**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Операционные системы»

**Лабораторная работа № 6-8**

Тема: Сервера сообщений, отложенные вычисления, интеграция программ друг с другом.

Студент: Гамов Павел

Группа: 80-207

Преподаватель: Миронов Е.С.

Дата:

Оценка:

Москва, 2019

1. **Постановка задачи**

Реализовать распределенную систему по асинхронной обработке запросов. В данной распределенной системе должно существовать 2 вида узлов: «управляющий» и «вычислительный». Необходимо объединить данные узлы в соответствии с той топологией, которая определена вариантом. Связь между узлами необходимо осуществить при помощи технологии очередей сообщений. Также в данной системе необходимо предусмотреть проверку доступности узлов в соответствии с вариантом.

Топология 2 – список списков вычислительных узлов.

Тип задания вычислительных узлов 4 – поиск подстроки в строке.

Команда проверки 2 – ping по id.

1. **Описание программы**

Был создан класс сообщений, где есть методы отправки и принятия различных сообщений. Класс работников содержит в себе параметры связи с сокетами, привязки к определенным портам, а также методы по перестройке или связи и отправке сообщений между собой. Основной файл main содержит в себе списки текущих процессов, и пути доступа к ним, при создании он отправляет работникам сообщения о создании и наше мини дерево перестраивается в соответствии с конфигурацией. Также тут в другом потоке находится коллектор, он отвечает за асинхронность выполнения задач, собирая данные после их завершения по команде update. Внутри каждого работника есть два нода, которые хранят данные о пути выше или ниже от себя, для пересылки им сообщений. Все данное действо написано используя 0mq – библиотеку для С++, в которой возможна реализация передачи данных по TCP протоколу.

**3.**           **Выводы**

Очень важно было добиться рационального использования сообщений, отправляя только то, что надо, я мог создать целую структуру для обработки и получения сообщений, но в виду архитектуры передачи текста нефиксированной длинны, размер которой может быть неограничен, мне пришлось разбивать сообщения по классам, в зависимости от их использования, что затрудняло отладку, так как в случае ошибки ты не сможешь понять что у тебя находится в сокете пока процесс не прочитает его и не обработает, часто в таком случае – неправильно, сломав всю программу. 0mq – как заверяют разработчики очень быстрый способ передачи сообщений, так оно в принципе и есть, но более того, я начинал писать лабораторную на Си, где функции слегка отличаются, но функционал тот же, как же это просто, особенно на С++, никаких заморочек, все запускается в 3 строчки, работает гладко, главное понимать, что ты делаешь.