## Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики Кафедра вычислительной математики и программирования

> Лабораторная работа № 6-8 по курсу «Операционные системы»

| Студент: Чирикова П. С.                 |
|---|
| Группа: М8О-201Б-21                     |
| Вариант: 11                             |
| Преподаватель Миронов Евгений Сергеевич |
| Оценка:                                 |
| Дата:                                   |
| Подпись:                                |
|   |

## Содержание

- 1. Репозиторий
- 2. Постановка задачи
- 3. Общие сведения о программе
- 4. Общий метод и алгоритм решения
- 5. Исходный код
- 6. Демонстрация работы программы
- 7. Выводы

## Постановка задачи

## Цель работы

Целью является приобретение практических навыков в:

- 1. Управлении серверами сообщений (№6)
- 2. Применение отложенных вычислений (№7)
- 3. Интеграция программных систем друг с другом (№8)

## Задание

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. При убийстве («kill -9») любого вычислительного узла система должна пытаться максимально сохранять свою работоспособность, а именно все дочерние узлы убитого узла могут стать недоступными, но родительские узлы должны сохранить свою работоспособность.

Управляющий узел отвечает за ввод команд от пользователя и отправку этих команд на вычислительные узлы. Список основных поддерживаемых команд:

#### Создание нового вычислительного узла

Формат команды: create id [parent]

id – целочисленный идентификатор нового вычислительного узла

рагепt – целочисленный идентификатор родительского узла. Если топологией не предусмотрено
введение данного параметра, то его необходимо игнорировать (если его ввели)

Формат вывода:

«Ok: pid», где pid – идентификатор процесса для созданного вычислительного узла

«Error: Already exists» - вычислительный узел с таким идентификатором уже существует

«Error: Parent not found» - нет такого родительского узла с таким идентификатором

«Error: Parent is unavailable» - родительский узел существует, но по каким-то причинам с ним не

удается связаться

«Error: [Custom error]» - любая другая обрабатываемая ошибка

Пример:

> create 10 5

Ok: 3128

Примечания: создание нового управляющего узла осуществляется пользователем программы при помощи запуска исполняемого файла. Id и pid — это разные идентификаторы.

#### Удаление существующего вычислительного узла

Формат команды: remove id

id – целочисленный идентификатор удаляемого вычислительного узла

Формат вывода:

«Ok» - успешное удаление

«Error: Not found» - вычислительный узел с таким идентификатором не найден

«Error: Node is unavailable» - по каким-то причинам не удается связаться с вычислительным узлом

«Error: [Custom error]» - любая другая обрабатываемая ошибка

Пример:

> remove 10

Ok

Примечание: при удалении узла из топологии его процесс должен быть завершен и работоспособность вычислительной сети не должна быть нарушена.

#### Исполнение команды на вычислительном узле

Формат команды: exec id [params]

id – целочисленный идентификатор вычислительного узла, на который отправляется команда

Формат вывода:

«Ok:id: [result]», где result – результат выполненной команды

«Error:id: Not found» - вычислительный узел с таким идентификатором не найден

«Error:id: Node is unavailable» - по каким-то причинам не удается связаться с вычислительным

узлом

«Error:id: [Custom error]» - любая другая обрабатываемая ошибка

Пример:

Можно найти в описании конкретной команды, определенной вариантом задания.

Примечание: выполнение команд должно быть асинхронным. Т.е. пока выполняется команда на одном из вычислительных узлов, то можно отправить следующую команду на другой

## Общие сведения о программе

Вариант 1, 4, 2 Вся бибиотека zmq.hpp

## Общие сведения о программе

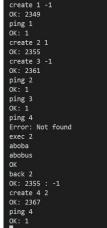
- dlopen динамически открывает нужную нам библиотеку.
- **dlsym** ищет нам нужную функцию

## Общий метод и алгоритм решения

Каждый узлы общаются только с соседями через очередь сообщений zmq. Структура – список с одним управляющим узлом. Каждый узел – отдельный процесс.

# Исходный код В репозитории

Демонстрация работы программы



## Выводы

Создание процессов в виде списка значительно ускоряет работу сервера, тк каждый узел – есть отдельный сервер. Zero MQ дает нам возможность при подключенном интернете создавать связи между родительским и дочерними процессами.