МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра телекоммуникационных систем и вычислительных средств (ТС и ВС)

Отчет по лабораторной работе №9
по дисциплине
Теория массового обслуживания

по теме:

ВВЕДЕНИЕ В SIMULINK. ПОСТРОЕНИЕ МОДЕЛЕЙ ОДНОФАЗНОЙ ОДНОКАНАЛЬНОЙ СМО С ОГРАНИЧЕННОЙ ОЧЕРЕДЬЮ И ОБРАТНОЙ СВЯЗЬЮ

Студент:
Группа ИА-331 *Я.А Гмыря*

Предподаватель:
Преподаватель *А.В Андреев*

Новосибирск 2025 г.

СОДЕРЖАНИЕ

ЦЕЛЬ И ЗАДАЧИ

Цель:

Использование средств инженерного программного пакета MATLAB для построения, отладки и тестирования моделей систем массового обслуживания (СМО)

Задание к лабораторной работе

Построить с помощью MATLAB Simulink систему массвого обслуживания M/M/1.

ХОД РАБОТЫ

2.1 Основные блоки СМО

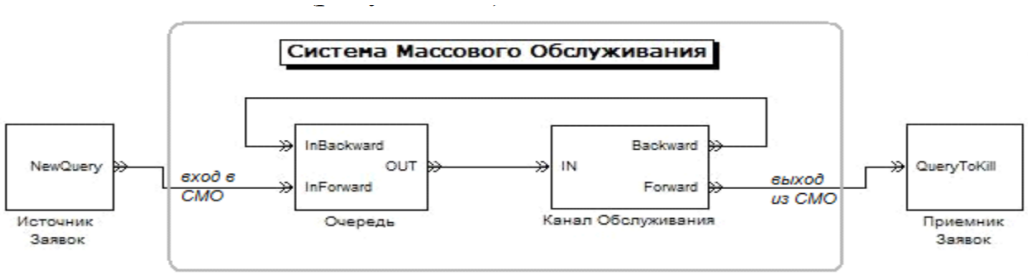

Рисунок 1 — Схема проектируемой модели

СМО должна содержать генератор заявок, очередь заявок, сервер обработки заявок. В случае, если заявка отброшена, то она должна снова поступать в очередь.

2.2 Генератор заявок

Для генерации заявок в Simulink есть специальный блок **entity generator**

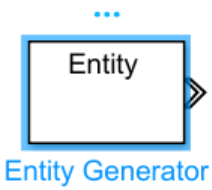

Рисунок 2 — Генератор заявок

2.2.1 Настройки генератора заявок

В блоке можно гибко настроить временные интервалы между заявками. Simulink предлагает 3 разных способа. Один из них - задать интервалы с помощью matlab кода. Поскольку мы создаем M/M/1 модель, я задам экспоненциальное распределение со средним $\frac{1}{\lambda}$

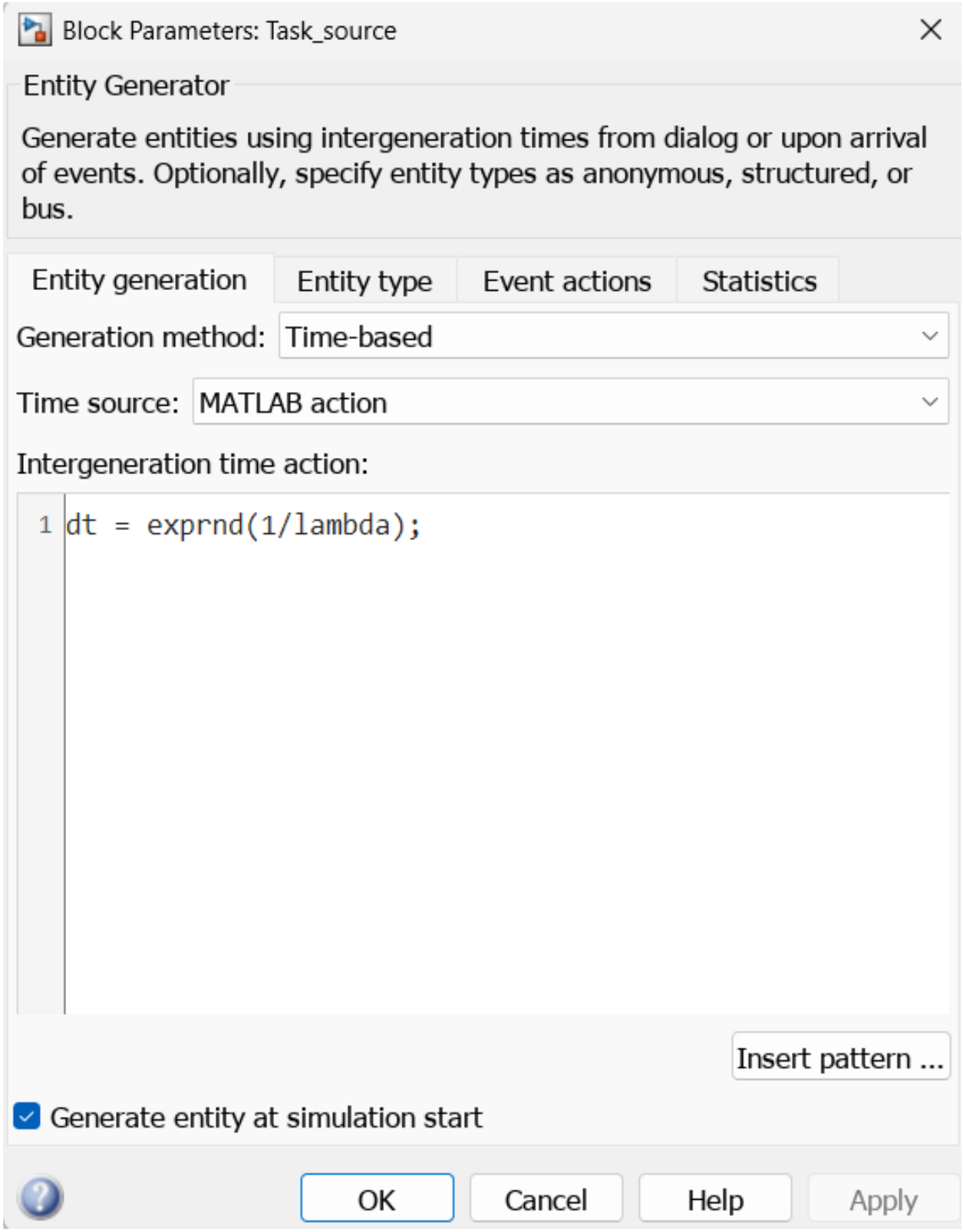

Рисунок 3 — Настройка временных интервалов поступления заявок

Из блока entity generator выходят не просто какие-то числа, а структуры. Их тоже можно гибко настроить. Я создам структуру, которая будет иметь приоритет и флаг reject.

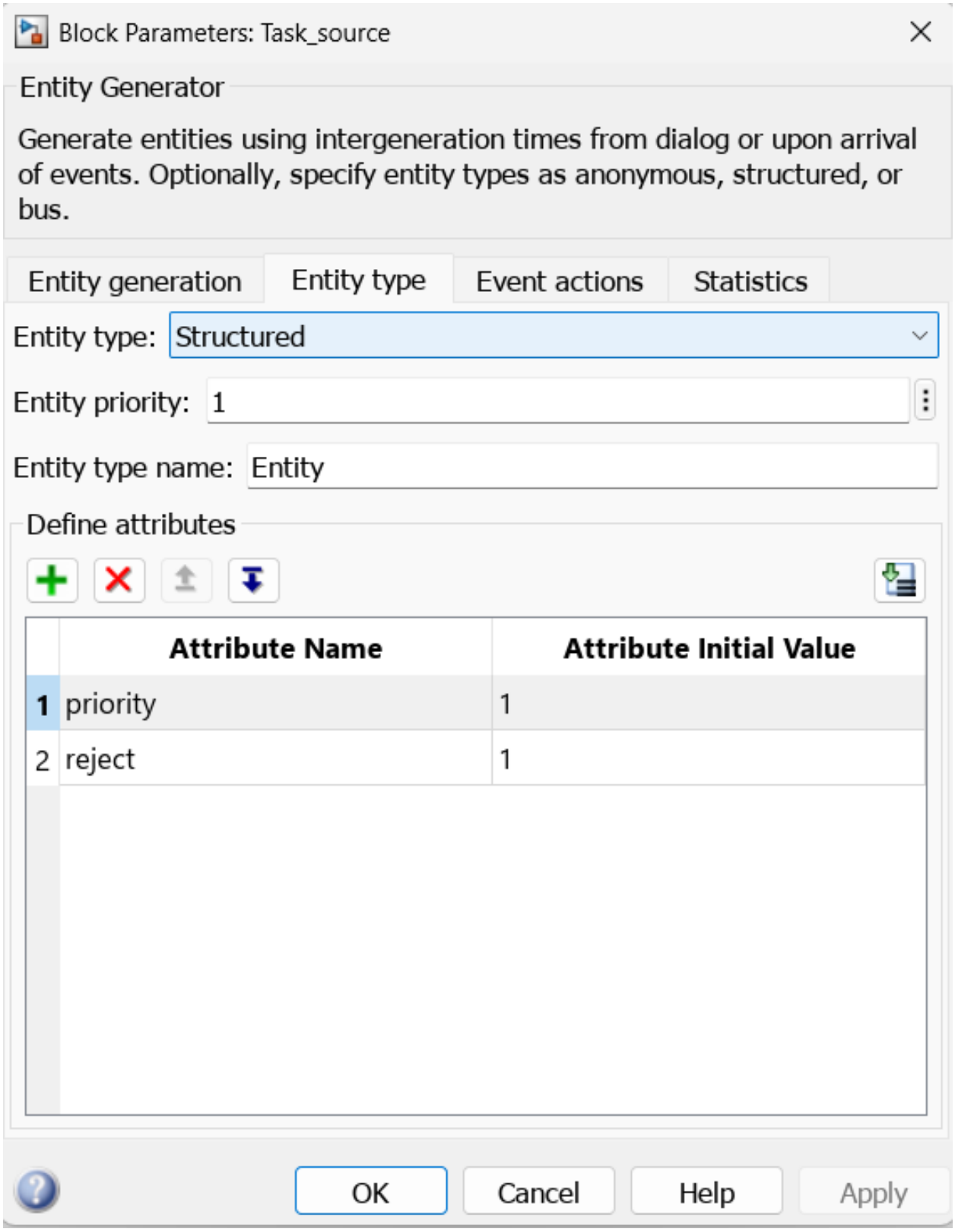

Рисунок 4 — Настройка атрибутов структур

Также можно гибко задать логику работы entity generator. В этой логике я буду рандомно задавать приоритет от 1 до 10.

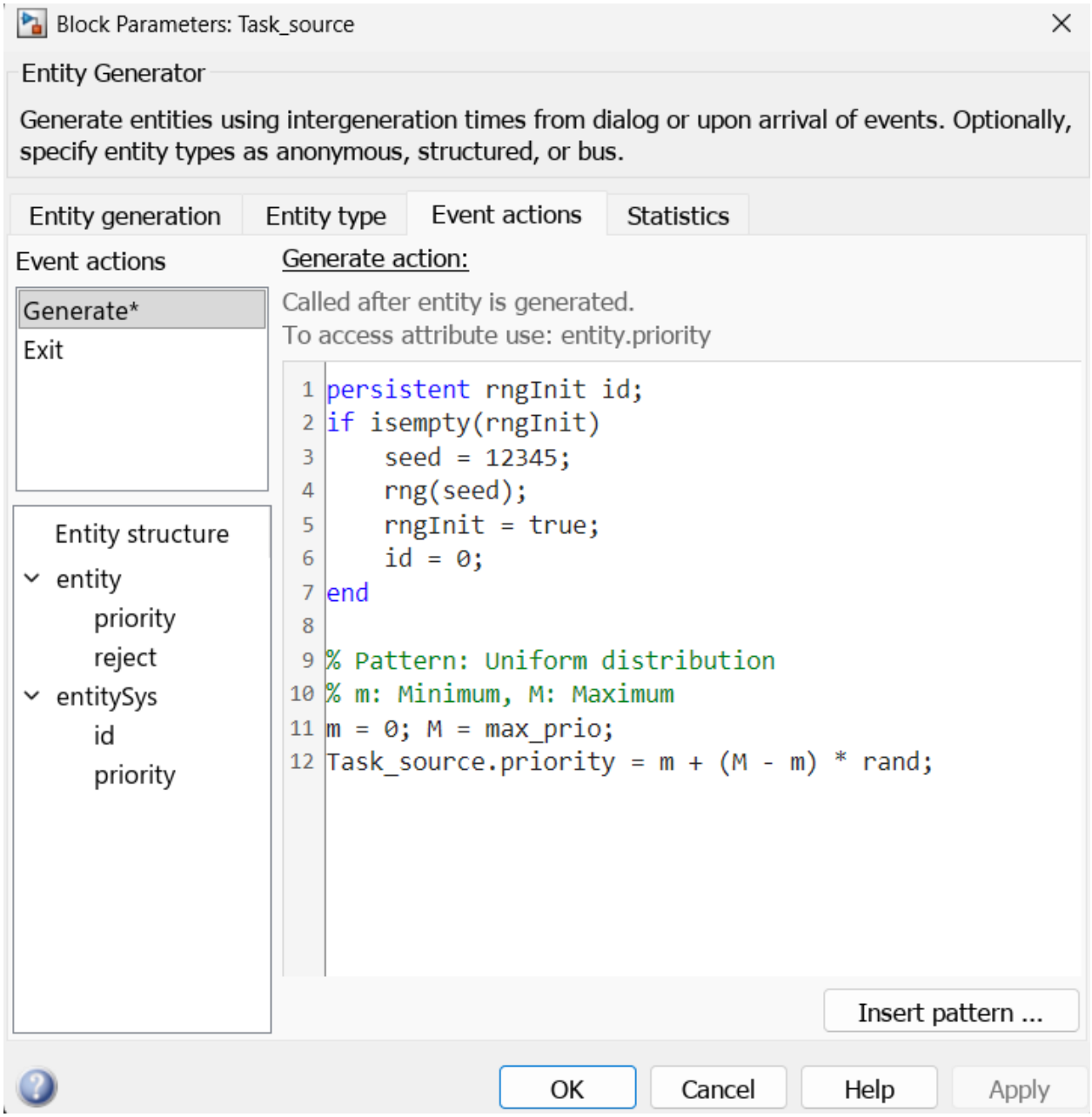

Рисунок 5 — Логика формирования заявок

Дополнительно matlab позволяет выводить статистику блока. Стастика высчитывается автоматически. Каждую статистику можно вывести на блок **scope**, который построит график.

2.3 Очередь

Для создания очереди используется блок **entity queue**

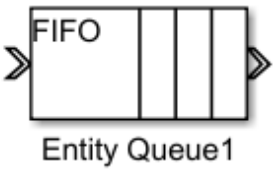

Рисунок 6 — Очередь заявок

2.3.1 Настройки очереди заявок

Очередь тоже можно настроить достаточно гибко. Можно создать размер очереди (50), задать порядок хранения заявок в очереди: FIFO, LIFO или по атрибуту структуры (как в моем случае). Я буду хранить entity в порядке возрастания приоритета, т.е на сервер в первую очередь поступят entity с минимальным (на самом деле максимальным) приоритетом.

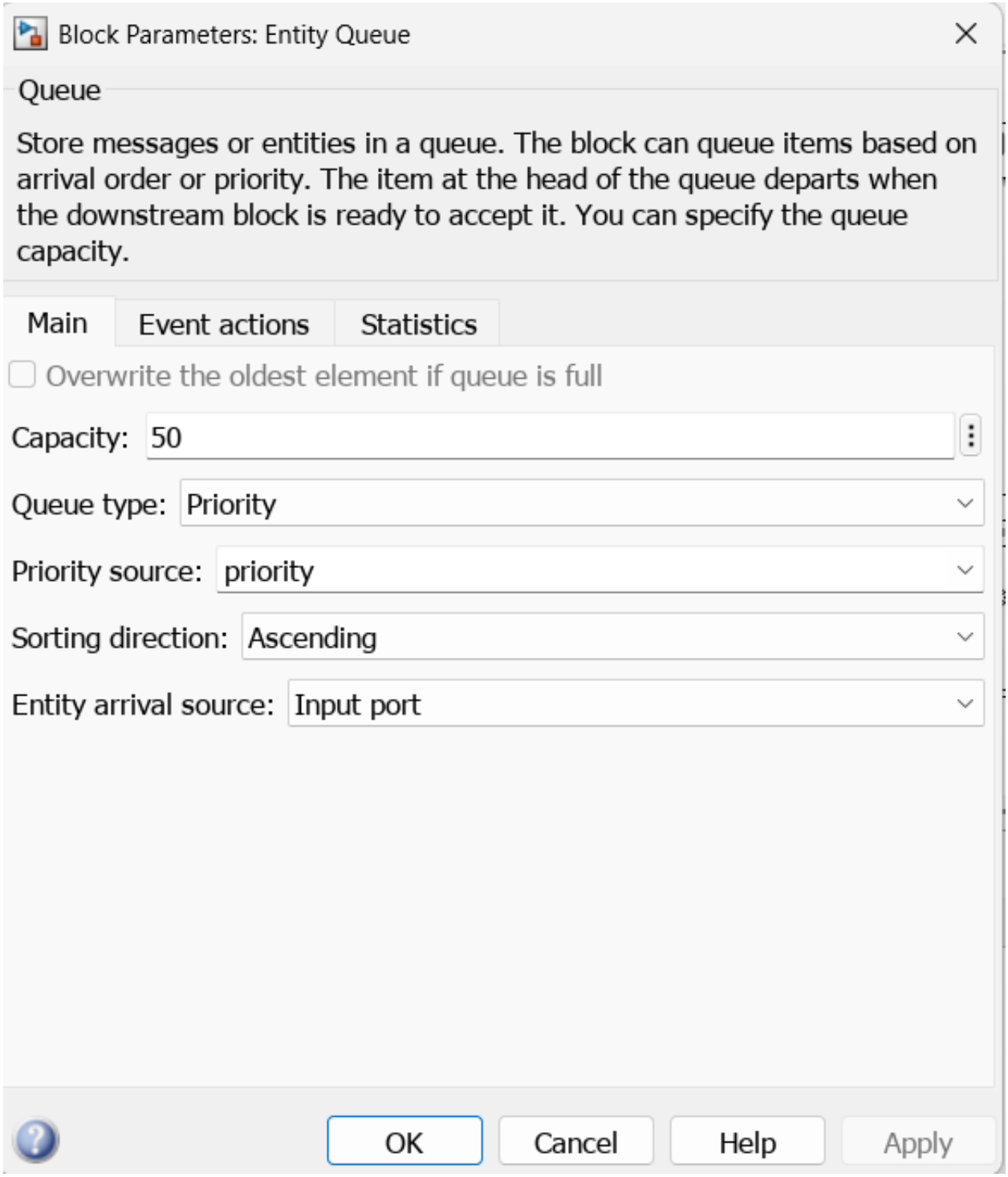

Рисунок 7 — Основная настройка очереди

Также можно вывести набор метрик на блок scope

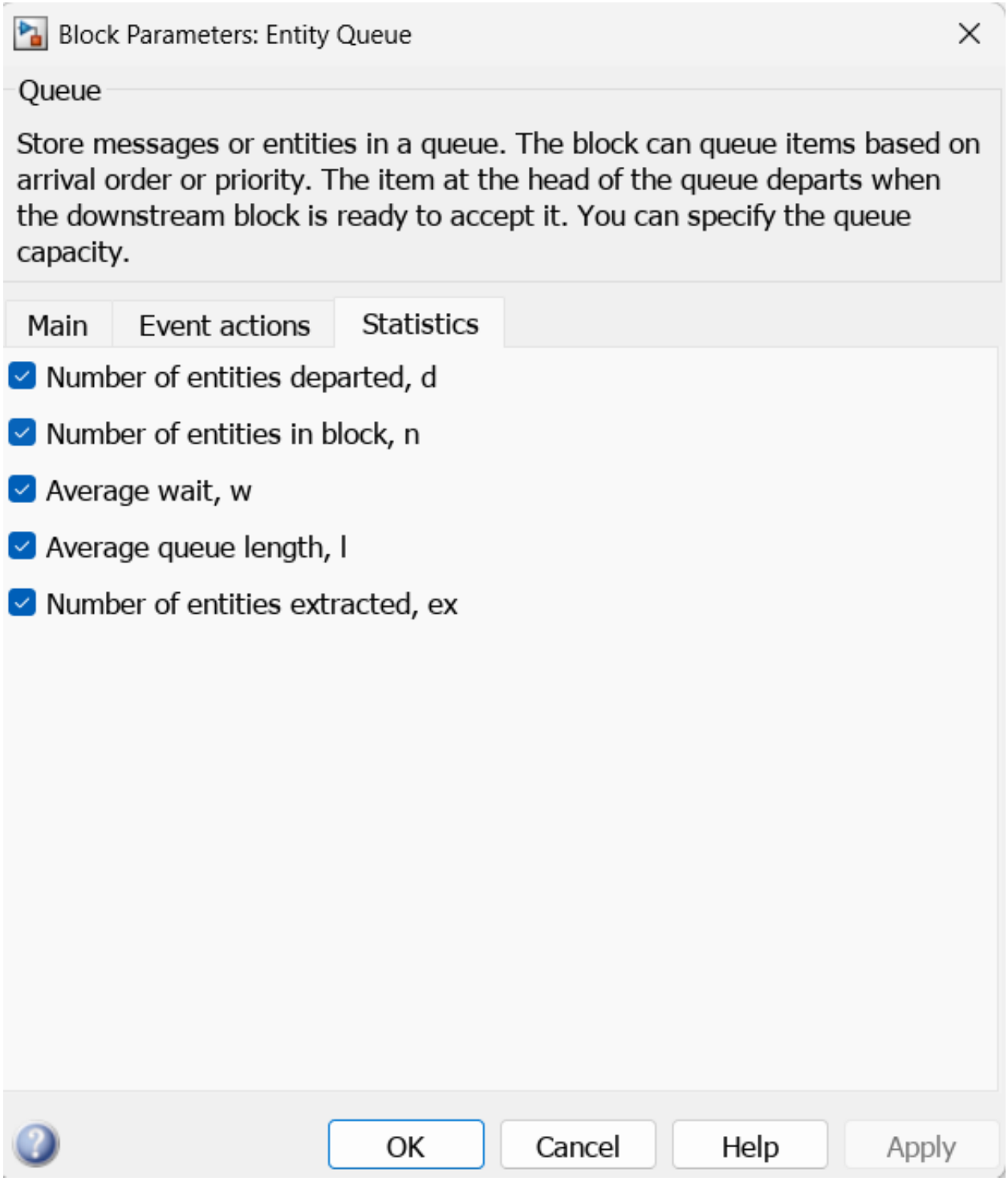

Рисунок 8 — Статистика очереди

Также можно задать дополнительную логику для работы очереди, но я ничего не добавлял.

2.4 Сервер

Для обработки заявок существует специальный блок **entity server**.

2.4.1 Настройка сервера

У сервера можно настроить собственную очередь, т.е за раз сервер может брать больше чем 1 заявку. Время обработки заявки можно задать с помощью matlab кода. Я задам экспоненциальное время обработки заявок со средним $\frac{1}{u}$.

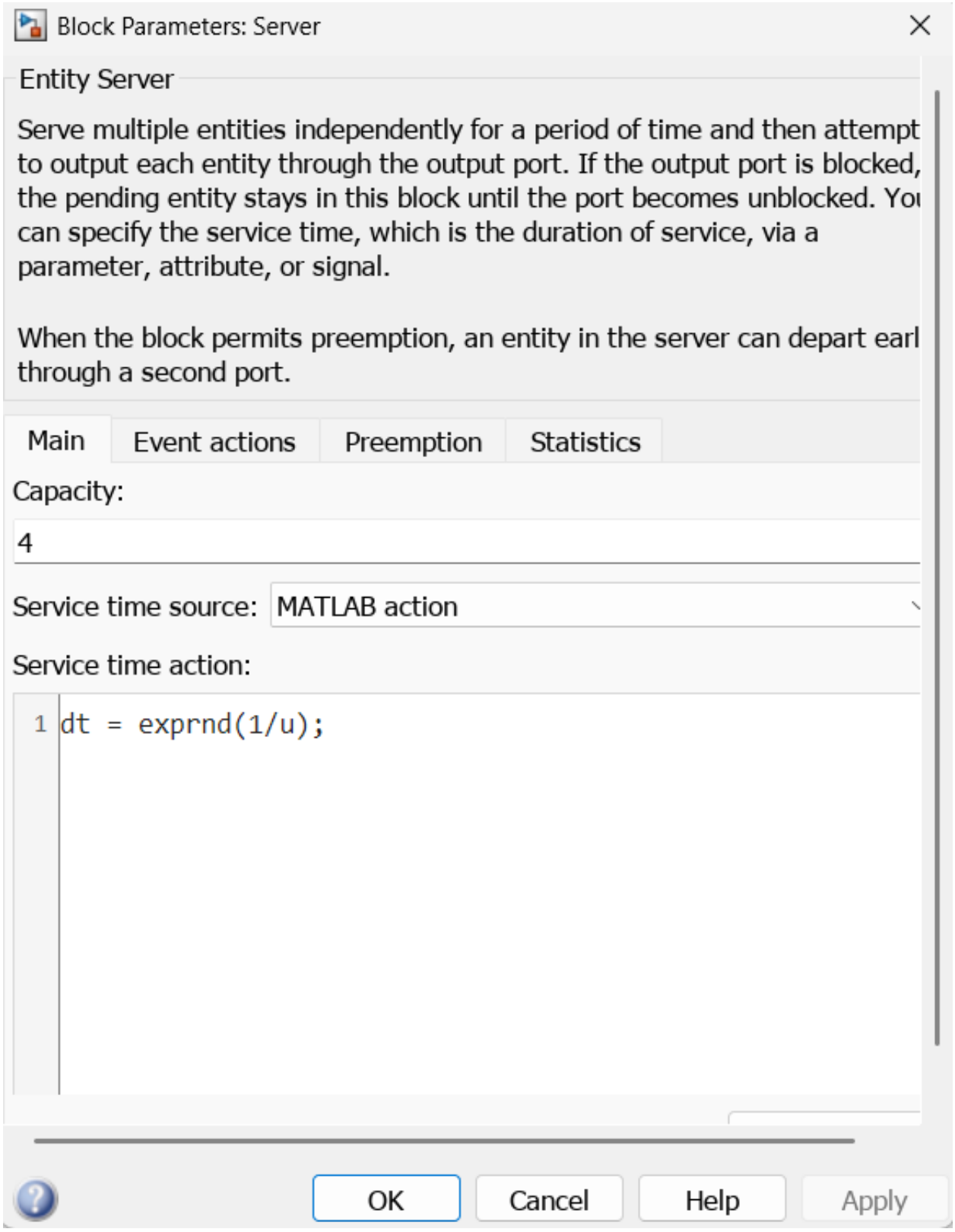

Рисунок 9 — Основная настройка сервера

Также можно задать дополнительно логику обработки заявок. Я буду с некоторой вероятностью отбрасывать заявки. Если заявка обработана, то reject = 1, в ином случае reject = 2.

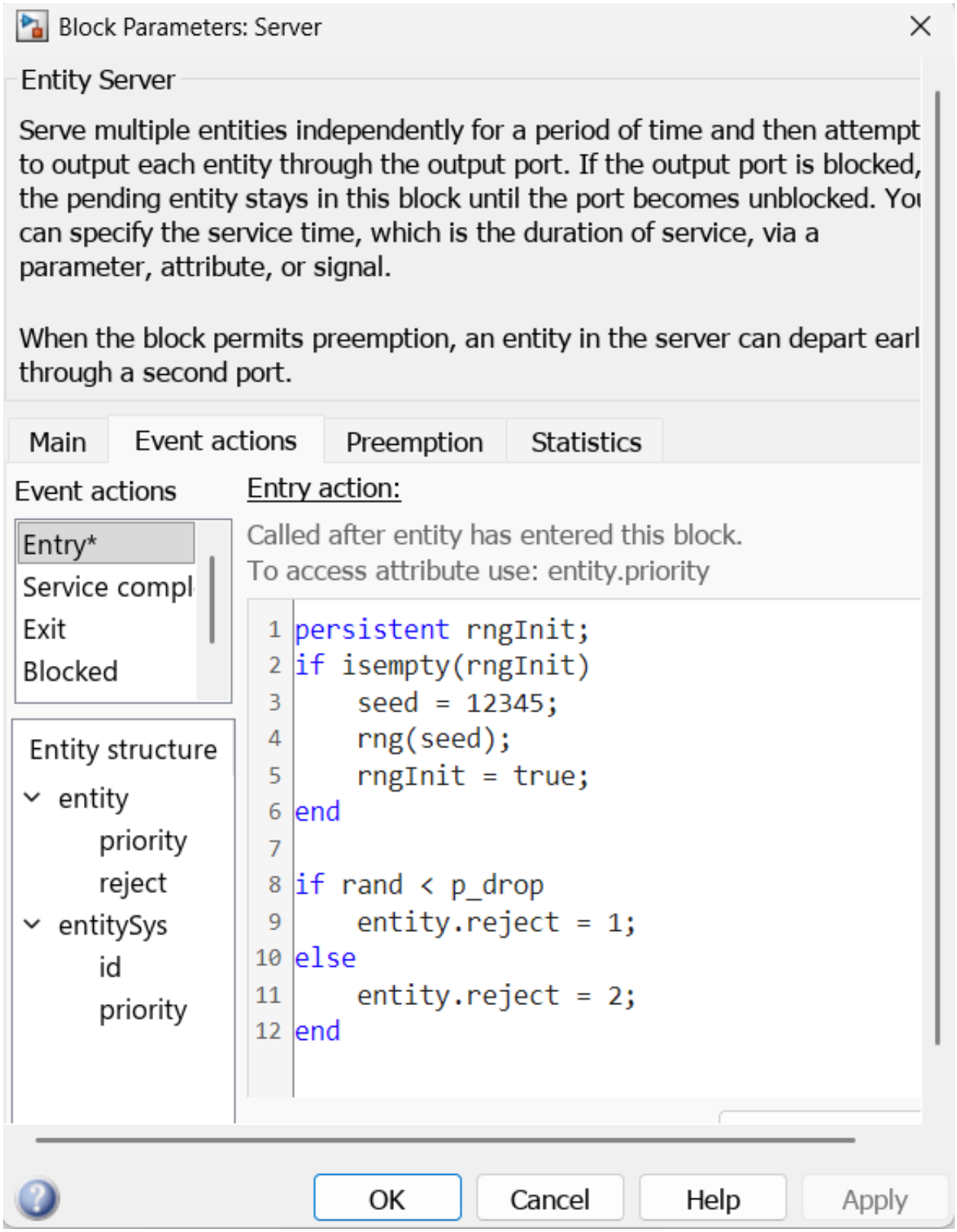

Рисунок 10 — Логика обработки заявок

Сервер тоже ведет расчет метрик, которые можно выводить на scope.

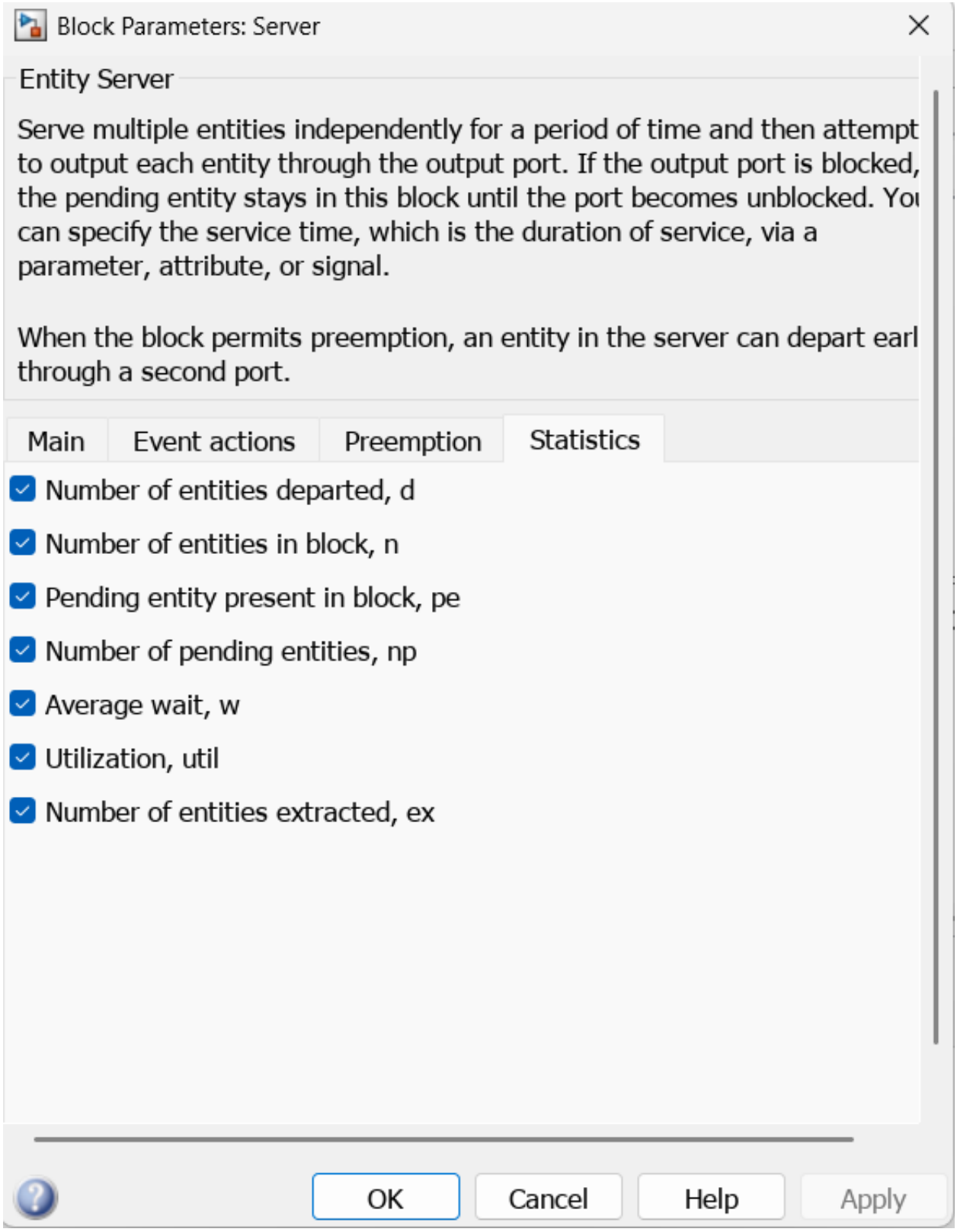

Рисунок 11 — Статистика сервера

2.5 Создание обратной связи

2.5.1 Entity output switch

Под обратной связью подразумевается возврат отброшенных заявок обратно в очередь. Для этого необходим блок **entity output switch**.

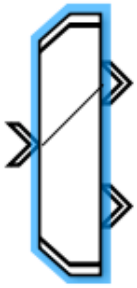

Рисунок 12 — Entity output switch

Логика блока такая-же, как у коммутатора - определенным образом распределять заявки по каналам.

2.5.2 Настройка entity output switch

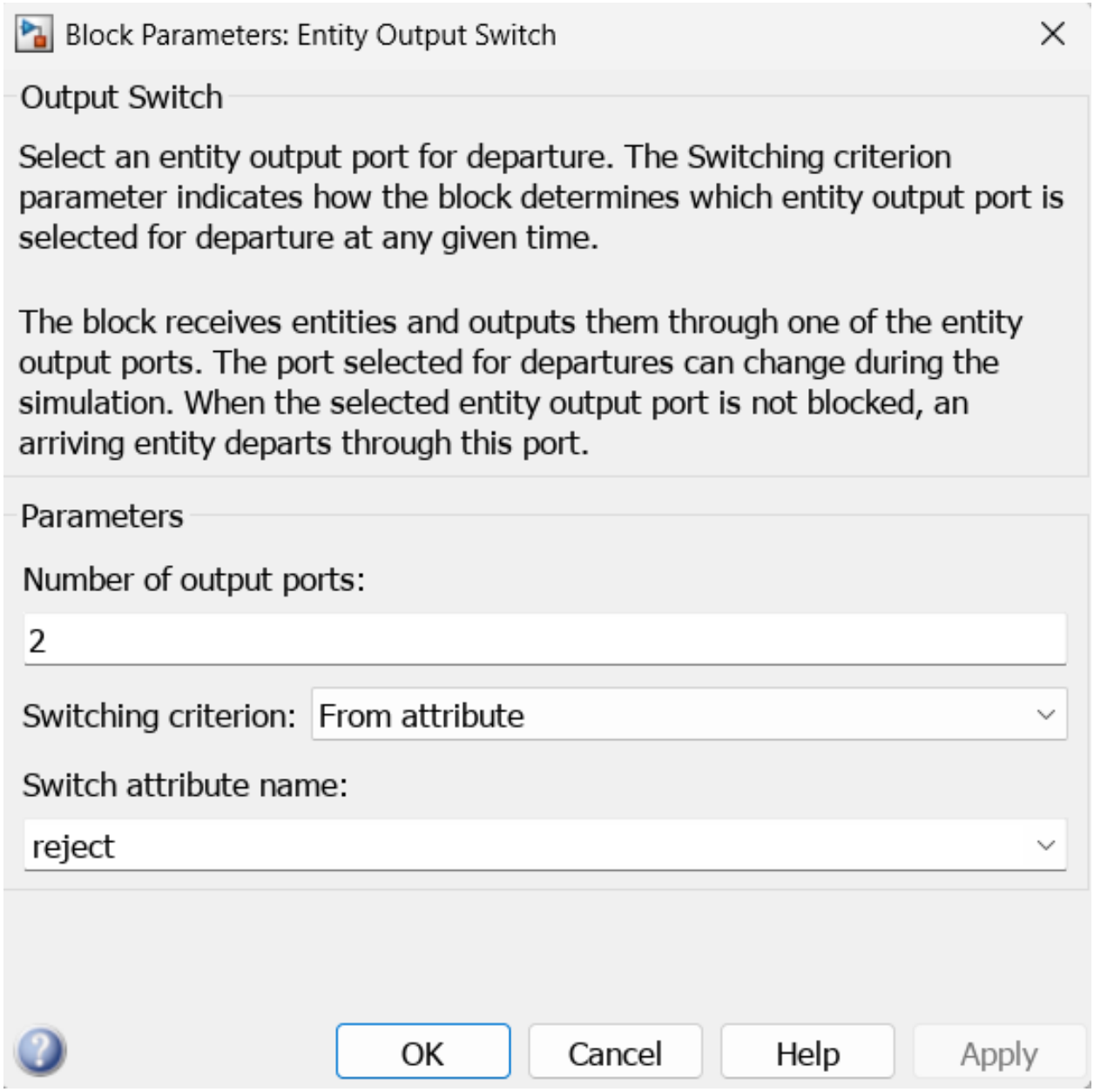

Рисунок 13 — Настройки

Заявки будут распределяться по каналам на основе атрибута reject. Если reject = 1, то заявка пойдет по каналу, который просто выводит заявку из системы, а если reject = 2, то заявка пойдет по каналу обратной связи обратно в очередь.

2.6 Entity input switch

Для того, чтобы до конца реализовать обратную связь, нужно мультиплексировать выходы entity output switch и entity generator, потому что очередь имеет только 1 вход. В роли мультиплексора можно использовать блок **entity input switch**.

Рисунок 14 — Entity input switch

2.7 Настройка entity input switch

Блок можно гибко настроить. Можно настроить кол-во входов и алгоритм, по которому с каждого входа будет выбираться одна заявка (2 заявки одновременно подать на очередь нельзя). Я выбрал алгоритм Round Robin, чтобы заявки с обоих входов поступали поочереди.

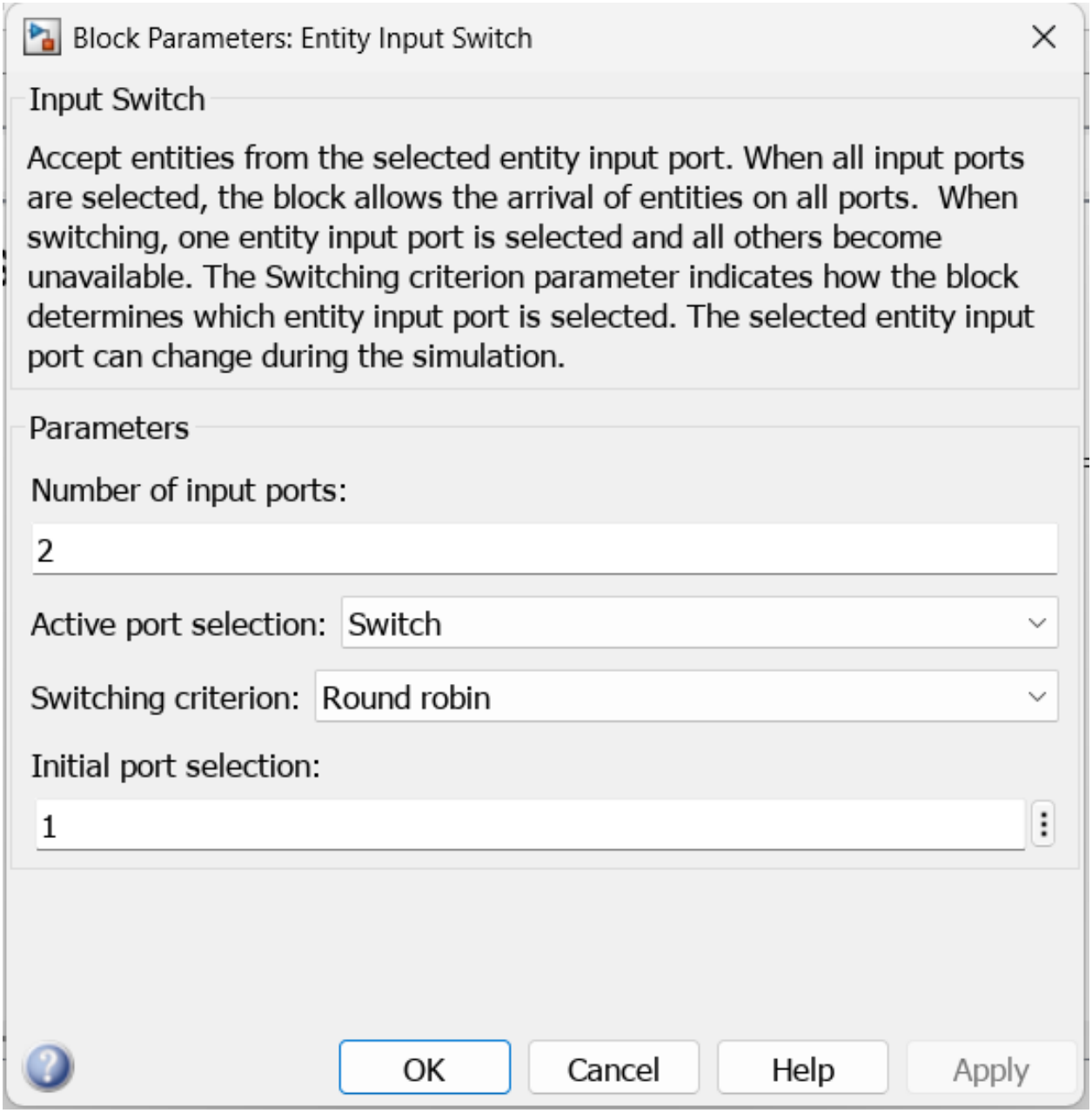

Рисунок 15 — Настройка entity input switch

2.8 Workspace переменные

На скринах выше использовались переменные, которые можно посмотреть и изменить в Workspace переменных

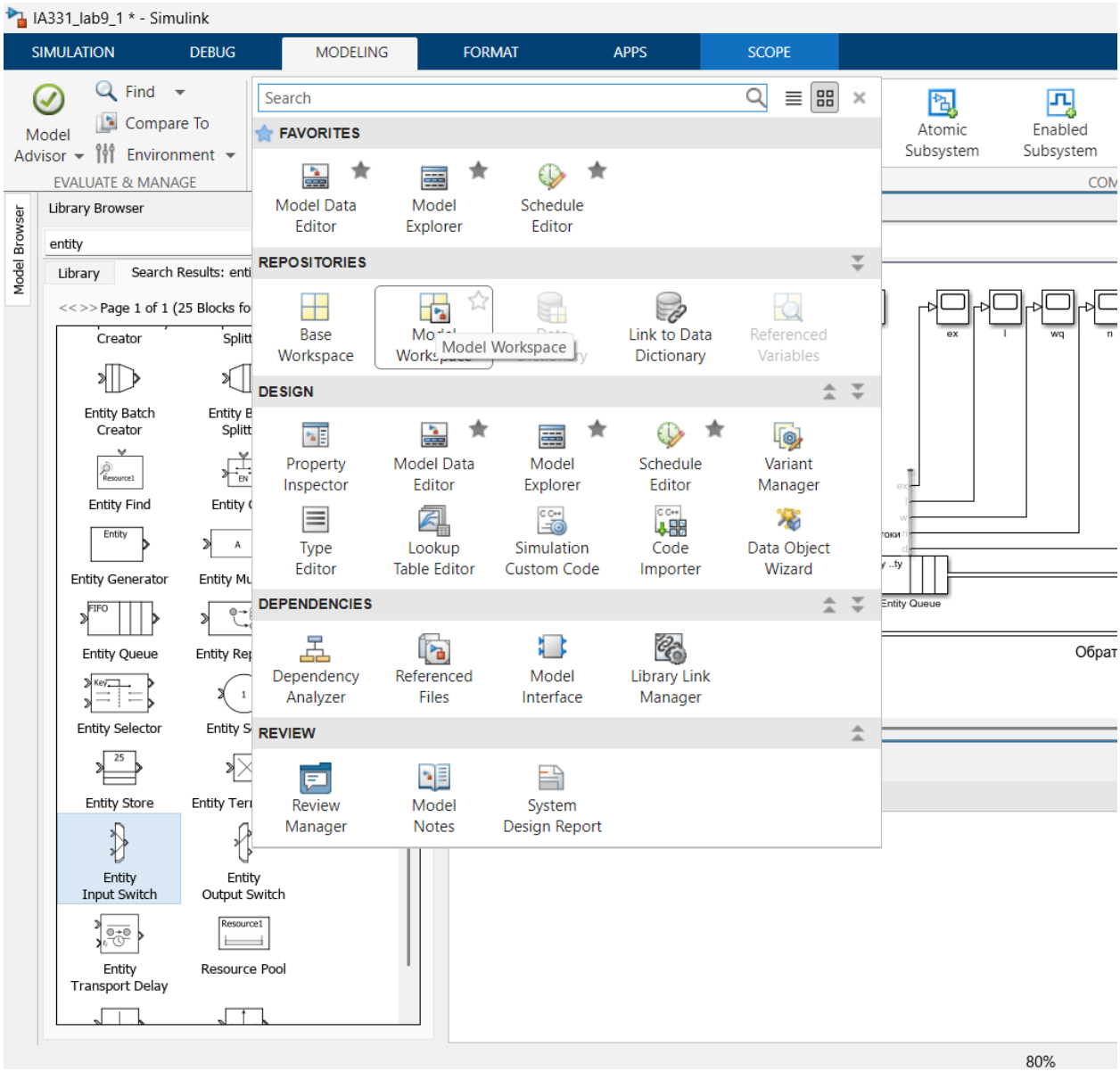

Рисунок 16 — Workspace переменные

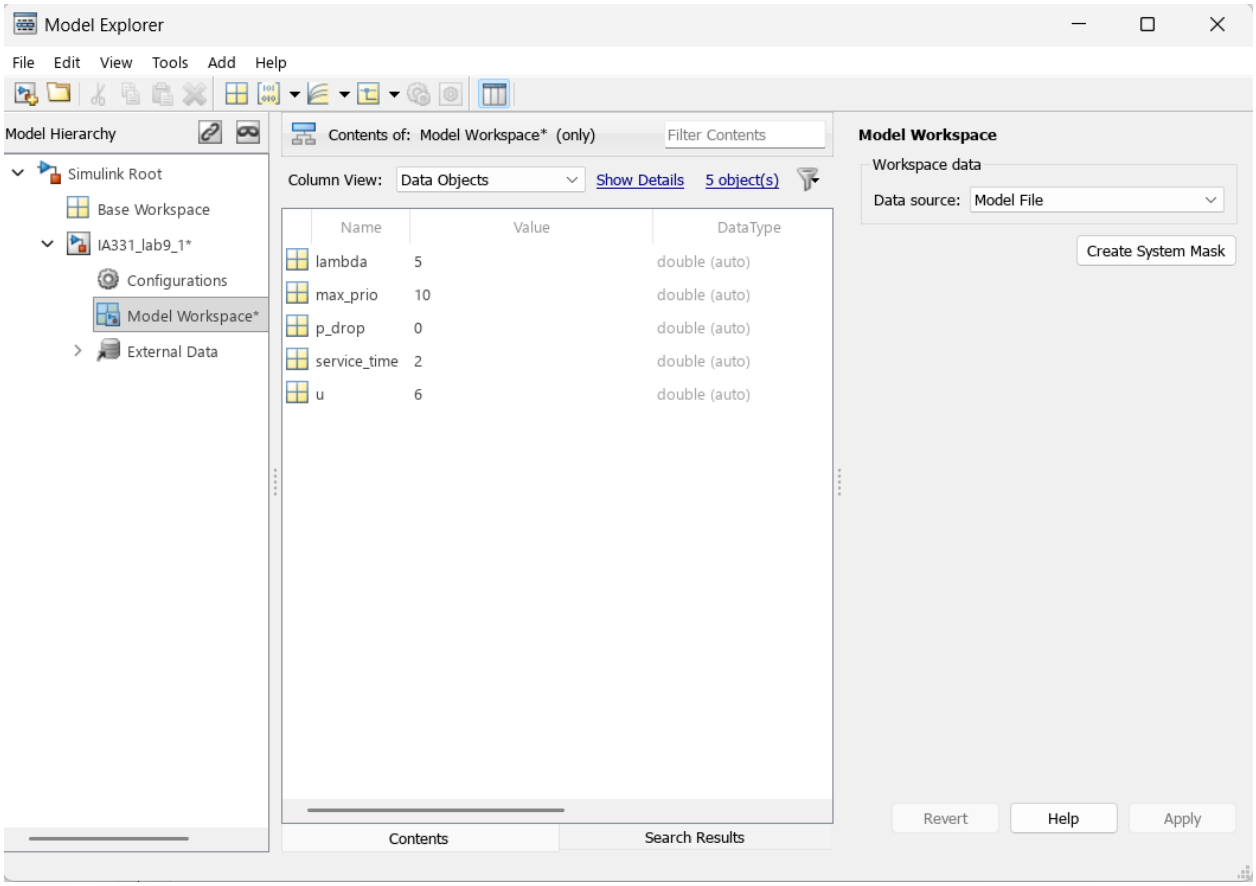

Рисунок 17 — Workspace переменные

2.9 Результат работы

2.9.1 Модель

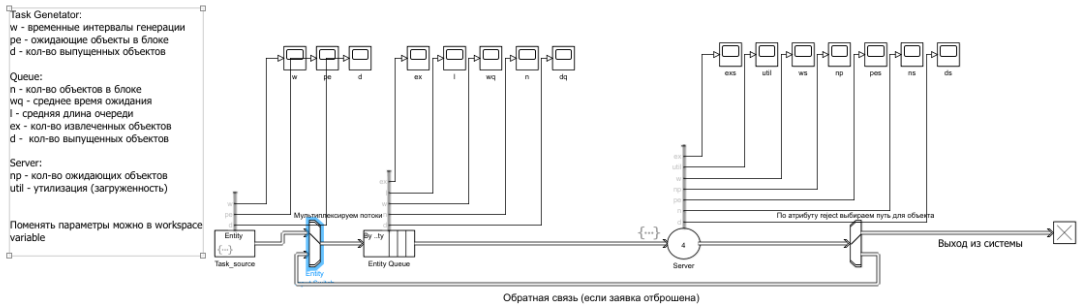

Рисунок 18 — СМО M/M/1

2.9.2 Графики

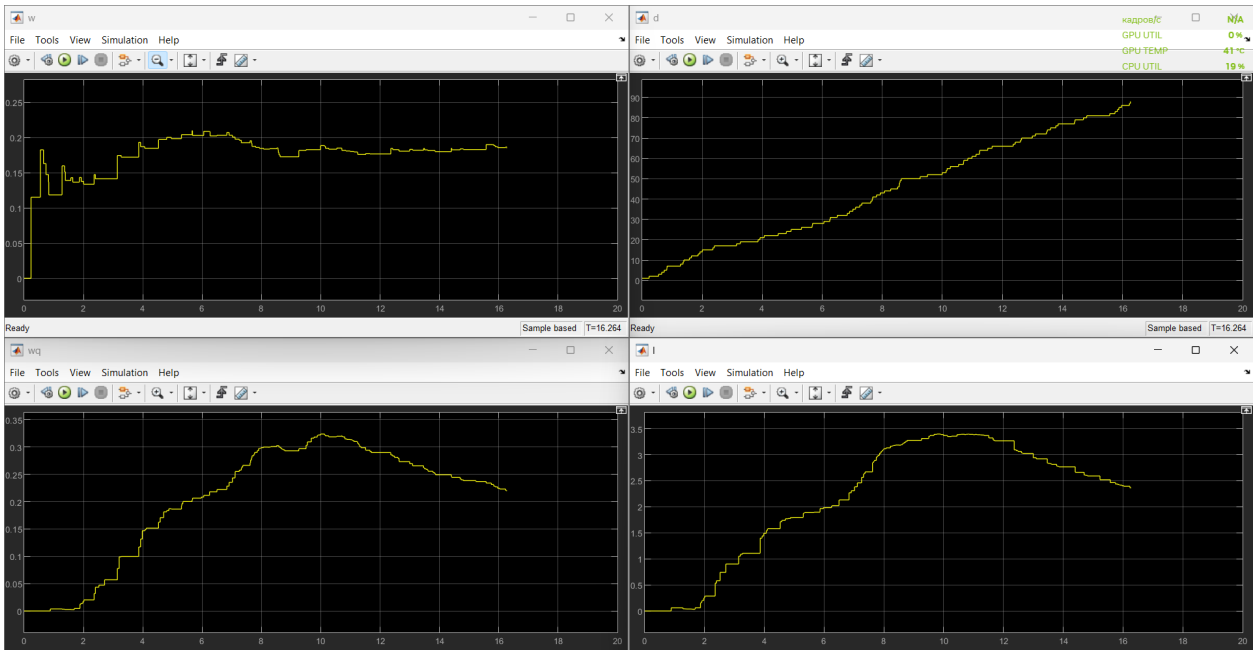

Рисунок 19 — Метрики СМО

Верхние графики: w - временные интервалы генерации заявок, d - кол-во созданных заявок. Нижние: l - средняя длина очереди, wq - среднее время ожидания в очереди. Можем заметить, что график w похож на экспоненциальное распределение. За все время генератор создал 55 заявок. На протяжении эксперимента длина очереди и время ожидания в ней росло, это значит, что СМО не справлялась с нагрузкой.

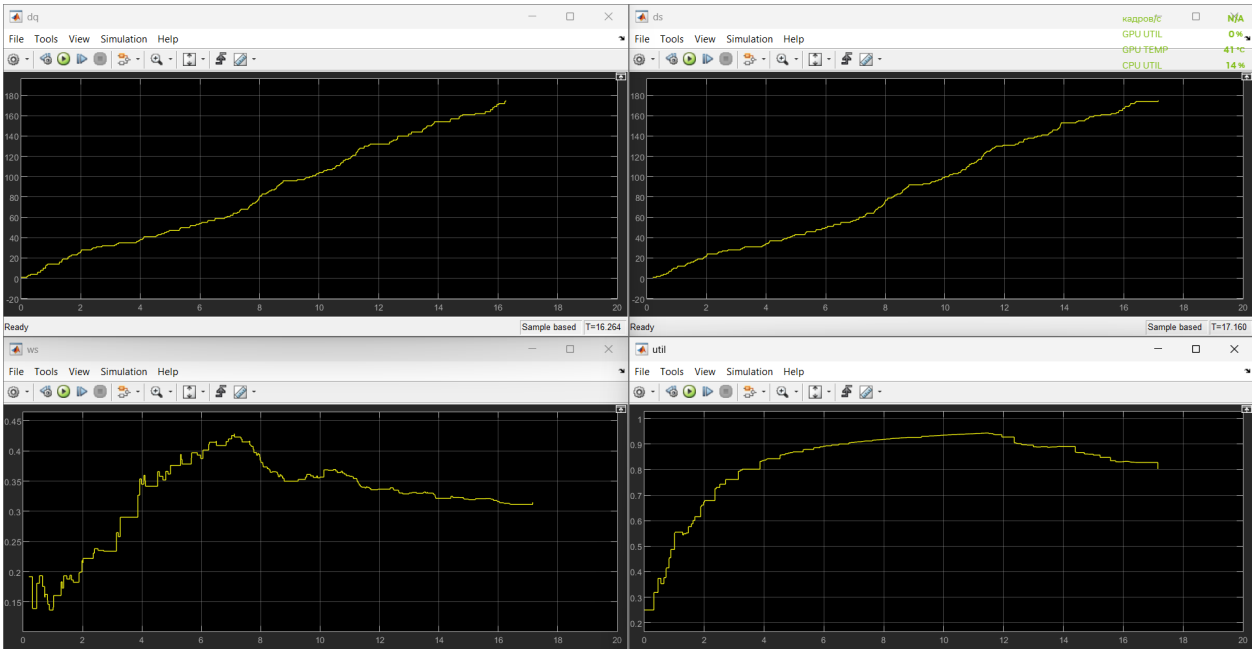

Рисунок 20 — Метрики СМО

Верхние графики: dq - кол-во заявок, прошедших через очередь, ds - кол-во заявок, прошедших через сервер. Нижние: wq - среднее время ожидания на сервере, util - утилизация сервера (нагрузка).

Нужно обратить внимание на то, что на графиках dq и ds кол-во заявок вдвое больше, чем на графике d (кол-во сгенерированных заявок). Это связано с наличием обратной связи. Вероятность отказа в системе 0.5, поэтому было созданно около 90 заявок, а в очереди и сервере побывало около 180 заявок.

ВЫВОД

В ходе работы я познакомился с новым инструментом Matlab Simulink, который позволяет моделировать СМО. С помощью Simulink построил модель системы M/M/1