# Отчет по лабораторной работе №11

**Дисциплина: Имитационное моделирование**

Лобанова Полина Иннокентьевна

# Содержание

# Список иллюстраций

# Список таблиц

# 1 Цель работы

Реализовать модель системы массового обслуживания M|M|1.

# 2 Задание

В систему поступает поток заявок двух типов, распределённый по пуассоновскому закону. Заявки поступают в очередь сервера на обработку. Дисциплина очереди - FIFO. Если сервер находится в режиме ожидания (нет заявок на сервере), то заявка поступает на обработку сервером.

# 3 Выполнение лабораторной работы

1. Построила модель с помощью CPNTools. Использовала три отдельных листа: на первом листе описала граф системы, на втором — генератор заявок, на третьем — сервер обработки заявок.

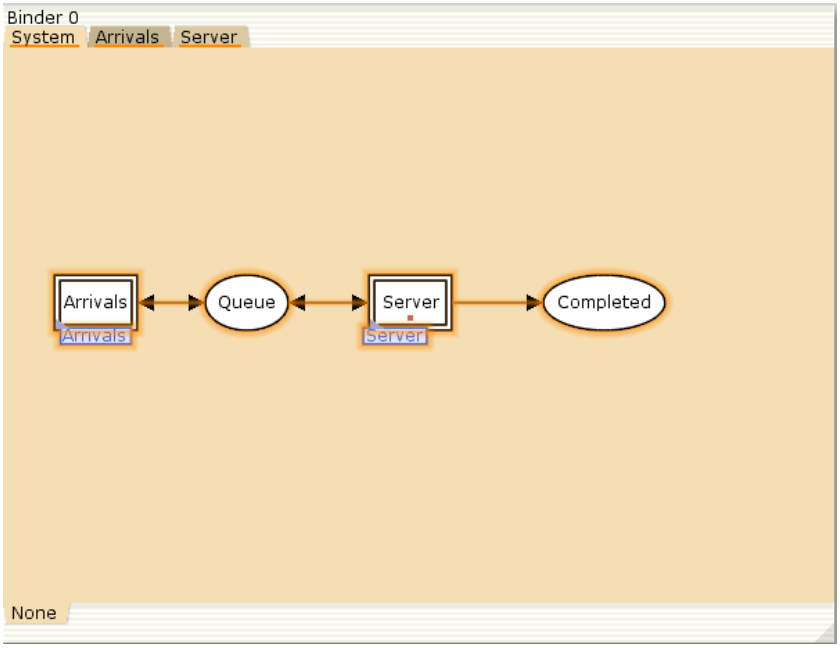


Рис. 3.1: *Граф сети системы обработки заявок в очереди*

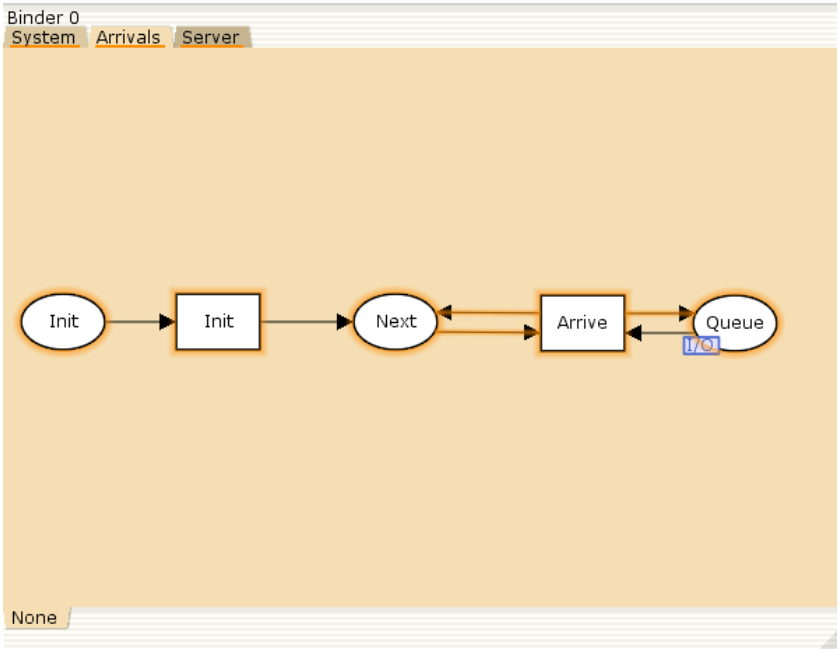


Рис. 3.2: *Граф генератора заявок системы*

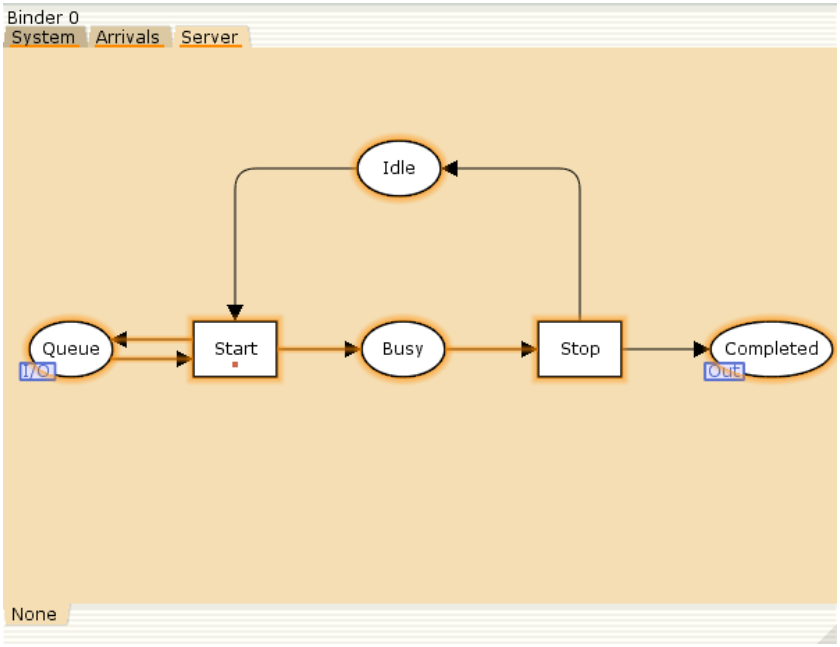


Рис. 3.3: *Граф процесса обработки заявок на сервере системы*

2. Задала декларации системы, переменные модели и определила функции системы.


Рис. 3.4: *Задание деклараций*

3. Задала параметры модели на графах сети.

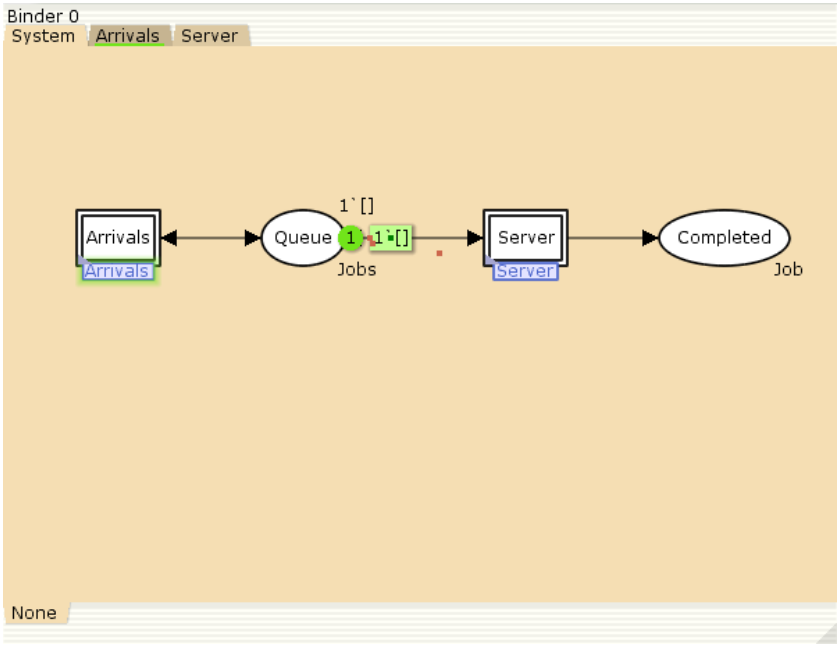


Рис. 3.5: *Параметры элементов основного графа системы обработки заявок в очереди*

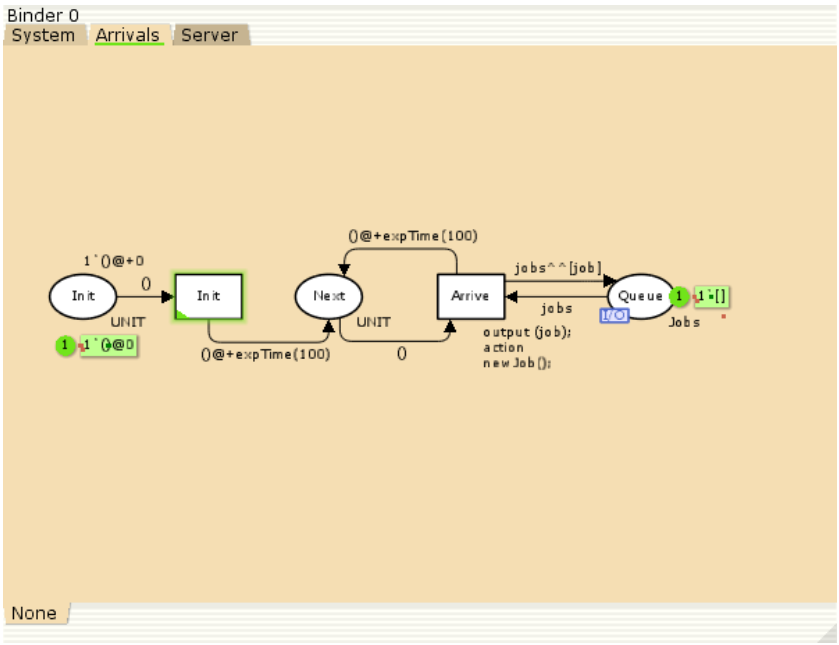


Рис. 3.6: *Параметры элементов генератора заявок системы*

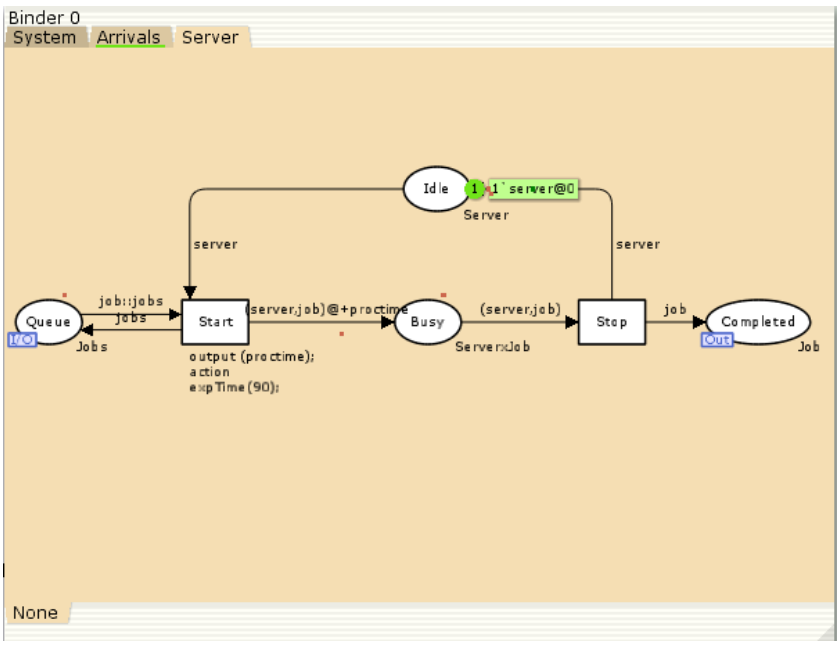


Рис. 3.7: *Параметры элементов обработчика заявок системы*

4. Для мониторинга параметров очереди системы M|M|1 потребовалась палитра Monitoring. Выбрала Break Point (точка останова) и установила её на переход Start. После этого в разделе меню Monitor появился новый подраздел, который назвала Ostanovka. В этом подразделе внесла изменения в функцию Predicate, которая будет выполняться при запуске монитора.

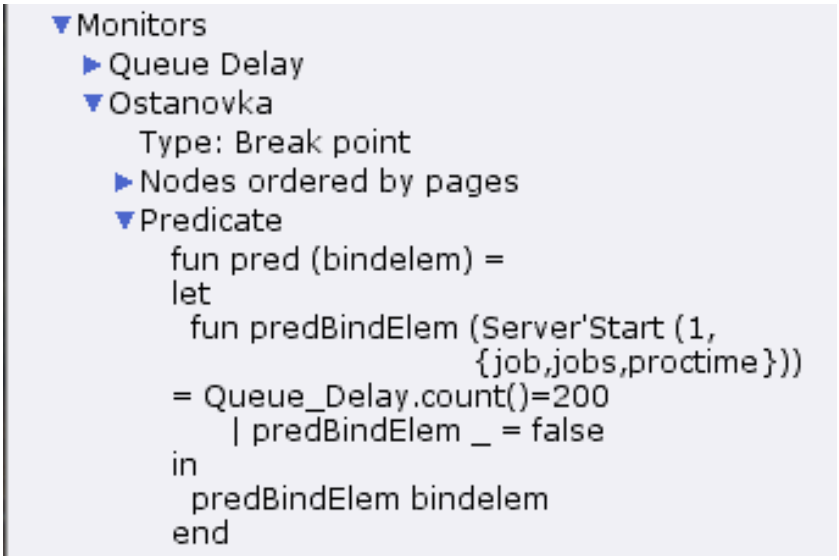


Рис. 3.8: *Функция Predicate монитора Ostanovka*

5. Далее необходимо было определить конструкцию Queue_Delay.count(). С помощью палитры Monitoring выбрала Data Call и установила на переходе Start. Появившийся в меню монитор назвала Queue Delay.Изменила функция Observer так, чтобы получить значение задержки в очереди.

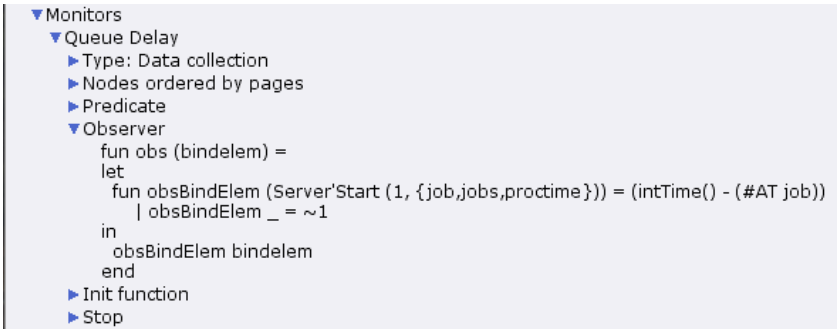


Рис. 3.9: *Функция Observer монитора Queue Delay*

6. После запуска программы на выполнение в каталоге с кодом программы появился файл Queue_Delay.log, содержащий в первой колонке — значение задержки очереди, во второй — счётчик, в третьей — шаг, в четвёртой —

время. С помощью gnuplot построила график значений задержки в очереди, выбрав по оси x время, а по оси y — значения задержки.

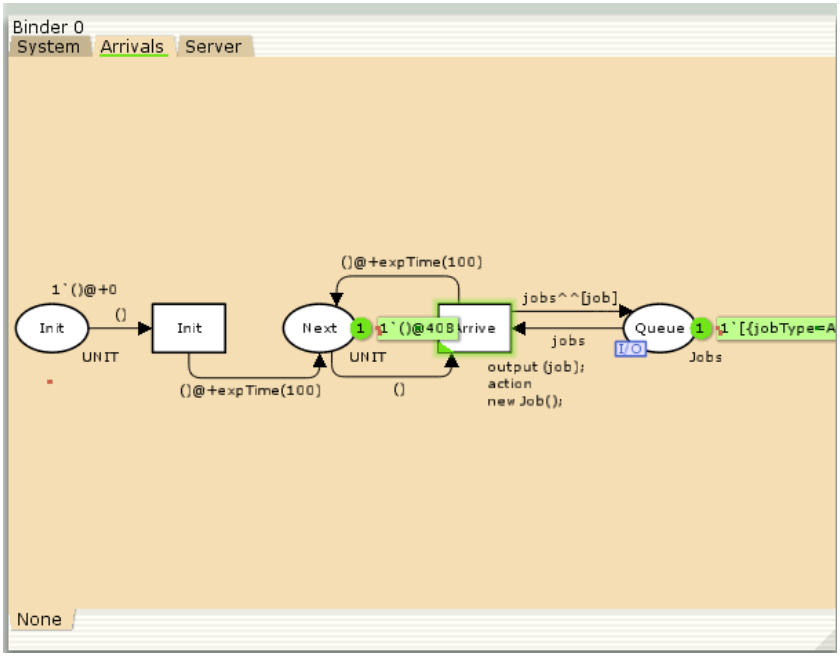


Рис. 3.10: *Запуск системы обработки заявок в очереди*

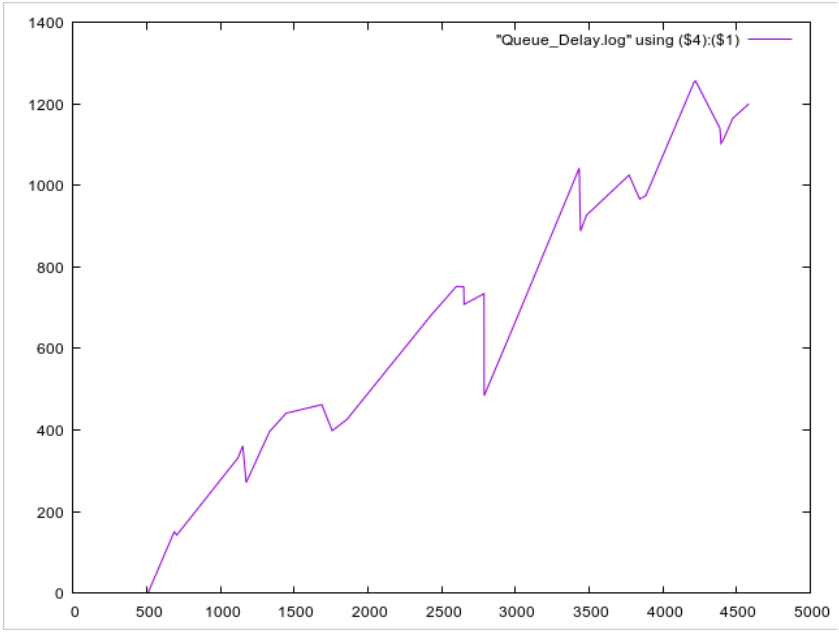


Рис. 3.11: *График изменения задержки в очереди*

7. Посчитала задержку в действительных значениях. Для этого с помощью палитры Monitoring выбрала Data Call и устанавила на переходе Start. Появившийся в меню монитор называла Queue Delay Real. Функцию Observer изменила. После запуска программы на выполнение в каталоге с кодом программы появился файл Queue_Delay_Real.log с содержимым, аналогичным содержимому файла Queue_Delay.log, но значения задержки имеют действительный тип.

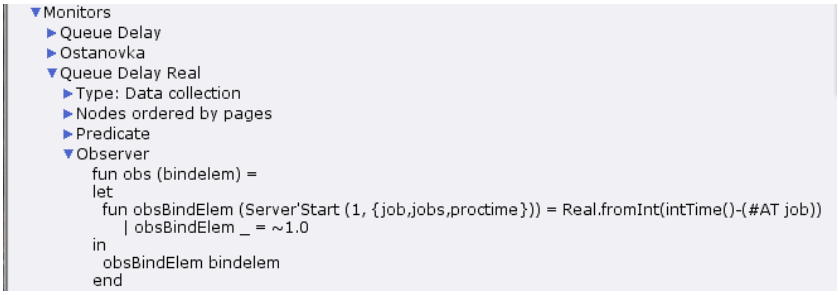


Рис. 3.12: *Функция Observer монитора Queue Delay Real*

8. Посчитала, сколько раз задержка превысила заданное значение. С помощью палитры Monitoring выбрала Data Call и установила на переходе Start. Монитор назвала Long Delay Time. Функцию Observer изменила. В декларациях задала глобальную переменную (в форме ссылки на число 200).

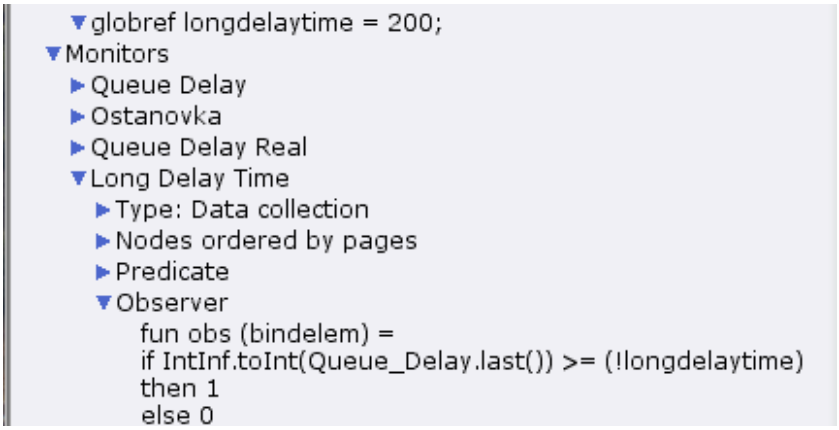


Рис. 3.13: *Функция Observer монитора Long Delay Time*

9. С помощью gnuplot построила график, демонстрирующий, в какие периоды времени значения задержки в очереди превышали заданное значение 200.

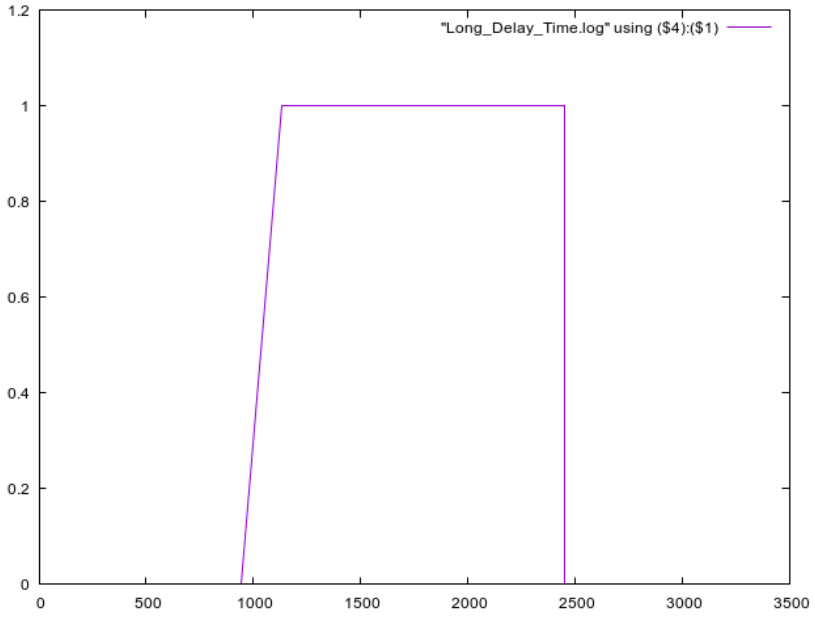


Рис. 3.14: *Периоды времени, когда значения задержки в очереди превышали заданное значение*

# 4 Выводы

Я реализовала модель системы массового обслуживания M|M|1.

# Список литературы