕХНИК ВЫПОЛНЯЕТ ЗАЯВКУ------------*

*Execute\_MediumPriorityInquiry DEPART MedPrInquiries\_Stats*

*LEAVE MedPriorInquiries*

*ASSIGN TimeSpentInQueue,MP$TimeSpentInQueue; Считаем, сколько времени транзакт провёл в очереди*

*Check\_MediumPriorityTimeExpired TEST LE P$TimeSpentInQueue,120,TerminateTransaction; Если заявка c PR=1 пробыла в очереди дольше 2 часов, то выкидываем её*

*ENTER Techs*

*ADVANCE 15,3*

*LEAVE Techs*

*UNLINK TechWaitLine,ExecuteAnotherRequest,1; Посылаем следующий транзакт на определение ветви исполнения*

*TRANSFER ,Terminate\_Transaction; Заявка выполнена, уничтожаем транзакт*

*\*----ВЫПОЛНЕНИЕ ЗАЯВКИ С ПРИОРИТЕТОМ 0----*

*Enter\_LowPriorityQueue ENTER LowPriorInquiries*

*QUEUE LowPrInquiries\_Stats*

*MARK TimeSpentInQueue; Отмечаем, в какое время мы зашли в очередь*

*GATE SNF Techs,WaitForTech; Если нет свободных техников, то встаём в очередь*

*\*------------ТЕХНИК ВЫПОЛНЯЕТ ЗАЯВКУ------------*

*Execute\_LowPriorityInquiry DEPART LowPrInquiries\_Stats*

*LEAVE LowPriorInquiries*

*ASSIGN TimeSpentInQueue,MP$TimeSpentInQueue;Считаем, сколько времени транзакт провёл в очереди*

*Check\_LowPriorityTimeExpired TEST LE P$TimeSpentInQueue,300,TerminateTransaction; Если заявка c PR=0 пробыла в очереди дольше 5 часов, то выкидываем её (потрачено)*

*ENTER Techs*

*ADVANCE 10,3*

*LEAVE Techs*

*UNLINK TechWaitLine,ExecuteAnotherRequest,1; Посылаем следующий транзакт на определение ветви исполнения*

*TRANSFER ,Terminate\_Transaction; Заявка выполнена, уничтожаем транзакт*

*\*---------ОЖИДАЕМ ТЫЖПРОГРАММИСТА---------*

*WaitForTech LINK TechWaitLine,FIFO*

*\*-------ОПРЕДЕЛЯЕМ ВЕТВЬ ИСПОЛНЕНИЯ--------*

*ExecuteAnotherRequest TEST E PR,2,TestMedExecute ; Если приоритет равен двум, то исполняемая заявка идёт по пути для заявок с высоким приоритетом*

*TRANSFER ,Execute\_HighPriorityInquiry*

*TestMedExecute TEST E PR,1,TestLowExecute; Если приоритет равен единице, то исполняемая заявка идёт по пути для заявок со средним приоритетом*

*TRANSFER ,Execute\_MediumPriorityInquiry*

*TestLowExecute TEST E PR,0,TerminateTransaction; Если приоритет равен нулю, то исполняемая заявка идёт по пути для заявок с низким приоритетом (остальные случаи не рассматриваются)*

*TRANSFER ,Execute\_LowPriorityInquiry*

# Модель #1

# Постановка задачи

Пусть в первом варианте модели у нас будет один оператор и очередь *OperatorQueue* на 10 заявок к нему, 10 технических администраторов, и три очереди вместимостью по 100 заявок каждая для заявок с разными уровнями приоритета.

Заявка считается отклонённой после 15 минут нахождения в очереди к оператору, это проверяется через параметр *P$QueueWaitingTime*.

Пусть также оператор регистрирует одну заявку в течение одной единицы модельного времени (минуты), заявки с низким приоритетом выполняются во временном диапазоне от 7 до 13 минут, со средним – от 12 до 18 минут, с высоким – от 18 до 22 минут.

*Operator STORAGE 1*

*Techs STORAGE 10*

*OperatorQueue STORAGE 10*

*HighPriorInquiries STORAGE 100*

*MedPriorInquiries STORAGE 100*

*LowPriorInquiries STORAGE 100*

# Результаты экспериментов

Моделирование проводилось в течение 480 единиц модельного времени (единица времени прибавляется при каждом уничтожении заявки независимо от причины).

**Многоканальные устройства**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя** | **Емкость памяти** | **Число свободных единиц памяти к концу моделирования** | **Кол-во входов в память** | **Коэффициент использования памяти** |
| *OPERATOR* | 1 | 0 | 132 | 0,8 |
| *TECHS* | 10 | 7 | 131 | 0,33 |
| *OPERATORQUEUE* | 10 | 0 | 266 | 0,937 |
| *HIGHPRIORINQUIRIES* | 100 | 100 | 18 | 0 |
| *MEDPRIORINQUIRIES* | 100 | 100 | 35 | 0 |
| *LOWPRIORINQUIRIES* | 100 | 100 | 78 | 0 |

**Очереди**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Имя** |  |  |  |  |  |  |  |
| *OPERATORQUEUE\_STATS* | 141 | 141 | 266 | 125 | 75,5 | 140,2 | 264,5 |
| *MEDPRINQUIRIES\_STATS* | 1 | 0 | 35 | 35 | 0 | 0 | 0 |
| *LOWPRINQUIRIES\_STATS* | 1 | 0 | 78 | 78 | 0 | 0 | 0 |
| *HIGHPRINQUIRIES\_STATS* | 1 | 0 | 18 | 18 | 0 | 0 | 0 |

Где – максимальное содержимое очереди за период моделирования

– текущее содержимое очереди,

– общее количество входов транзактов в очередь,

– общее количество входов транзактов в очередь с нулевым временем ожидания,

– среднее значение содержимого очереди,

– среднее время пребывания одного транзакта в очереди,

– время пребывания одного транзакта в очереди без учета 'нулевых'.

**Блоки**

|  |  |  |  |
| --- | --- | --- | --- |
| **Метка** | **Тип блока** | **Количество вошедших в блок транзактов** | **Кол-во тран. в блоке в конце моделирования** |
| *TERMINATE\_TRANSACTION* | *TERMINATE* | 480 | 0 |
| *GENERATE* | 494 | 0 |
| *CHECK\_OPQUEUE*  *OCCUPANCY* | *GATE* | 494 | 0 |
| *ENTER\_OPQUEUE* | *ENTER* | 266 | 0 |
| *QUEUE* | 266 | 0 |
| *MARK* | 266 | 0 |
| *GATE* | 266 | 0 |
| *DEPART* | 256 | 0 |
| *LEAVE\_OPQUEUE* | *LEAVE* | 256 | 0 |
| *ASSIGN* | 256 | 0 |
| *CHECK\_WAITTIMEEXPIRED* | *TEST* | 256 | 0 |
| *ENTER* | 132 | 0 |
| *ADVANCE* | 132 | 1 |
| *LEAVE* | 131 | 0 |
| *UNLINK* | 131 | 0 |
| *TRANSFER* | 131 | 0 |
| *ENTER\_OPERATORWAITLINE* | *LINK* | 141 | 10 |
| *TEST\_HIGHPRIORITYQUEUE* | *TEST* | 131 | 0 |
| *GATE* | 18 | 0 |
| *TRANSFER* | 18 | 0 |
| *TEST\_MEDPRIORITYQUEUE* | *TEST* | 113 | 0 |
| *GATE* | 35 | 0 |
| *TRANSFER* | 35 | 0 |
| *TEST\_LOWPRIORITYQUEUE* | *TEST* | 78 | 0 |
| *GATE* | 78 | 0 |
| *TRANSFER* | 78 | 0 |
| *ENTER\_HIGHPRIORITYQUEUE* | *ENTER* | 18 | 0 |
| *QUEUE* | 18 | 0 |
| *MARK* | 18 | 0 |
| *GATE* | 18 | 0 |
| *DEPART* | 18 | 0 |
| *EXECUTE\_HIGHPRIORITY*  *INQUIRY* | *LEAVE* | 18 | 0 |
| *ASSIGN* | 18 | 0 |
| *CHECK\_HIGHPRIORITY*  *TIMEEXPIRED* | *TEST* | 18 | 0 |
| *ENTER* | 18 | 0 |
| *ADVANCE* | 18 | 0 |
| *LEAVE* | 18 | 0 |
| *UNLINK* | 18 | 0 |
| *TRANSFER* | 18 | 0 |
| *ENTER\_MEDIUMPRIORITYQUEUE* | *ENTER* | 35 | 0 |
| *QUEUE* | 35 | 0 |
| *MARK* | 35 | 0 |
| *GATE* | 35 | 0 |
| *DEPART* | 35 | 0 |
| *EXECUTE\_MEDIUMPRIORITY*  *INQUIRY* | *LEAVE* | 35 | 0 |
| *ASSIGN* | 35 | 0 |
| *CHECK\_MEDIUMPRIORITY*  *TIMEEXPIRED* | *TEST* | 35 | 0 |
| *ENTER* | 35 | 0 |
| *ADVANCE* | 35 | 0 |
| *LEAVE* | 35 | 0 |
| *UNLINK* | 35 | 0 |
| *TRANSFER* | 35 | 0 |
| *ENTER\_LOWPRIORITYQUEUE* | *ENTER* | 78 | 0 |
| *QUEUE* | 78 | 0 |
| *MARK* | 78 | 0 |
| *GATE* | 78 | 0 |
| *DEPART* | 78 | 0 |
| *EXECUTE\_LOWPRIORITY*  *INQUIRY* | *LEAVE* | 78 | 0 |
| *ASSIGN* | 78 | 0 |
| *CHECK\_LOWPRIORITY*  *TIMEEXPIRED* | *TEST* | 78 | 0 |
| *ENTER* | 78 | 0 |
| *ADVANCE* | 78 | 3 |
| *LEAVE* | 75 | 0 |
| *UNLINK* | 75 | 0 |
| *TRANSFER* | 75 | 0 |
| *WAITFORTECH* | *LINK* | 0 | 0 |
| *EXECUTEANOTHERREQUEST* | *TEST* | 0 | 0 |
| *TRANSFER* | 0 | 0 |
| *TESTMEDEXECUTE* | *TEST* | 0 | 0 |
| *TRANSFER* | 0 | 0 |
| *TESTLOWEXECUTE* | *TEST* | 0 | 0 |
| *TRANSFER* | 0 | 0 |

# Анализ результатов моделирования

Было сгенерировано 494 заявки, модель прекратила работу после того, как в блок *TERMINATE* попали 480 из них. В очередь к оператору (метка *ENTER*\_*OPQUEUE* и число в очереди *OPERATORQUEUE*\_*STATS*) попали только 266 из 494 заявок.

Таким образом, только 54% заявок «добрались» до очереди к оператору, а 46% было выкинуто из системы, поскольку очередь к оператору была полностью занята.

При этом после проверки *CHECK*\_*WAITTIMEEXPIRED* на обработку к оператору попали 132 заявки из 256, то есть 124 покинули очередь из-за ожидания дольше 15 минут. было потеряно, так что в следующей реализации модели мы можем увеличить максимальное время ожидания в очереди. В общей сложности было потеряно 3 заявки, то есть . Было обработано 78 заявок с низким приоритетом, 35 – со средним, 18 – с высоким.

Коэффициент использования многоканального устройства *OPERATOR* равен 0,8, что можно считать довольно хорошим результатом – не будем ничего предпринимать. Коэффициент использования многоканального устройства *OPERATORQUEUE* равен 0,937, что тоже является прекрасным результатом, но, как было отмечено выше, огромное число заявок не может встать в очередь на обслуживание к оператору, поскольку она занята, поэтому в следующей конфигурации модели стоит увеличить ёмкость памяти *OPERATORQUEUE*.

Коэффициент использования многоканального устройства *TECHS* равен 0,33, что говорит о том, что наши технические администраторы, мягко говоря, простаивают. Есть два варианта: или уменьшить их число (сократив объём памяти *TECHS*), или увеличить время, которое технический администратор тратит на обработку одной заявки.

# Модель #2

# Постановка задачи

Для второго варианта модели применим выводы, полученные из анализа результатов работы первой модели. Теперь ёмкость многоканального устройства *Techs* будет равняться 5, а не 10 (сократим штат технических администраторов вдвое). Увеличим размер очереди заявок на обработку *OperatorQueue* с 10 до 200.

Также теперь пусть заявка будет считаться отклонённой после 90 минут в очереди к оператору, а не 15.

Очередь для заявок с низким приоритетом будет состоять из 10 ячеек памяти, со средним – из 8, с высоким – из 6.

*Operator STORAGE 1*

*Techs STORAGE 5*

*OperatorQueue STORAGE 200*

*HighPriorInquiries STORAGE 4*

*MedPriorInquiries STORAGE 6*

*LowPriorInquiries STORAGE 10*

# Результаты экспериментов

Как и в прошлый раз, цикл моделирования – 480 единиц (уничтоженных заявок).

**Многоканальные устройства**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя** | **Емкость памяти** | **Число свободных единиц памяти к концу моделирования** | **Кол-во входов в память** | **Коэффициент использования памяти** |
| *OPERATOR* | 1 | 0 | 143 | 0,8 |
| *TECHS* | 5 | 0 | 72 | 0,95 |
| *OPERATORQUEUE* | 30 | 0 | 300 | 0,94 |
| *HIGHPRIORINQUIRIES* | 4 | 0 | 16 | 0,58 |
| *MEDPRIORINQUIRIES* | 6 | 3 | 37 | 0,84 |
| *LOWPRIORINQUIRIES* | 10 | 1 | 55 | 0,84 |

**Очереди**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Имя** |  |  |  |  |  |  |  |
| *OPERATORQUEUE\_STATS* | 200 | 200 | 568 | 163 | 149,8 | 185,4 | 260 |
| *MEDPRINQUIRIES\_STATS* | 33 | 33 | 49 | 16 | 18,8 | 269,9 | 400,7 |
| *LOWPRINQUIRIES\_STATS* | 54 | 54 | 65 | 11 | 29,9 | 323,9 | 389,8 |
| *HIGHPRINQUIRIES\_STATS* | 17 | 17 | 27 | 10 | 9,1 | 236,8 | 376,1 |

Где – максимальное содержимое очереди за период моделирования

– текущее содержимое очереди,

– общее количество входов транзактов в очередь,

– общее количество входов транзактов в очередь с нулевым временем ожидания,

– среднее значение содержимого очереди,

– среднее время пребывания одного транзакта в очереди,

– время пребывания одного транзакта в очереди без учета 'нулевых'.

**Блоки**

|  |  |  |  |
| --- | --- | --- | --- |
| **Метка** | **Тип блока** | **Количество вошедших в блок транзактов** | **Количество транзактов в блоке в конце моделирования** |
| *TERMINATE\_TRANSACTION* | *TERMINATE* | 480 | 0 |
| *GENERATE* | 703 | 0 |
| *CHECK\_OPQUEUEOCCUPANCY* | *GATE* | 703 | 0 |
| *ENTER\_OPQUEUE* | *ENTER* | 568 | 0 |
| *QUEUE* | 568 | 0 |
| *MARK* | 568 | 0 |
| *GATE* | 568 | 0 |
| *LEAVE\_OPQUEUE* | *DEPART* | 368 | 0 |
| *LEAVE* | 368 | 0 |
| *ASSIGN* | 368 | 0 |
| *CHECK\_WAITTIMEEXPIRED* | *TEST* | 368 | 0 |
| *ENTER* | 206 | 0 |
| *ADVANCE* | 206 | 1 |
| *LEAVE* | 205 | 0 |
| *UNLINK* | 205 | 0 |
| *TRANSFER* | 205 | 0 |
| *ENTER\_OPERATORWAITLINE* | *LINK* | 405 | 200 |
| *TEST\_HIGHPRIORITYQUEUE* | *TEST* | 205 | 0 |
| *GATE* | 36 | 0 |
| *TRANSFER* | 27 | 0 |
| *TEST\_MEDPRIORITYQUEUE* | *TEST* | 169 | 0 |
| *GATE* | 74 | 0 |
| *TRANSFER* | 49 | 0 |
| *TEST\_LOWPRIORITYQUEUE* | *TEST* | 95 | 0 |
| *GATE* | 95 | 0 |
| *TRANSFER* | 65 | 0 |
| *ENTER\_HIGHPRIORITYQUEUE* | *ENTER* | 27 | 0 |
| *QUEUE* | 27 | 0 |
| *MARK* | 27 | 0 |
| *GATE* | 27 | 0 |
| *DEPART* | 10 | 0 |
| *EXECUTE\_HIGHPRIORITY*  *INQUIRY* | *LEAVE* | 24 | 0 |
| *ASSIGN* | 24 | 0 |
| *CHECK\_HIGHPRIORITY*  *TIMEEXPIRED* | *TEST* | 24 | 0 |
| *ENTER* | 10 | 0 |
| *ADVANCE* | 10 | 2 |
| *LEAVE* | 8 | 0 |
| *UNLINK* | 8 | 0 |
| *TRANSFER* | 8 | 0 |
| *ENTER\_MEDIUM*  *PRIORITYQUEUE* | *ENTER* | 49 | 0 |
| *QUEUE* | 49 | 0 |
| *MARK* | 49 | 0 |
| *GATE* | 49 | 0 |
| *DEPART* | 16 | 0 |
| *EXECUTE\_MEDIUM*  *PRIORITYINQUIRY* | *LEAVE* | 44 | 0 |
| *ASSIGN* | 44 | 0 |
| *CHECK\_MEDIUMPRIORITY*  *TIMEEXPIRED* | *TEST* | 44 | 0 |
| *ENTER* | 25 | 0 |
| *ADVANCE* | 25 | 0 |
| *LEAVE* | 25 | 0 |
| *UNLINK* | 25 | 0 |
| *TRANSFER* | 25 | 0 |
| *ENTER\_LOW*  *PRIORITYQUEUE* | *ENTER* | 65 | 0 |
| *QUEUE* | 65 | 0 |
| *MARK* | 65 | 0 |
| *GATE* | 65 | 0 |
| *DEPART* | 11 | 0 |
| *EXECUTE\_LOW*  *PRIORITYINQUIRY* | *LEAVE* | 55 | 0 |
| *ASSIGN* | 55 | 0 |
| *CHECK\_LOWPRIORITY*  *TIMEEXPIRED* | *TEST* | 55 | 0 |
| *ENTER* | 55 | 0 |
| *ADVANCE* | 55 | 2 |
| *LEAVE* | 53 | 0 |
| *UNLINK* | 53 | 0 |
| *TRANSFER* | 53 | 0 |
| *WAITFORTECH* | *LINK* | 104 | 18 |
| *EXECUTEANOTHER*  *REQUEST* | *TEST* | 86 | 0 |
| *TRANSFER* | 14 | 0 |
| *TESTMEDEXECUTE* | *TEST* | 72 | 0 |
| *TRANSFER* | 28 | 0 |
| *TESTLOWEXECUTE* | *TEST* | 44 | 0 |
| *TRANSFER* | 44 | 0 |

# Анализ результатов моделирования

Было сгенерировано 703 заявки, модель прекратила работу после того, как в блок *TERMINATE* попали 480 из них. В очередь к оператору (метка *ENTER\_OPQUEUE* и число в очереди *OPERATORQUEUE*\_*STATS*) попали 568 из 703 заявок. Таким образом, процент заявок, проваленных на этапе обработки:

Дальше 405 заявок из этих 568 попали в очередь через блок *LINK*, а 163 «проскочили» без очереди. Даже при условии, что максимальное время ожидание в очереди (после которого заявка считается отклонённой) увеличилось с 15 до 90 минут, всё равно прошло через оператора только 205 заявок из 703 (29%). Ещё 200 заявок остались в очереди на обслуживание к оператору на момент завершения моделирования. Было обработано 53 заявки с низким приоритетом, 25 – со средним, 8 – с высоким.

Что касается коэффициентов использования, то главное изменение здесь – это коэффициент использования тех. администраторов: теперь он равен не 0,3, а 0,955. Это говорит о том, что каждый сотрудник работает практически на пределе возможностей; мы также можем увидеть, что снижение числа тех. администраторов до пяти при данной постановке задачи оказалось губительным, поскольку по итогу число выполненных заявок оказалось даже меньше, чем в первом эксперименте. Также заметим, что использование буфера для очереди оператора снизилось несильно даже при таком радикальном изменении: увеличив размерность в 20 раз, мы получили снижение только на 20%.

# Модель #3

# Постановка задачи

В рамках третьего, завершающего эксперимента мы слегка поменяем схему анализа: попробуем построить график процента проваленных заявок в зависимости от значения числа операторов и техников. Для этого мы внесём изменения в нашу модель:

1. Будем проводить моделирование для конкретного отрезка времени, а не для числа заявок. С этой целью мы убираем единицу из блока *TERMINATE* и добавляем второй блок генерации:

*\*------CЧЁТЧИК МОДЕЛЬНОГО ВРЕМЕНИ-------*

*ModelTime GENERATE 480,,480,,3*

*TERMINATE 1*

1. Сам блок *TERMINATE* теперь будет продублирован в двух местах: мы обозначим в программе метки для проваленной и успешно выполненной заявки:

*\*----УНИЧТОЖАЕМ ПРОВАЛЕННУЮ ЗАЯВКУ----*

*Terminate\_Failured\_Transaction TERMINATE 0*

*\*----УНИЧТОЖАЕМ ВЫПОЛНЕННУЮ ЗАЯВКУ----*

*Terminate\_Succeded\_Transaction TERMINATE 0*

1. Зафиксируем остальные числовые параметры модели:

* Размерность буферов очереди оператора и технического администратора сделаем 500, чтобы не ограничивать её искусственно;
* Заявки будут поступать по экспоненциальному закону с мат. ожиданием 30 секунд и дисперсией 15 секунд.

# Результаты экспериментов

В этот раз вместо табличек из стандартного отчёта *GPSS* приведём сводную статистику, собранную с помощью *MS Excel*. Эксперимент был проведён для одного, двух, трёх и четырёх операторов.

**Один оператор**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Число технических администраторов** | **Число созданных заявок** | **Число успешно завершённых заявок** | **Число проваленных заявок** | **Число находящихся в модели заявок** | **Процент выполненных заявок** | **Процент невыполненных заявок** |
| 1 | 632 | 32 | 153 | 447 | 5% | 95% |
| 2 | 632 | 70 | 160 | 402 | 11% | 89% |
| 3 | 632 | 111 | 154 | 367 | 18% | 82% |
| 4 | 632 | 136 | 136 | 360 | 22% | 78% |
| 5 | 632 | 138 | 136 | 358 | 22% | 78% |
| 6 | 632 | 139 | 136 | 357 | 22% | 78% |
| 7 | 632 | 139 | 136 | 357 | 22% | 78% |
| 8 | 632 | 139 | 136 | 357 | 22% | 78% |
| 10 | 632 | 139 | 136 | 357 | 22% | 78% |
| 12 | 632 | 139 | 136 | 357 | 22% | 78% |
| 30 | 632 | 139 | 136 | 357 | 22% | 78% |

**Два оператора**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Число технических администраторов** | **Число созданных заявок** | **Число успешно завершённых заявок** | **Число проваленных заявок** | **Число находящихся в модели заявок** | **Процент выполненных заявок** | **Процент невыполненных заявок** |
| 1 | 632 | 36 | 283 | 313 | 6% | 94% |
| 2 | 632 | 77 | 292 | 263 | 12% | 88% |
| 3 | 632 | 111 | 305 | 216 | 18% | 82% |
| 4 | 632 | 144 | 312 | 176 | 23% | 77% |
| 5 | 632 | 184 | 296 | 152 | 29% | 71% |
| 6 | 632 | 220 | 298 | 114 | 35% | 65% |
| 7 | 632 | 247 | 273 | 112 | 39% | 61% |
| 8 | 632 | 270 | 262 | 100 | 43% | 57% |
| 9 | 632 | 272 | 262 | 98 | 43% | 57% |
| 10 | 632 | 272 | 262 | 98 | 43% | 57% |
| 11 | 632 | 271 | 262 | 99 | 43% | 57% |
| 12 | 632 | 271 | 262 | 99 | 43% | 57% |
| 16 | 632 | 271 | 262 | 99 | 43% | 57% |
| 20 | 632 | 271 | 262 | 99 | 43% | 57% |
| 30 | 632 | 271 | 262 | 99 | 43% | 57% |

**Три оператора**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Число технических администраторов** | **Число созданных заявок** | **Число успешно завершённых заявок** | **Число проваленных заявок** | **Число находящихся в модели заявок** | **Процент выполненных заявок** | **Процент невыполненных заявок** |
| 1 | 632 | 37 | 190 | 405 | 6% | 94% |
| 2 | 632 | 74 | 203 | 355 | 12% | 88% |
| 4 | 632 | 152 | 226 | 254 | 24% | 76% |
| 6 | 632 | 225 | 239 | 168 | 36% | 64% |
| 8 | 632 | 298 | 223 | 111 | 47% | 53% |
| 10 | 632 | 367 | 208 | 57 | 58% | 42% |
| 12 | 632 | 416 | 167 | 49 | 66% | 34% |
| 14 | 632 | 433 | 167 | 32 | 69% | 31% |
| 16 | 632 | 432 | 167 | 33 | 68% | 32% |
| 18 | 632 | 433 | 167 | 32 | 69% | 31% |
| 20 | 632 | 433 | 167 | 32 | 69% | 31% |
| 24 | 632 | 433 | 167 | 32 | 69% | 31% |
| 28 | 632 | 433 | 167 | 32 | 69% | 31% |
| 30 | 632 | 433 | 167 | 32 | 69% | 31% |

**Четыре оператора**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Число технических администраторов** | **Число созданных заявок** | **Число успешно завершённых заявок** | **Число проваленных заявок** | **Число находящихся в модели заявок** | **Процент выполненных заявок** | **Процент невыполненных заявок** |
| 1 | 632 | 36 | 23 | 573 | 6% | 94% |
| 2 | 632 | 77 | 41 | 514 | 12% | 88% |
| 4 | 632 | 153 | 72 | 407 | 24% | 76% |
| 6 | 632 | 237 | 89 | 306 | 38% | 63% |
| 8 | 632 | 313 | 89 | 230 | 50% | 50% |
| 10 | 632 | 370 | 91 | 171 | 59% | 41% |
| 12 | 632 | 450 | 70 | 112 | 71% | 29% |
| 14 | 632 | 511 | 64 | 57 | 81% | 19% |
| 16 | 632 | 557 | 12 | 63 | 88% | 12% |
| 18 | 632 | 597 | 0 | 35 | 94% | 6% |
| 20 | 632 | 608 | 0 | 24 | 96% | 4% |
| 22 | 632 | 609 | 0 | 23 | 96% | 4% |
| 26 | 632 | 609 | 0 | 23 | 96% | 4% |
| 30 | 632 | 609 | 0 | 23 | 96% | 4% |

# Анализ результатов моделирования

Продемонстрируем зависимость доли успешно выполненных заявок от количества операторов и технических администраторов в виде графика:

В целом видим следующую базовую характеристику: при текущей постановке задачи на одного оператора должно приходиться 4-5 технических администраторов. Обратимся к результатам моделирования, где у нас было настроено 4 оператора и 20 администраторов:

1. Коэффициент использования равняется 0.982 для операторов, 0.855 для администраторов, 0.006 для очереди к операторам и 0 для каждой из очереди заявок;
2. Среднее время пребывания заявки в очереди к оператору (на рассмотрение) – 2 минуты 33 секунды, а к администратору (на исполнение) – 50 секунд;
3. В среднем за всё время моделирования было занято 3.928 из 4 операторов и 17.093 из 20 администраторов;
4. Всего в очередь ожидания на исполнение (то есть, к администраторам) вошло всего лишь 46 транзактов.

Из этой статистики мы видим, что виртуальные сотрудники работали на пике нагрузки и что нашу модель от реальной ситуации отличает «бесперебойность», то есть отсутствие обеденных перерывов и прочих непредвиденных, затрудняющих работу обстоятельств. При этом если посмотреть на отчёт о работе другой модели – где 4 оператора и 26 администраторов, - то мы там увидим снижение коэффициента использования администраторов до 0.662, но не увидим никаких изменений для операторов. Таким образом, видим, что критическим местом всё ещё являются операторы, поскольку именно они подают ~~патроны~~ заявки для исполнения.

# Заключение

В ходе выполнения курсовой работы была спроектирована имитационная модель обслуживания офисного печатного оборудования. Было выполнено предварительное проектирование в виде описания модели с помощью различных схем, непосредственное проектирование в виде написания программы в среде *GPSS World*, а также была проведена серия экспериментов для выявления слабых и сильных сторон модели.

Созданная модель является первым шагом на пути к воссозданию рабочей ситуации, с помощью которого можно было бы получать прогнозы и таким образом предупреждать все возможные проблемы. Дальнейшая работа должна быть направлена на усложнение различных аспектов модели: работа со временем внутри рабочего дня (обеденные перерывы, задержки в общении между отделами и др.), законы появления заявок и их обработки на различных этапах (возможно, следует добавить даже другие этапы) и прочее.

Для созданной модели была проанализирована роль производительности различных элементов в неудачах при выполнении поступающих заявок. Полученные результаты могут считаться предварительными выводами, которые, однако, не должны сильно видоизмениться в дальнейшем.