|  |  |  |
| --- | --- | --- |
| **КУРСОВОЙ ПРОЕКТ** | | |
| МОДЕЛИРОВАНИЕ СИСТЕМЫ, ФОРМАЛИЗОВАННОЙ КАК СИСТЕМА МАССОВОГО ОБСЛУЖИВАНИЯ | | |
| по дисциплине «Архитектура программных систем» | | |
| Выполнил | | |
| студент гр. 3530904/70104 |  | Григоренко С.А. |
|  | | |
| Руководитель |  | Дробинцев Д.Ф. |
| «13» декабря 2020г. | | |

Оглавление

[Введение 3](#_Toc26145098)

[Исходные данные 4](#_Toc26145099)

[Временная диаграмма 5](#_Toc26145100)

[Вывод законов распределения 6](#_Toc26145101)

[Ограничения и требуемые характеристики 6](#_Toc26145102)

[Обобщенная блок-схема 7](#_Toc26145103)

[Модульная структура 8](#_Toc26145104)

[Описание работы программы 9](#_Toc26145105)

[Результаты работы 11](#_Toc26145106)

[Анализ результатов: 11](#_Toc26145107)

[Вывод 11](#_Toc26145108)

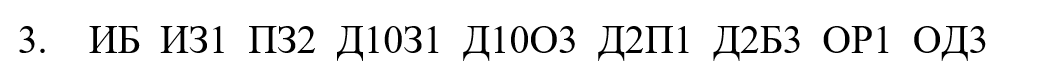
# Введение

Целью практической курсовой является создание модели ВС или ее компонентов на некотором уровне детализации, описывающей и имитирующей ее структуру и функциональность.

Каждый реальный объект ВС обладает огромной сложностью, определяемой множеством состояний, множеством внутренних и внешних связей, множеством анализируемых характеристик. Модель дает приближенное описание объекта с целью получения требуемых результатов с определенной точностью и достоверностью. Степень приближения модели к описываемому объекту может быть различной и зависит от требований задачи.

Существуют различные типы моделей ВС: аналитические, аналоговые, физические и имитационные. В данной работе будет использоваться имитационная модель ВС. Одним из подходов к построению имитационной модели является построение ее в виде системы массового обслуживания (СМО).

# Исходные данные



**1. Параметры элементов модели.**

**1.1 Источники:**

ИБ — бесконечный источник;

И31 — пуассоновский закон распределения заявок;

**1.2 Приборы:**

П32 — равномерный закон распределения времени обслуживания;

**2. Описание дисциплин постановки и выбора:**

**2.1. Дисциплина буферизации:**

Д10З1 — по кольцу;

**2.2. Дисциплина отказа:**

Д10О3 — самая старая в буфере;

**2.3. Дисциплина постановки на обслуживание:**

**2.3.1. Дисциплина выбора заявок на обслуживание.**

Д2Б5 — по кольцу;

**2.3.2. Дисциплина выбора прибора:**

Д2П1 — приоритет по номеру прибора;

**3. Виды отображения результатов работы программной модели:**

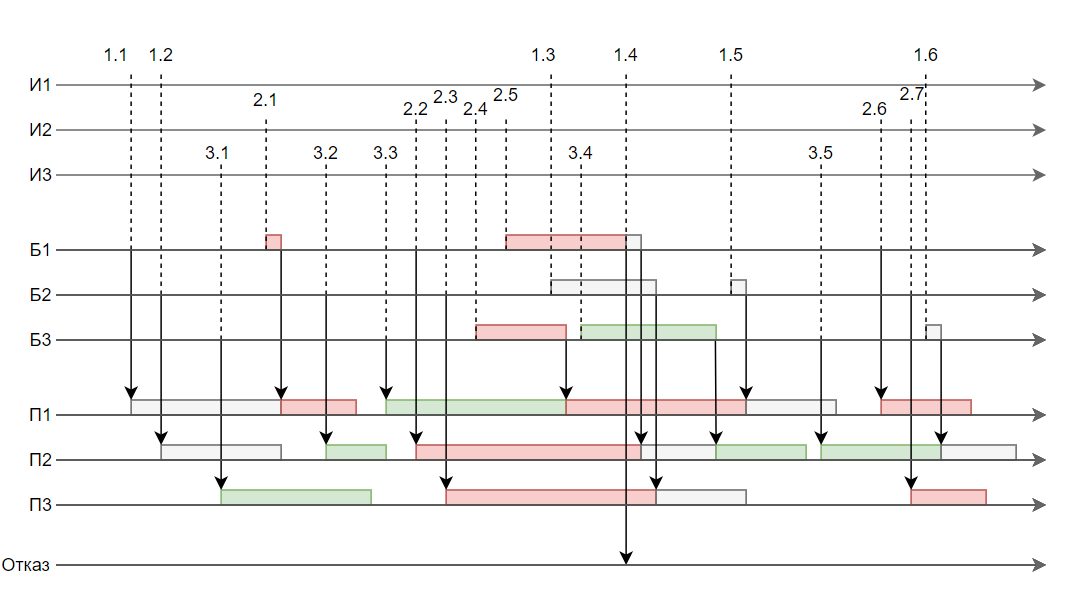
**3.1. Динамическое отражение результатов:**

ОД3 — временные диаграммы, текущее состояние;

**3.2. Отражение результатов после сбора статистики:**

ОР1 — сводная таблица результатов;

# Временная диаграмма



# Вывод законов распределения

Равномерный закон распределения:

Изображение выглядит как снимок экрана

Автоматически созданное описание

В программе данное выражение записано следующим образом:

**((float)qrand() / (float)RAND\_MAX) \* (beta - alpha) + alpha;**

Пуассоновский закон распределения

Изображение выглядит как снимок экрана

Автоматически созданное описание

В программе данное выражение записано следующим образом:

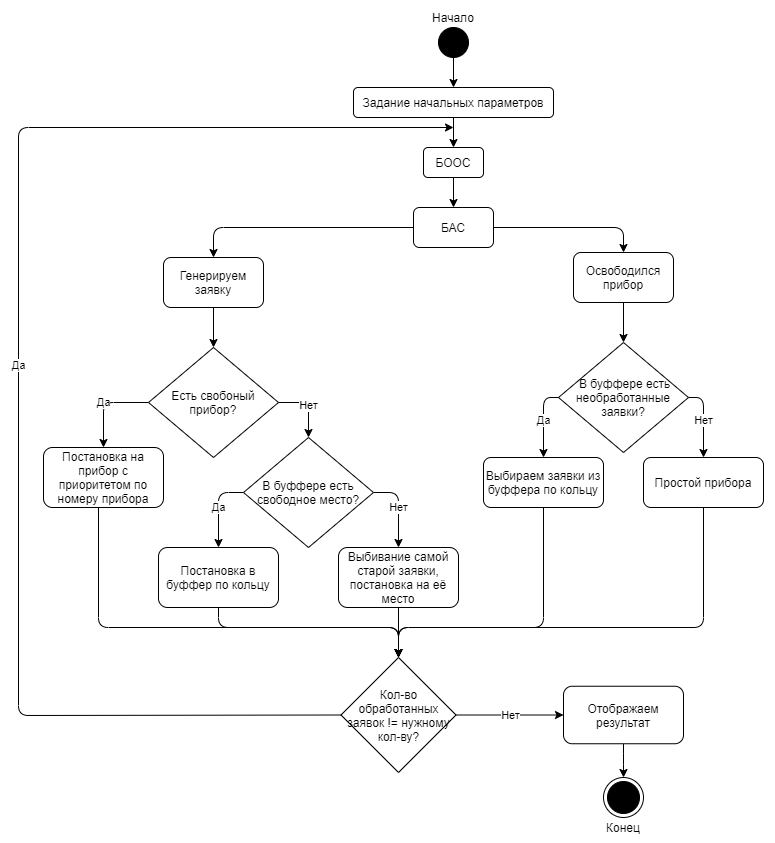
**(-1 / lambda) \* std::log(((float)qrand() / (float)RAND\_MAX));**

# Ограничения и требуемые характеристики

* Вероятность отказа должна составлять не более 10%.
* Загрузка приборов более 90%.
* Alpha: 1.0
* Beta: 1.1
* Lambda: 0.275

Время пребывания заявки в системе не ограничено, т.к. в зависимости от присланных данных, заявка может обрабатываться длительное время для получения верного результата.

# Обобщенная блок-схема



# Модульная структура

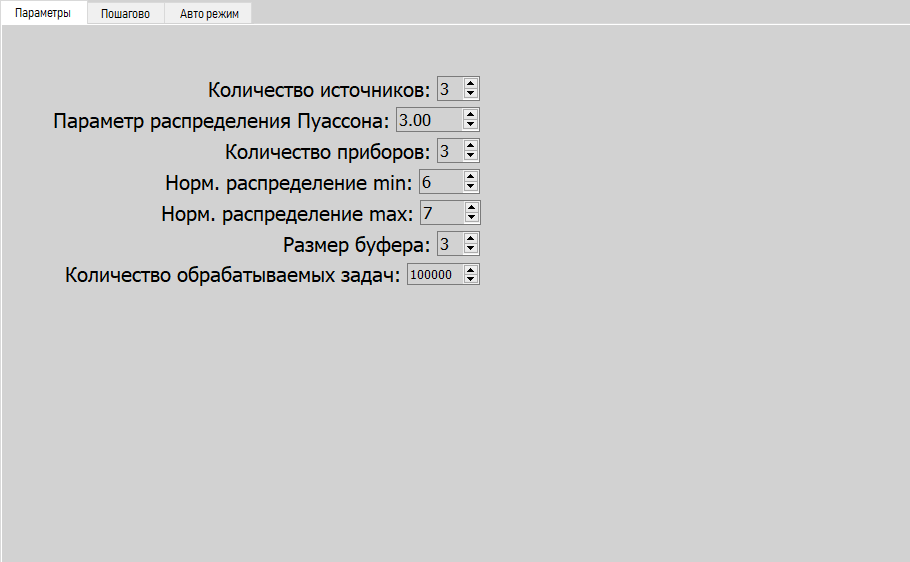
Разработка производилась в среде Qt Creator 5.13.2 на языке С++ с использованием графической библиотеки QT.

Приложение является объектно-ориентированным и содержит следующие классы:

* Request — содержит описание заявки. Реализует методы для получения этих описаний;
* Source — реализует методы создания заявки и генерации времени заявки;
* Buffer — реализует методы проверки свободного места в буфере, добавления заявки в буфер, выбора заявки из буфера;
* Device — реализует методы проверки свободных приборов, загрузки заявки на прибор, удаления заявки из прибора;
* Controller — реализует основной цикл работы системы;
* Analytics — реализует сбор аналитики для пошагового и автоматического отображения;
* Main Window — реализует графический интерфейс.

# Описание работы программы

При запуске программы появляется следующее окно:



Мы находимся во вкладке Configure. В ней необходимо задать параметры для СМО и нажать кнопку Configure для сохранения параметров. Переключаясь между вкладками «Automode» и «Stepmode» мы можем наблюдать результаты работы нашей системы.

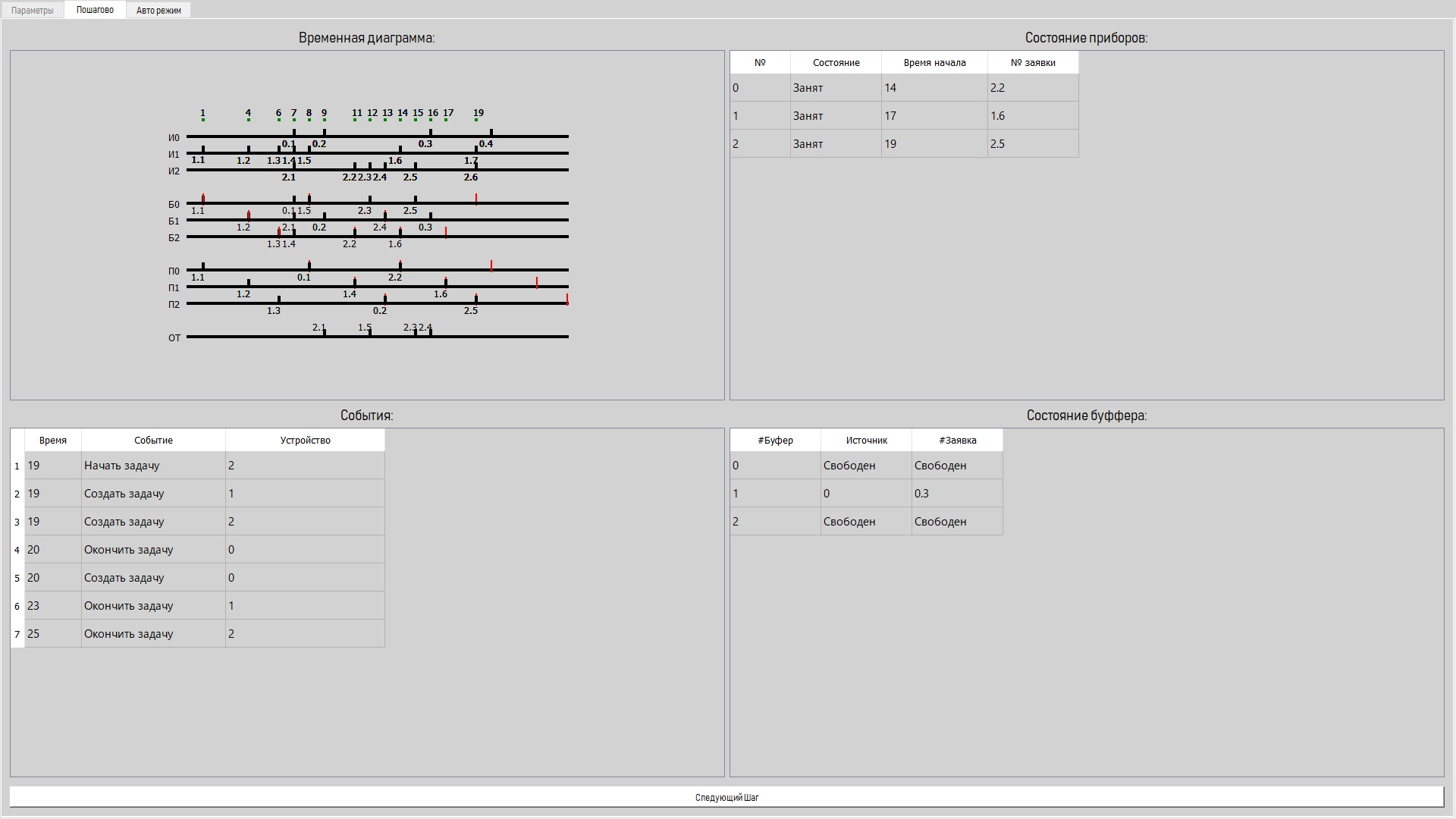
Пример отображения Automode:





При нажатии на кнопку Refresh у нас отображаетя последний результат работы.

Пример отображения Stepmode:



# Результаты работы

**Определение количества реализаций:**

Количество реализаций, необходимое для получения нужной точности при заданной доверительной вероятности, можно оценивать по формуле:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *t* 2 | (1  p) |  |  |
| N  |  |  | , |  |
|  | p 2 |  |
|  |  |  |  |

где p — вероятность отказа заявкам в обслуживании,

*t*= 1.643для=0.9,

=0.1 — относительная точность.

По результатам работы программы получено, что в большинстве случаев для достижения заданной точности необходимо от 2000 до 6000 заявок. Однако, в случаях, когда p мало (<0.05) для достижения точности в 10% может потребоваться существенно больше заявок (20000-30000).

# Анализ результатов:

Т. к. целью моделирования является выбор конфигурации системы, требующей наименьшее количество ресурсов и обрабатывающей максимальный поток информации, то начнем с проверки конфигурации с макс. числом источников и минимальным числом приборов и мин. размером буфера. Возьмем фиксированное количество источников равное 10.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Число источников | Число приборов | Размер буфера | Alpha | Beta | Lambda | Загруженность | P отк | T в сист |
| 10 | 5 | 5 | 4 | 5 | 4 | 0.9999 | 0.55 | 3.9 |

Из таблицы видно, что в последнем случае мы получили необходимую загруженность приборов, но вероятность отказов не удовлетворяет нашим условиям. Попробуем увеличить количество приборов.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Число источников | Число приборов | Размер буфера | Alpha | Beta | Lambda | Загруженность | P отк | T в сист |
| 10 | 10 | 5 | 4 | 5 | 4 | 0.9986 | 0.11 | 5.3 |

Видим, что вероятность отказов почти удовлетворяет нашим условиям, при этом загруженность все ещё на высоком уровне, попробуем добавить ещё один прибор.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Число источников | Число приборов | Размер буфера | Alpha | Beta | Lambda | Загруженность | P отк | T в сист |
| 10 | 11 | 5 | 4 | 5 | 4 | 0.9780 | 0.04 | 5.2 |

Этот случай удовлетворяет всем условиям, но не является самым дешевым. Попробуем заменить приборы на их более производительные версии.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Число источников | Число приборов | Размер буфера | Alpha | Beta | Lambda | Загруженность | P отк | T в сист |
| 10 | 7 | 5 | 2 | 4 | 4 | 0.9870 | 0.07 | 3.9 |

В этом случае вероятность отказа немного выше, но все ещё внутри заданных условий. С другой стороны мы существенно снизили время обработки одной заявки и сделали нашу систему немного дешевле.

# Вывод

В ходе курсовой работы была написана система массового обслуживания на языке C++ с использованием графической библиотеки Qt. С помощью данной программы была проанализирована реальная система и подобрана максимально выгодная комплектация данной системы.