



**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 «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5**  
**Определение вероятности отказа**

Студент \_\_\_\_\_ Соколов Ефим \_\_\_\_\_  
Группа \_\_\_\_\_ ИУ7-73Б \_\_\_\_\_  
Дисциплина \_\_\_\_\_ Моделирование \_\_\_\_\_

Преподаватель:

\_\_\_\_\_ Рудаков И.В.  
подпись, дата                      Фамилия, И.О.

Оценка \_\_\_\_\_

Москва — 2022 г.

## Задание

В информационный центр приходят клиенты через интервал времени  $10 \pm 2$  минуты. Если все три имеющихся оператора заняты, клиенту отказывают в обслуживании. Операторы имеют разную производительность и могут обеспечивать обслуживание среднего запроса пользователя за  $20 \pm 5$ ;  $40 \pm 10$ ;  $40 \pm 20$ . Клиенты стремятся занять свободного оператора с максимальной производительностью. Полученные запросы сдаются в накопитель. Откуда выбираются на обработку. На первый компьютер запросы от 1 и 2-ого операторов, на второй – запросы от 3-его. Время обработки запросов первым и 2-м компьютером равны соответственно 15 и 30 мин. Промоделировать процесс обработки 300 запросов.

На рисунке 1 приведена схема системы.

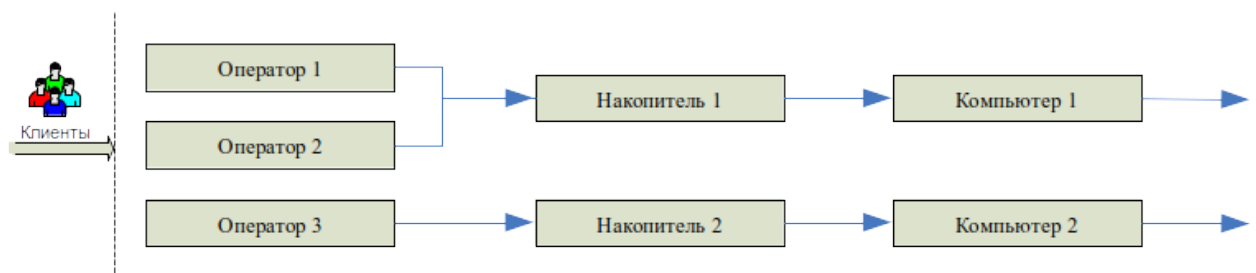


Рисунок 1: Схема системы

Для выполнения поставленного задания необходимо создать концептуальную модель в терминах СМО, определить эндогенные и экзогенные переменные и уравнения модели. За единицу системного времени выбрать 0,01 минуты.

## Теоретическая часть

В процессе взаимодействия клиентов с информационным центром возможно:

- режим нормального обслуживания, т.е. клиент выбирает одного из

свободных операторов, отдавая предпочтение тому у которого меньше номер;

- режим отказа в обслуживании клиента, когда все операторы заняты.

## Переменные и уравнения имитационной модели

Эндогенные переменные отвечают за время обработки задания  $i$ -ым оператором, время решения этого задания  $j$ -ым компьютером.

Экзогенные переменные - это число клиентов, которых обслужили и получившие отказ.

## Концептуальная схема

На рисунке 2 приведена концептуальная схема системы в терминах СМО.

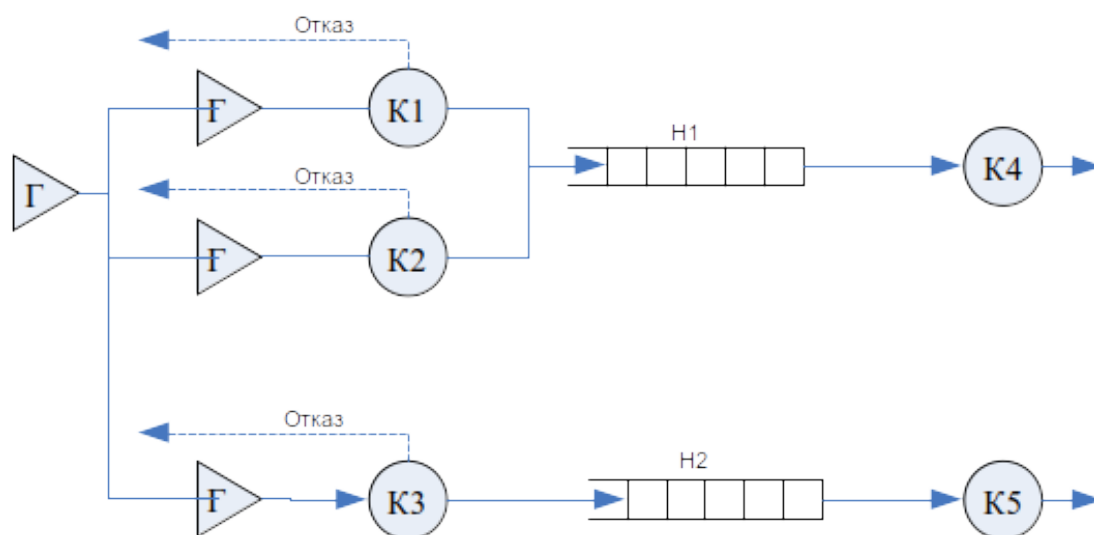


Рисунок 2: Концептуальная схема в терминах СМО

$$P_{react} = \frac{C_{react}}{C_{react} + C_{proc}}$$

# Листинги

На листинге 1 приведена функция обработки инкремента шага по времени.

Листинг 1: Обработка инкремента шага по времени

```
1 def one_step(generator, operators, processors, request_info, generate_new=True):
2     if generate_new:
3         request = generator.update_time(time_unit)
4         if request:
5             request_info['gen'] += 1
6             i_operator = pick_operator(operators)
7             if i_operator == -1:
8                 request_info['loss'] += 1
9             else:
10                operators[i_operator].accept_request(request)
11
12    for cur_operator in operators:
13        cur_operator.update_time(time_unit)
14
15    for cur_processor in processors:
16        res = cur_processor.update_time(time_unit)
17        if res == 'request done':
18            request_info['ok'] += 1
```

На листинге 2 приведена код цикла, обеспечивающие пошаговую работу (моделирования) системы.

Листинг 2: Цикл пошагового моделирования системы

```
1 def modeling(generator, operators, processors, total_incoming_requests):
2     request_info = { 'gen': 0, 'loss': 0, 'ok': 0 }
3
4     while request_info['gen'] < total_incoming_requests:
5         one_step(generator, operators, processors, request_info)
6
7     while request_info['loss'] + request_info['ok'] < total_incoming_requests:
8         one_step(generator, operators, processors, request_info, False)
9
10    return request_info
```

На листинге 3 приведена код функции main: задание входных параметров и получение результата моделирования.

Листинг 3: Функция main

```
1 def main():
2     client_generator = Generator(EvenDistribution(8, 12))
3
4     first_queue = []
5     second_queue = []
6
7     operators = [
8         Operator(first_queue, EvenDistribution(15, 25)),
9         Operator(second_queue, EvenDistribution(30, 50)),
```

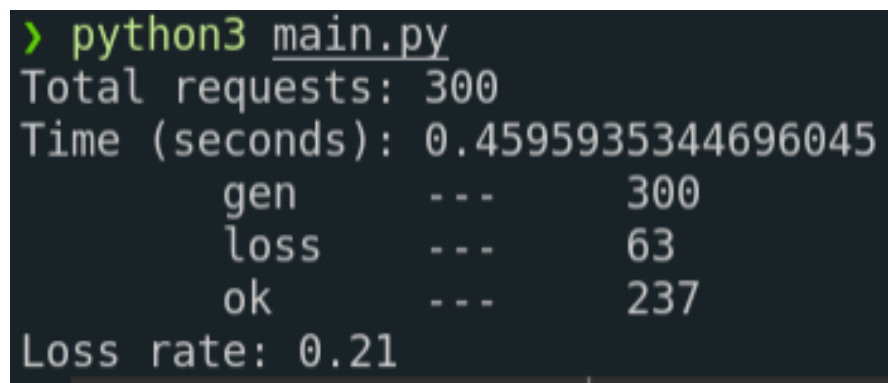
```

10 Operator(second_queue, EvenDistribution(20, 60))
11 ]
12
13 processors = [
14 Processor(first_queue, EvenDistribution(15, 15)),
15 Processor(second_queue, EvenDistribution(30, 30))
16 ]
17
18 total_requests = 300
19
20 t_start = time()
21 res = modeling(client_generator, operators, processors, total_requests)
22
23 print('Time (seconds)', time() - t_start)
24 for key in res.keys():
25     print(f"\t{key}\t——\t{res[key]}")
26
27 print('Loss rate', res['loss'] / total_requests)

```

## Результаты выполнения работы

На рисунках 3-5 приведены результаты работы системы с 300, 1000, 5000 заявок.



```

> python3 main.py
Total requests: 300
Time (seconds): 0.4595935344696045
      gen      --      300
      loss     --      63
      ok       --     237
Loss rate: 0.21

```

Рисунок 3: Результат моделирования системы с 300 заявками

При моделировании системы с 300 заявками процент потерянных заявок составляет 22%.

```
> python3 main.py
Total requests: 1000
Time (seconds): 1.572176218032837
      gen      ---      1000
      loss     ---      202
      ok       ---      798
Loss rate: 0.202
```

Рисунок 4: Результат моделирования системы с 1000 заявками

```
> python3 main.py
Total requests: 5000
Time (seconds): 7.30997896194458
      gen      ---      5000
      loss     ---      1080
      ok       ---      3920
Loss rate: 0.216
```

Рисунок 5: Результат моделирования системы с 5000 заявками