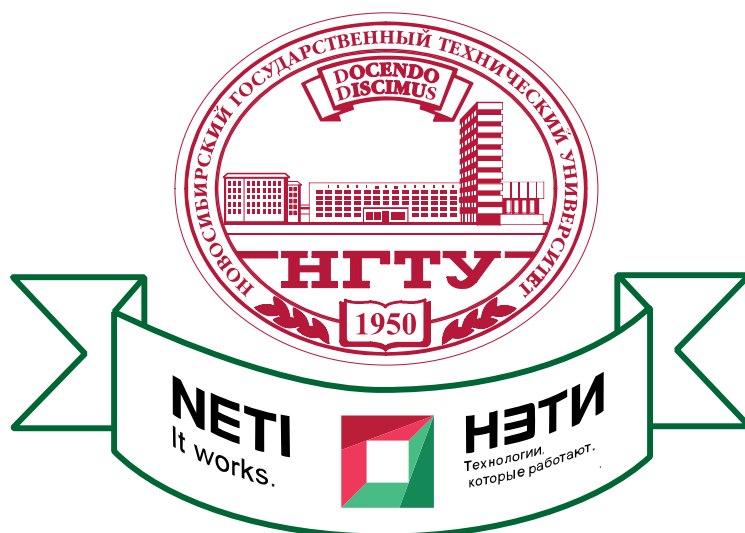


Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования

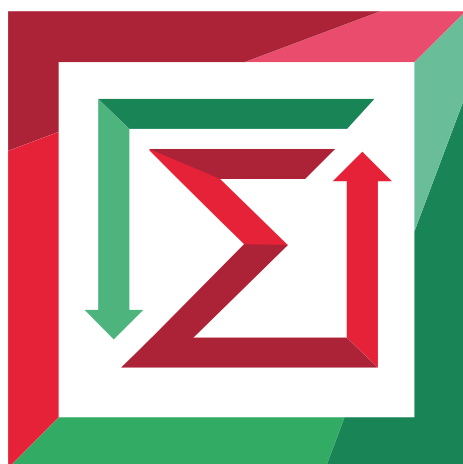
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



Теоретической и прикладной информатики

Лабораторная работа № 2
по дисциплине «Компьютерное моделирование»

ОПТИМИЗАЦИЯ РАБОТЫ СЕРВЕРА



Факультет:	ПМИ
Группа:	ПМИ-02
Студент:	Сидоров Даниил, Дюков Богдан
Преподаватель:	Карманов Виталий Сергеевич

Новосибирск

2026

1. Формулировка задания

Построить модель работы сервера и определить математическое ожидание времени и вероятности обработки запросов.

Сервер, имеющий входной буфер на 4 запроса, обрабатывает запросы, поступающие с рабочих станций с временными интервалами, распределенными по показательному закону со средним значением 3 мин/с. Время обработки сервером одного запроса распределено по экспоненциальному закону со средним значением 2 мин.

2. Цели работы

Построить имитационную модель работы сервера и оптимизировать ее параметры.

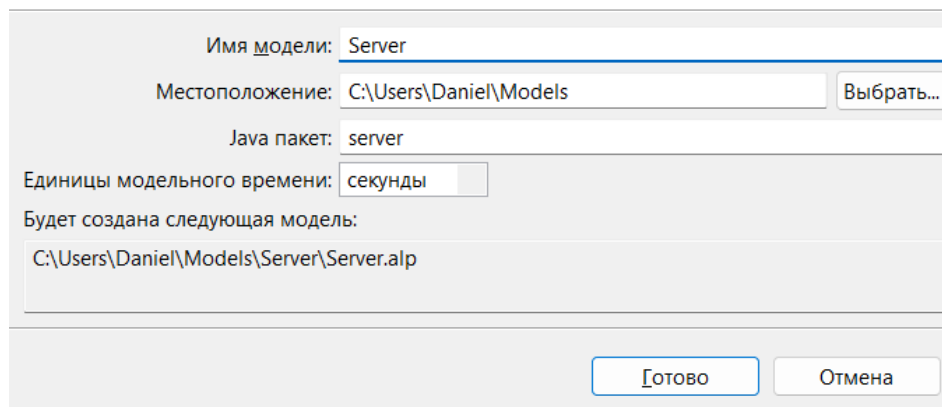
3. Описание выполненных действий

Создание модели

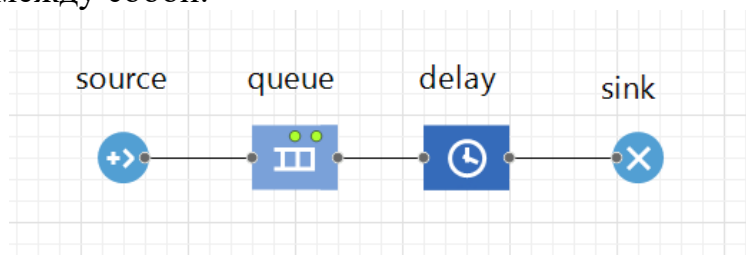
Зададим имя модели Server и установим единицы модельного времени – секунды. После создания модели откроется пользовательский интерфейс.

Новая модель

Создание новой модели



Добавим блоки библиотеки моделирования процессов на диаграмму и соединим их между собой.



Перейдем к установке свойств блоков модели и настройке модели. Изменим свойства объекта Source, учитывая, что время между прибытиями заявок определено в секундах согласно экспоненциальному закону:

source - Source

Имя: ☒ Отображать имя

☐ Исключить

Прибывают согласно:

Время между прибытиями:

Первое прибытие происходит:

Считать параметры агентов из БД: ☐

За 1 раз создается несколько агентов: ☐

Ограниченное кол-во прибытий: ☐

Определим свойства объекта Queue с учетом, что длина очереди (вместимость) равна пяти, и включим сбор статистики.

queue - Queue

Имя: ☒ Отображать имя

☐ Исключить

Вместимость:

Максимальная вместимость: ☐

Место агентов:

Специфические

Очередь:

Разрешить уход по таймауту: ☐

Разрешить вытеснение: ☐

Вернуть агента в исходную точку: ☒

Включить сбор статистики: ☒

Изменим свойства объекта Delay, считая, что время задержки определено согласно экспоненциальному закону и включим сбор статистики:

delay - Delay

Имя: ☒ Отображать имя ☐ Исключить

Тип задержки: ☒ Определенное время ☐ До вызова функции stopDelay()

Время задержки:

Вместимость:

Максимальная вместимость: ☐

Место агентов:

Специфические

Выталкивать агентов: ☐

Вернуть агента в исходную точку: ☒

Включить сбор статистики: ☒

Изменим свойства Simulation:Main, установив следующие параметры:

- режим выполнения: виртуальное время (максимальная скорость);
- остановка: в заданное время;

- конечное время: 3600 с;
- генератор случайных чисел: фиксированное начальное число (воспроизводимые прогоны), равное 4.

Simulation - Простой эксперимент

Имя: ☐ Исключить

Агент верхнего уровня:

Максимальный размер памяти: Мб

☒ Пропустить экран эксперимента и запустить модель

Параметры

Модельное время

Режим выполнения: ☒ Виртуальное время (максимальная скорость)
☐ Реальное время со скоростью

Остановить:

Начальное время: Конечное время:

Начальная дата: Конечная дата:

Случайность

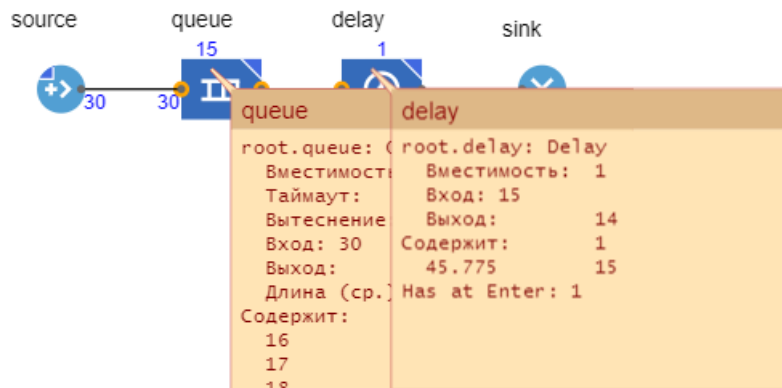
Генератор случайных чисел:

☐ Случайное начальное число (уникальные "прогоны")

☒ Фиксированное начальное число (воспроизводимые "прогоны") Начальное число:

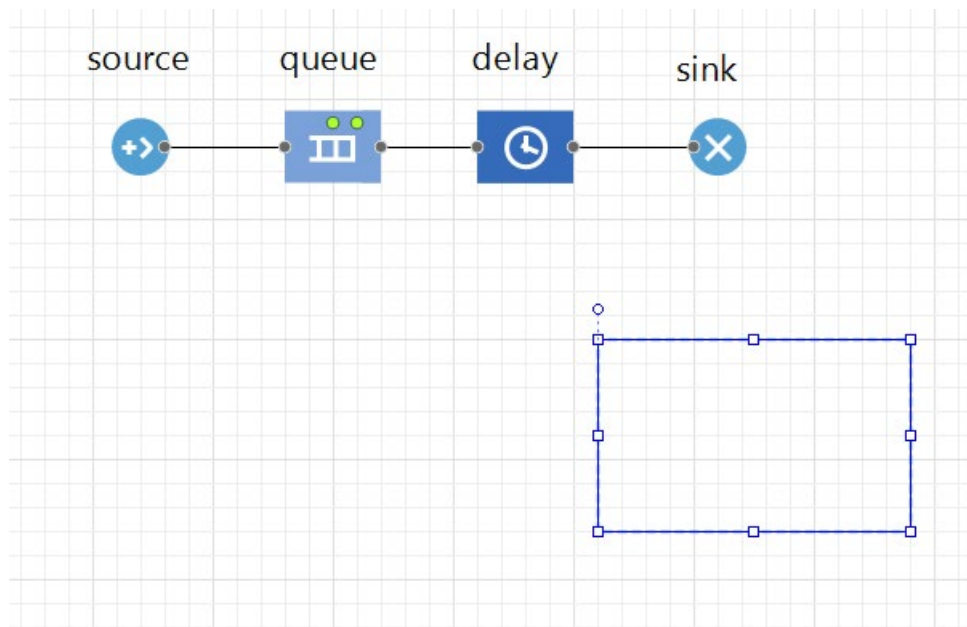
☐ Нестандартный генератор (подкласс класса Random):

Окно свойств объектов имеет такой вид:



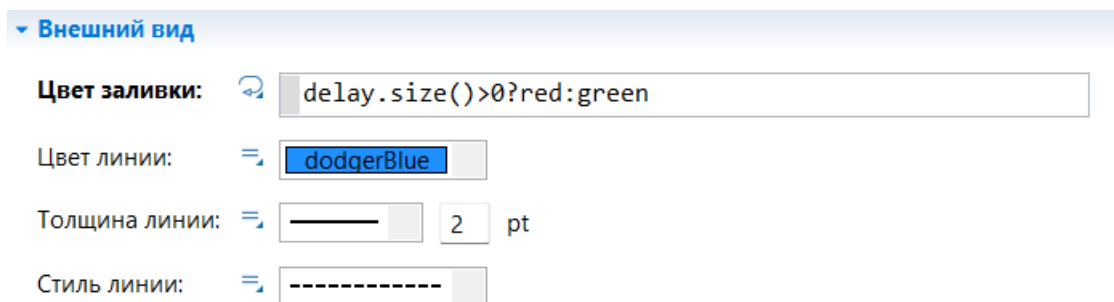
Создание анимации модели

Выберем прямоугольный узел и обозначим сервер.

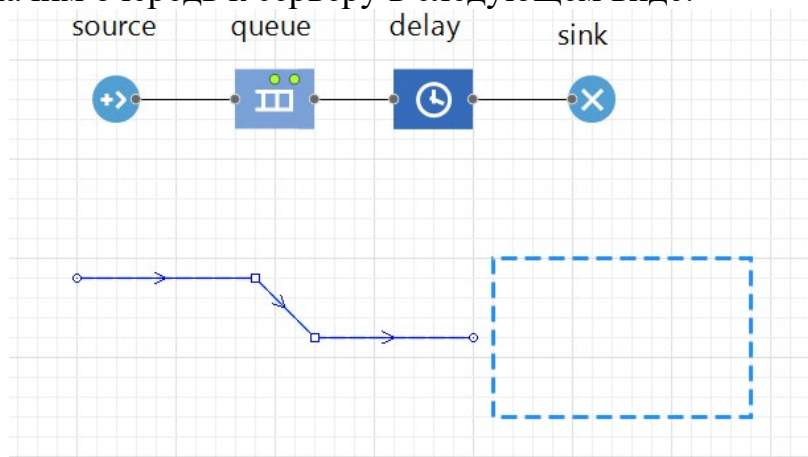


В свойствах прямоугольного узла установим динамический цвет заливки, который изменяется в зависимости от того, обрабатывает ли сервер в данный момент запрос или нет:

`delay.size()>0?red:green`



Обозначим очередь к серверу в следующем виде:



queue - Queue

Имя:
☒ Отображать имя
☐ Исключить

Вместимость:

Максимальная вместимость:

Место агентов:

delay - Delay

Имя:
☒ Отображать имя
☐ Исключить

Тип задержки:
☒ Определенное время
☐ До вызова функции stopDelay()

Время задержки:

Вместимость:

Максимальная вместимость:

Место агентов:

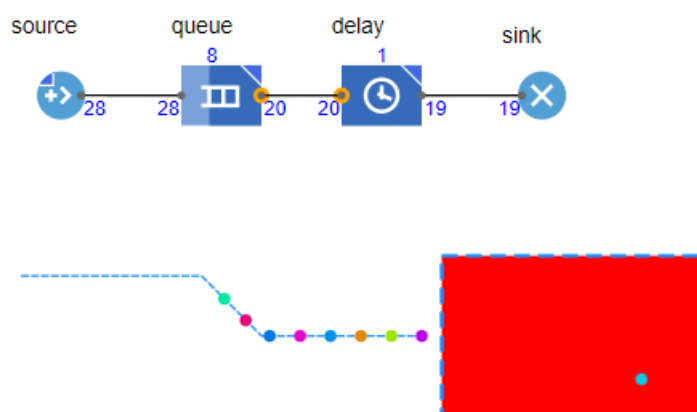
Специфические

Выталкивать агентов:

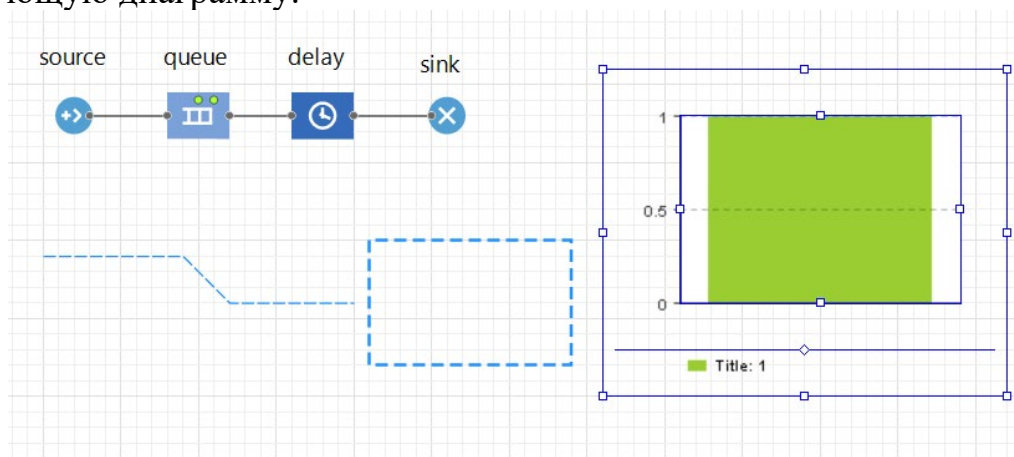
Вернуть агента в исходную точку:

Включить сбор статистики:

Запустим модель:



Для отображения среднего коэффициента использования сервера добавим следующую диаграмму.



Установим свойства диаграммы.

Данные

Заголовок: Server Utilization

Цвет: yellowGreen

Значение: delay.statsUtilization.mean()

Укажем параметры диаграммы:

Местоположение и размер

Уровень: level

X: 590

Ширина: 200

Y: 40






Высота: 360

Легенда

☒ Отображать легенду

Высота: 30

Цвет текста: black

Расположение:     

Область диаграммы

Смещение по оси X: 50

Ширина: 120

Смещение по оси Y: 30






Высота: 270

Цвет фона: white

Цвет границы: black

и выберем внешний вид диаграммы:

Внешний вид

Направление столбцов:     

Относительная ширина столбцов: 80%

Положение подписей у осей: Слева

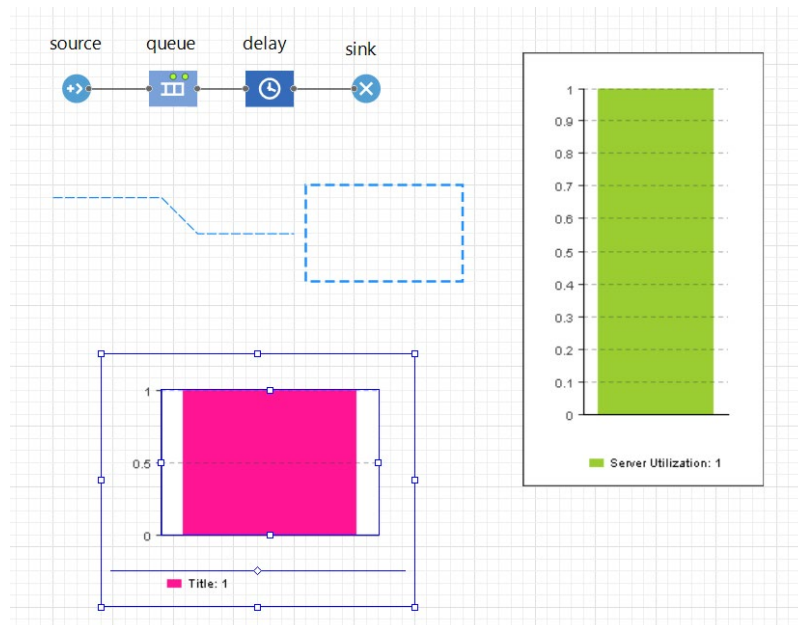
Цвет фона: white

Цвет границы: darkGray

Цвет меток: darkGray

Цвет сетки: darkGray

Добавим еще одну диаграмму для отображения средней длины очереди.



Также установим свойства диаграммы:

▼ Данные





Заголовок:

Цвет:

Значение:

и выберем внешний вид диаграммы:

▼ Внешний вид

Направление столбцов: ☐  ☒  ☐  ☐ 

Относительная ширина столбцов: 80%

Положение подписей у осей:

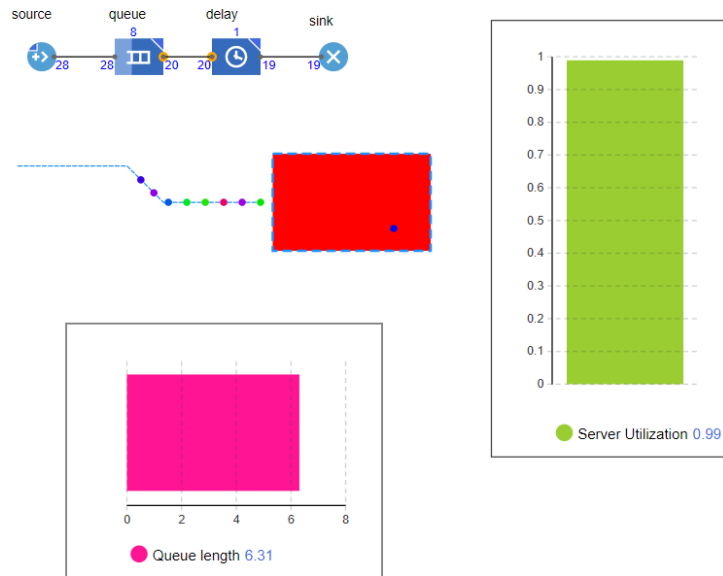
Цвет фона:

Цвет границы:

Цвет меток:

Цвет сетки:

Запустив модель, получим:



Все запросы, вырабатываемые объектом Source, имеют один и тот же приоритет, поэтому при полном заполнении накопителя (пять за- просов) теряться будет последний запрос. Уточним модель, изменив свойства объекта Source, и разрешим вытеснение.

queue - Queue

Имя: ☒ Отображать имя ☐ Исключить

Вместимость:

Максимальная вместимость:

Место агентов:

Специфические

Очередь:

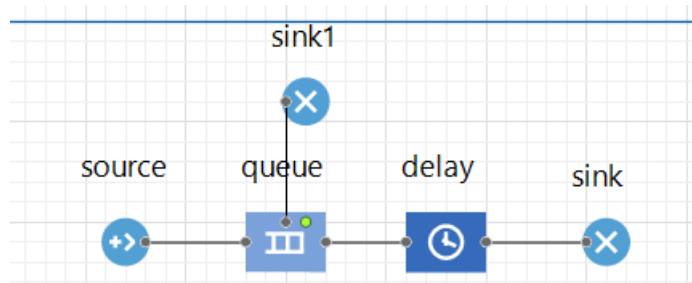
Разрешить уход по таймауту: ☐

Разрешить вытеснение: ☒

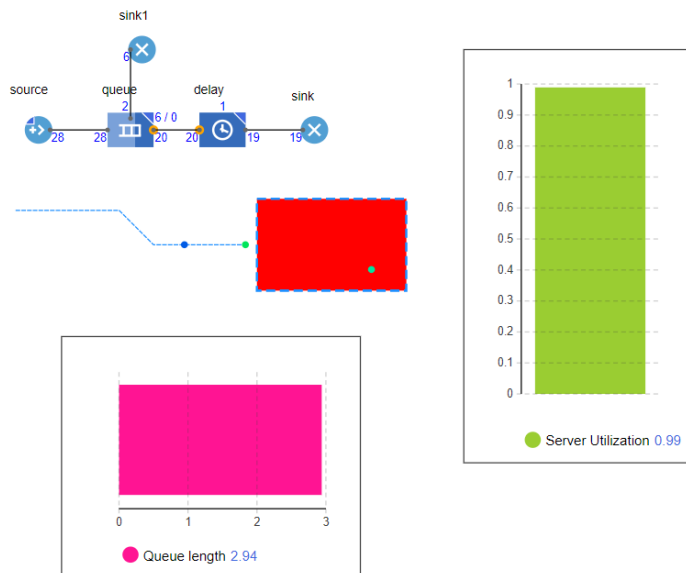
Вернуть агента в исходную точку: ☒

Включить сбор статистики: ☒

Для уничтожения потерянных запросов вследствие полного запол- нения накопителя нужно добавить второй объект Sink.



Соединим порт outPreempted объекта Queue с входным портом InPort блока Sink1.



Далее создадим тип заявок Inquiry. Для этого в панели «Проекты» правой кнопкой мыши выберем «Создать – Java-класс» с именем Inquiry и базовым классом Entity.

Новый Java класс

Java класс
Создание нового Java класса

Имя: Inquiry

Базовый класс: Entity

☒ Добавить возможность сохранения состояния

< Назад Далее > Готово Отмена

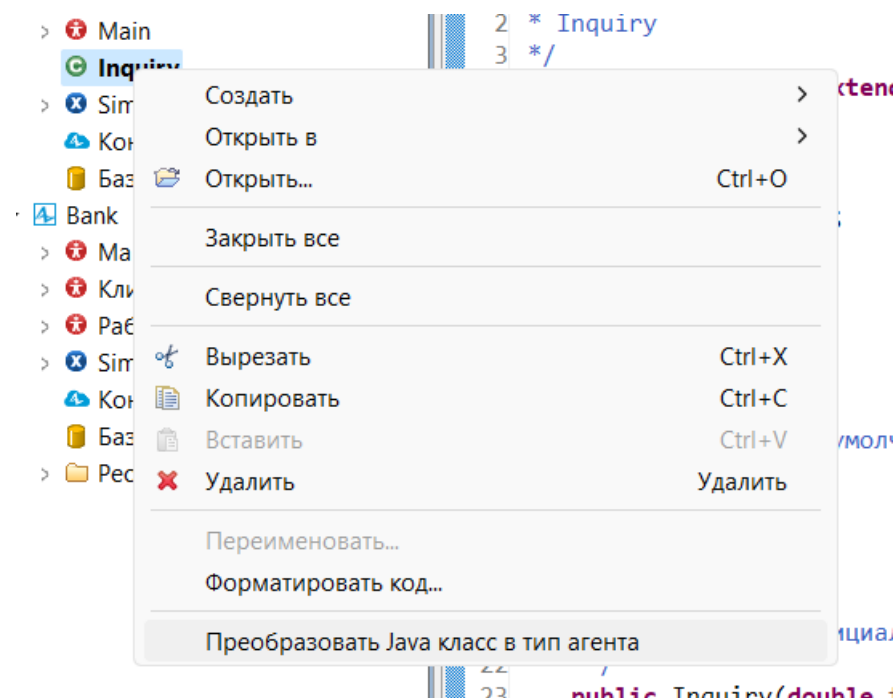
Поля класса
Добавьте поля Java класса

Имя	Тип	Доступ	Начальное зн...
time_vhod	double		
time_vihod	double		
col_vhod	int		
col_vihod	int		

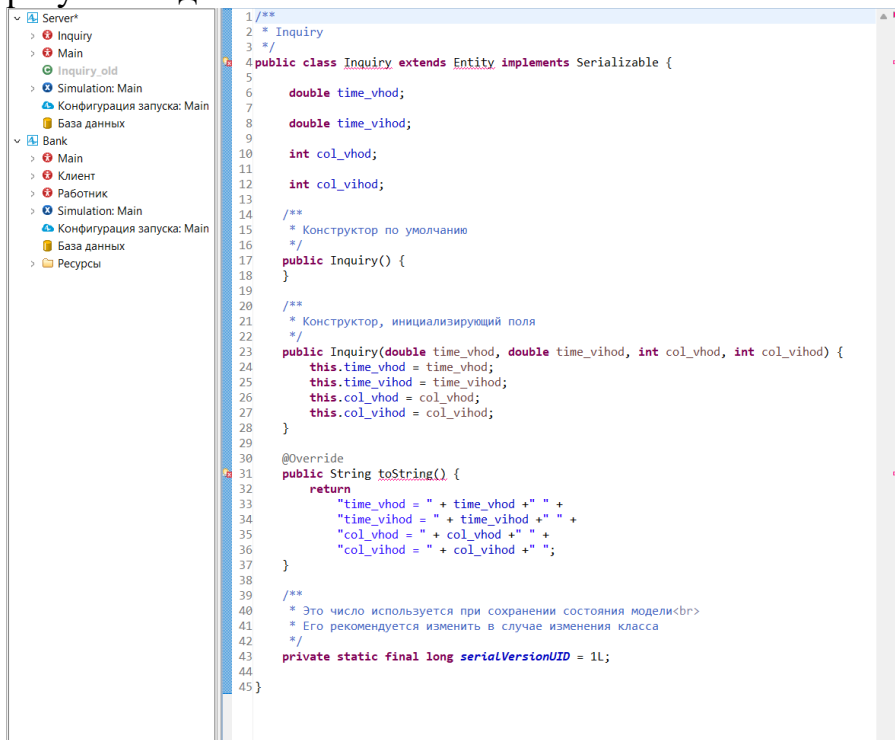
☒ Создать конструктор
☒ Создать функцию toString()

< Назад Далее > Готово Отмена

Преобразуем класс в тип агента следующим образом:



Окно с автоматически созданными параметрами нестандартного типа заявок Inquiry выглядит так:



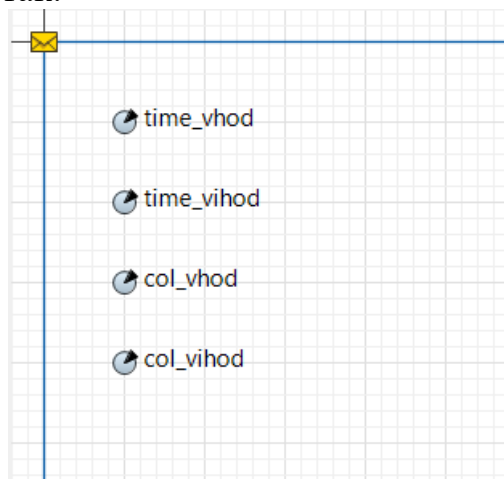
Приведем по шагам действия, необходимые для формирования нестандартного типа заявки:

- 1) создадим тип заявок Inquiry;
- 2) откроем палитру «Библиотека моделирования процессов»;
- 3) поместим элемент Agent в графический редактор;
- 4) в поле «Имя нового агента» введем Inquiry;
- 5) выберем анимацию агента, установив 2D, и выберем из выпадающего списка «Сообщение»;
- 6) выполним настройку параметров агента.

Параметр	Тип
time_vhod	double

time_vihod	double
col_vhod	int
Col_vihod	int

Окно с автоматически созданными параметрами нестандартного типа заявок Inquiry выглядит так:



Для сбора статистических данных о времени обработки запросов сервером необходимо добавить элемент статистики. Этот элемент будет запоминать значения времени для каждого запроса, предоставляя пользователю стандартную статистическую информацию (среднее, минимальное, максимальное из измеренных значений, среднеквадратичное отклонение и др.). Добавим элемент «Данные гистограммы», выбрав имя `time_obrabotki`, количество интервалов 50, начальный размер интервала 0,01.

time_obrabotki - Данные гистограммы

Имя: ☒ Отображать имя ☐ Исключить

Видимость: ☒ да

Значение:

Кол-во интервалов:

☒ Считать CDF

☐ Вычислять процентиля: Нижний: Верхний:

☒ Вести журнал в базе данных
[Вести журнал выполнения модели](#)

Диапазон значений

☒ Выбирается автоматически

☐ Фиксированный

Нач. размер интервала:

Добавим еще один элемент «Данные гистограммы» с именем `ver_obrabotki`, количеством интервалов 50 и начальным размером интервала 0,01.

ver_obrabotki - Данные гистограммы

Имя: ☒ Отображать имя ☐ Исключить

Видимость: ☒ да

Значение:

Кол-во интервалов:

☒ Считать CDF

☐ Вычислять процентили: Нижний: Верхний:

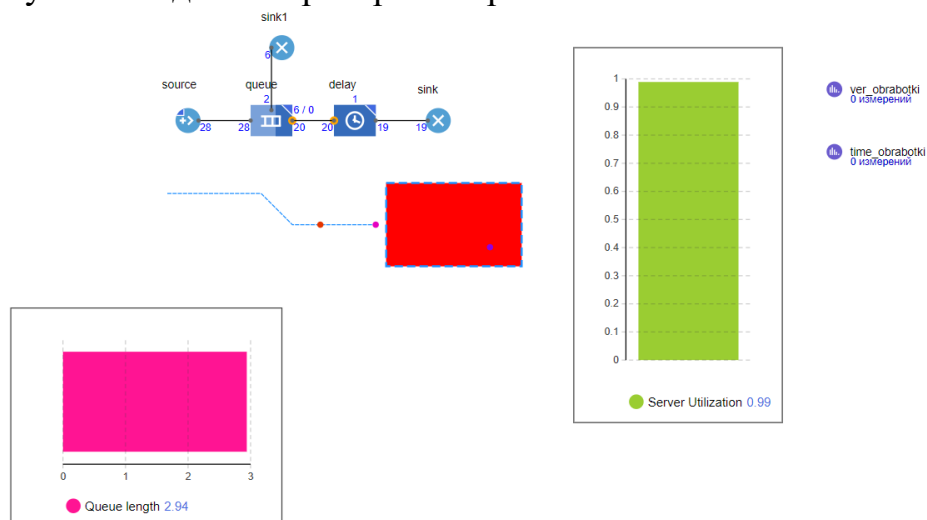
☒ Вести журнал в базе данных
[Вести журнал выполнения модели](#)

Диапазон значений

- ☒ Выбирается автоматически
- ☐ Фиксированный

Нач. размер интервала:

Запустим модель и проверим ее работоспособность:



Изменим в свойствах объекта Source тип заявки на Inquiry_.

▼ **Агент**

Новый агент:

Изменить размеры: ☐

▼ **Специфические**

Установить время начала: ☐

Добавить агентов в: ☒ Популяцию по умолчанию
☐ Другую популяцию агентов

Выталкивать агентов: ☒

▼ **Действия**

До прибытия:

При подходе к выходу:

При выходе:

▼ **Специфические**

Тип агента:

☒ Одиночный агент ☐ Популяция агентов

Модель/Библиотека: Библиотека моделирования процессов [\(Изменить...\)](#)

Видимость: ☒ да

☐ Отображается на верхнем агенте

☒ Вести журнал в базе данных
[Вести журнал выполнения модели](#)

Установим `agent.time_vhod=time()` в поле действие при выходе. Для остальных объектов поле «Тип агента» автоматически изменится на `Inquiry_`, поэтому ниже приведем пример объекта `Sink`.

▼ **Специфические**

Тип агента:

☒ Одиночный агент ☐ Популяция агентов

Модель/Библиотека: Библиотека моделирования процессов [\(Изменить...\)](#)

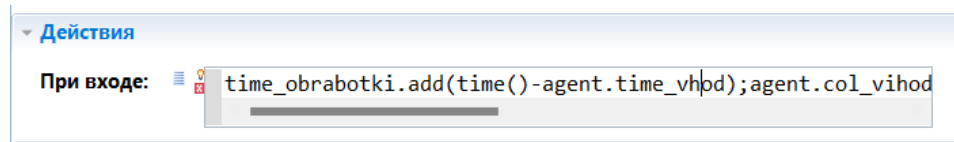
Видимость: ☒ да

☐ Отображается на верхнем агенте

☒ Вести журнал в базе данных
[Вести журнал выполнения модели](#)

Установим следующие действия при входе в объект `Sink`:

- `time_obrabotki.add(time() - agent.time_vhod)` – добавляется время обработки одного запроса в объект сбора данных гистограммы `time_obrabotki`. Данное время определяется как разность между текущим модельным временем `time()` и временем входа запроса в модель;
- `agent.col_vihod=sink.count(); agent.col_vhod=source.count()` – определяется количество запросов, вошедших в блок `Sink` и вышедших из блока `Source` соответственно;
- `ver_obrabotki.add(agent.col_vihod/agent.col_vhod)` – добавляется относительная доля обработанных запросов в объект сбора данных гистограммы `ver_obrabotki` при поступлении каждого обработанного запроса в блок `Sink`. На основе множества относительных долей обработанных запросов определяется математическое ожидание вероятности обработки запросов сервером.



Изменим тип параметров на double.

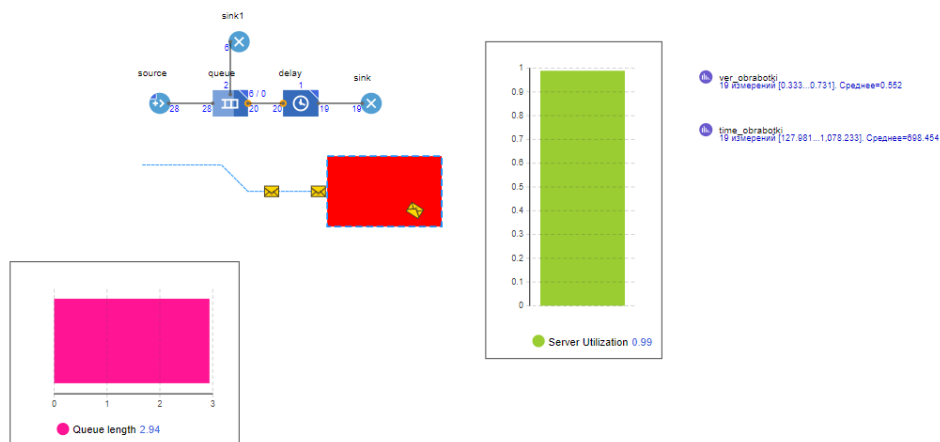
col_vihod - Параметр

Имя: ☒ Отображать имя ☐ Исключить
Видимость: ☒ да
Тип:
Значение по умолчанию:
☐ Массив системной динамики

col_vihod - Параметр

Имя: ☒ Отображать имя ☐ Исключить
Видимость: ☒ да
Тип:
Значение по умолчанию:
☐ Массив системной динамики

Запустим модель:



Создадим параметр объекта Delay со следующими свойствами:

- имя: time_mean (среднее время);
- значение по умолчанию: 120;
- тип: double.

time_mean - Параметр

Имя: ☒ Отображать имя ☐ Исключить

Видимость: ☒ да

Тип:

Значение по умолчанию:

☐ Массив системной динамики

Изменение свойств объекта Delay:

delay - Delay

Имя: ☒ Отображать имя ☐ Исключить

Тип задержки: ☒ **Определенное время**
☐ До вызова функции stopDelay()

Время задержки:

Вместимость:

Максимальная вместимость:

Место агентов:

Добавим бегунок, с помощью которого будем изменять среднее время обработки запросов объекта Delay.

slider - Бегунок

Имя: ☐ Исключить

☒ Отображается на верхнем агенте ☐ Блокировать

Ориентация: ☐ Вертикальная ☒ Горизонтальная

☒ Связать с:

Минимальное значение:

Максимальное значение:

Шаг:

Доступность: ☒ да

Добавим бегунок под элементом Queue, что позволит изменять емкость буфера в ходе моделирования.

slider1 - Бегунок

Имя: ☐ Исключить

☒ Отображается на верхнем агенте ☐ Блокировать

Ориентация: ☐ Вертикальная ☒ Горизонтальная

☒ Связать с:

Параметр:

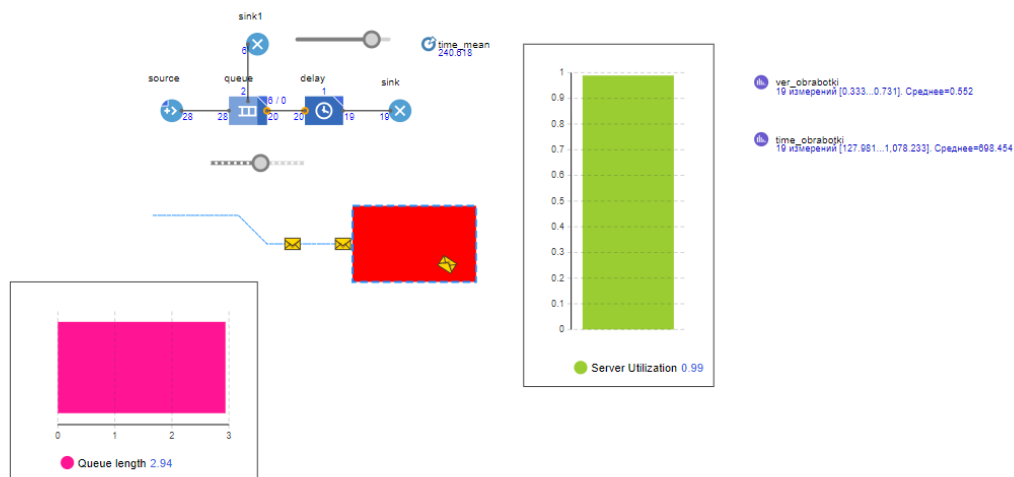
Минимальное значение:

Максимальное значение:

Шаг:

Доступность: ☒ да

Теперь в процессе моделирования можно изменять емкость входного буфера и среднее время обработки запросов с помощью бегунков. Запустим модель:



Добавим на диаграмму рассматриваемого потока гистограмму, отображающую собранную временную статистику.

Изменим свойства гистограммы:

- заголовок: Histogram Time obrabotki;
- данные: time_obrabotki;
- отображать среднее.

chart2 - Гистограмма

Имя: ☐ Исключить ☒ Отображается на верхнем агенте ☐ Блокировка

☒ Отображать плотность вер-ти ☐ Отображать ф-ю распределения ☐ Отображать среднее

Данные

Заголовок:

Данные:

Цвет плотности вер-ти: Цвет линии ф. распред.:

Толщина линии ф-ии распред. и среднего:

Цвет нижнего %: Цвет верхнего %:

Добавим еще одну гистограмму, отображающую собранную вероятностную статистику и изменим ее свойства:

- заголовок: Histogram Ver obrabotki;
- данные: ver_obrabotki;
- отображать среднее.

chart3 - Гистограмма

Имя: ☐ Исключить ☒ Отображается на верхнем агенте ☐ Блокировка

☒ Отображать плотность вер-ти ☐ Отображать ф-ю распределения ☐ Отображать среднее

▼ Данные

Заголовок:

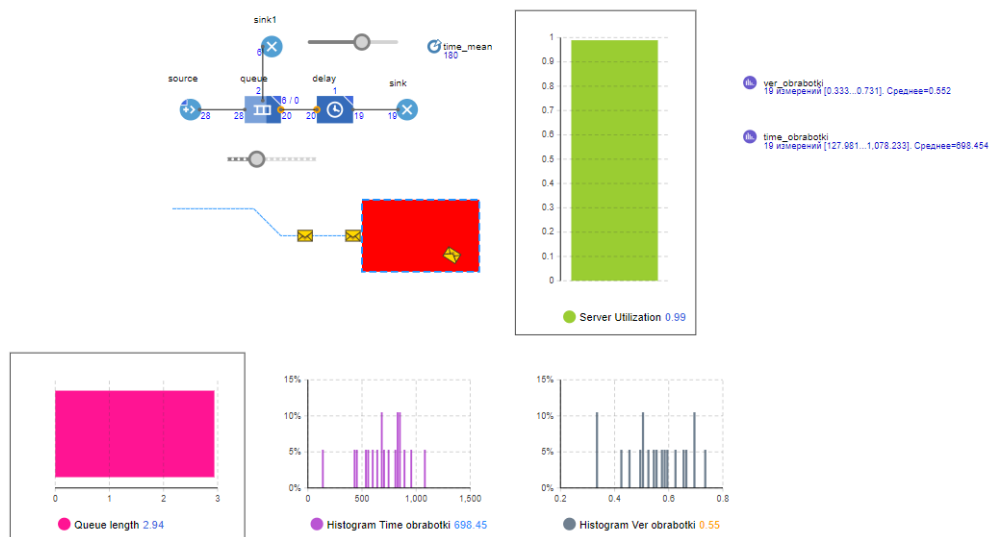
Данные:

Цвет плотности вер-ти: Цвет линии ф. распредел.:

Толщина линии ф-ии распредел. и среднего:

Цвет нижнего %: Цвет верхнего %:

Запустим окончательную модель:



4. Полученные результаты и их анализ

Количество запросов	Макс длина очереди	Среднее значение интервала времени между заявками и п, мин	Среднее время обслуживания заявки т, мин	Вероятность обслуживания	Мат ожидание времени обработки запросов, сек	Длина очереди, запросов	Загрузка сервера
4	5	3	1	0.92	60.35	0.09	0.2
4	5	3	2	0.92	140.73	0.28	0.37
4	5	3	3	0.88	221.82	0.46	0.56
10	5	3	2	0.9	179.34	0.45	0.74
4	5	2	2	0.92	118.08	0.32	0.51
4	5	1	2	0.35	662.42	4.05	0.99
4	1	3	2	0.08	125.61	0.17	0.37

Вероятность обслуживания 0.92, это означает, что почти все запросы обрабатываются. Если мы хотим увеличить этот показатель, то необходимо уменьшить время обработки запроса или обрабатывать запросы в многопоточном режиме. Если увеличить среднее время обслуживания заявки до 3 мин, это негативно скажется на работе сервера. Если уменьшить среднее значение интервала времени между заявками до 1 мин, то Сервер будет работать на пределе, будет постоянная очередь, высокое мат ожидание времени обработки запросов. Если увеличить число начальных запросов, это особо не скажется на работе сервера. При уменьшении макс. длины очереди количество необработанных заявок увеличивается.

5. Вывод

В работе построена имитационная модель работы сервера, позволяющая моделировать систему обслуживания с переменной длиной очереди и возможностью отказов. Определены вероятность и математическое ожидание времени обработки запросов