

**Московский авиационный институт
(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»
Дисциплина: «Операционные системы»

Курсовой проект
Тема: Совместный текстовый редактор

Студент: Алексеев Павел

Группа: 80-201

Преподаватель: Соколов А.А.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи

Написать программы, позволяющие совместно редактировать текстовый документ. Первая программа должна делать документ доступным в сети, вторая должна обеспечивать доступ к файлу.

2. Описание программы

Первая программа-сервер (server.cpp) обеспечивает доступ к указанному файлу. При запуске необходимо указать имя файла, порт для рассылки обновлений текста и порт для получения обновлений текста от пользователей: ./server [имя файла] [порт для отправки] [порт для получения]. После запуска сразу же считывается файл и сохраняется в памяти в виде вектора строк. Изменения применяются в виде номера строки и номера измененного символа. Файл сохраняется через 5 секунд после первого изменения.

Для того, что бы сервер мог в любой момент обработать подключение нового пользователя, запускается два потока:

- 1) Поток, в котором постоянно прослушивается порт для подключения (connecting_port_listening).
- 2) Поток, который перезапустит первый, после подключения пользователя (void* inf_listen_connect(void * args);).

Такие же два потока запускаются для прослушивания порта для обновлений текста (inf_listen_pull и pull_port_listener).

Когда приходит сообщение от пользователя оно помещается в очередь на обработку в потоке process_message. Есть два вида событий (Message.task) в сообщении:

- Отправлен измененный символ и его адрес в тексте (UPDATE_TEXT). В этом случае сначала применяются изменения к тексту, после чего сообщение отправляется остальным пользователям.
- Отправлен сигнал отключения пользователя (DISCONNECT). В этом случае пользователь удаляется из списка пользователей на сервере и счетчик пользователей обновляется у всех подключенных пользователей.

После обработки сообщение удаляется из очереди и начинается обработка следующего сообщения. Если сообщения закончились поток завершает свою работу и флаг его работы (pr_state) принимает значение false. При получении нового сообщения проверяется этот флаг. В случае, когда он имеет значение false, запускается поток process_message. В противном случае происходит только добавление сообщения в очередь.

Сервер имеет несколько внутренних команд, считываемых с клавиатуры:

- show — выводит весь текст, который был набран.
- Save — принудительно сохраняет текст в файл
- !/exit — завершение программы. После выполнения этой команды всем пользователям рассылается сообщение DISCONNECT, сохраняется в файл введенный текст и только потом программа завершается.

Вторая программа-клиент подключается к серверу по указанному адресу, который запрашивается у пользователя при запуске. После запуска программа пытается подключиться к серверу по указанному адресу. Если подключиться не получилось, выводится соответствующее сообщение и программа завершается. Если же подключиться удалось, программа получает от сервера номера портов для отправки и получения изменений текста. Затем программа-клиент получает текст от сервера и открывает текстовый редактор с полученным текстом.

Для получения и обработки изменений текста клиентская программа, так же как и сервер, запускает два потока: слушающий и перезапускающий. Для отправки изменений создается сообщение, в которое вносятся необходимые данные:

- where_x и where_y — где произошли изменения.
- change — измененный символ, если таковой имеется.
- wch — тип изменения (INSERT_CH, DELETE_CH).

- task — тип сообщения (UPDATE_TEXT).

После заполнения необходимых полей сообщение добавляется в очередь на отправку. Очередь на отправку обрабатывает специальный поток send_from_queue. Работает он по такому же принципу, как и принимающий сообщения поток на сервере, но только на отправку.

Текстовый редактор отображает количество строк, имя редактируемого файла и количество подключенных к серверу пользователей. Имеется две обрабатываемые комбинации клавиш:

- ctrl+o — сохранение копии текстового файла на данной машине. Файл сохраняется с именем файла и прибавленным к нему окончанием #copy.
- ctrl+x — выход из программы. Перед выходом проверяется, все ли сообщения из очереди были отправлены. Если же остались сообщения, пользователю будет выведено сообщение об этом и количество оставшихся сообщений для отправки. Как только все сообщения будут отправлены, программа отправит серверу сообщение DISCONNECT и завершится.

3. Набор тестов

Для серверной программы

№	Тип теста	Команды
1	Проверка существования файла	./server t.txt 4040 7523
2	Проверка правильности параметров запуска	./server text.txt 2002
3	Обычный запуск	./server text.txt 4040 7523
4	Проверка вывода всего текста	show

Для программы клиента

№	Тип теста	Команды
1	Проверка доступности сервера по указанному адресу	./client test test n
2	Обычный запуск	./client localhost localhost n

Так как программа-клиент имеет пользовательский интерфейс, некоторые способности программы будут приведены в пункте демонстрация работы программы.

4. Результаты выполнения тестов

Для серверной программы

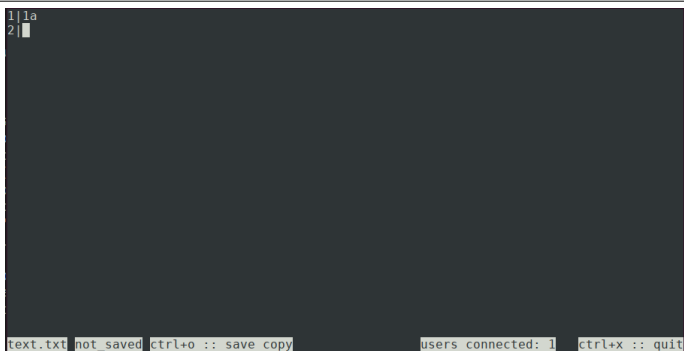
№	Результат
1	pavel@mahina:~/Documents/labki/3sem/osi/kursach\$./server t.txt 4040 7523 Can't open file!
2	pavel@mahina:~/Documents/labki/3sem/osi/kursach\$./server text.txt 2002 Wrong using! ./server /path/to/file port_publicher port_pull
3	pavel@mahina:~/Documents/labki/3sem/osi/kursach\$./server text.txt 4040 7523

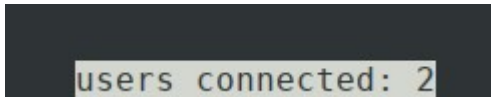
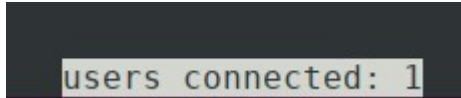
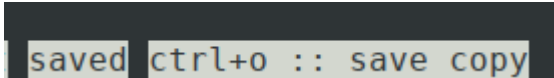
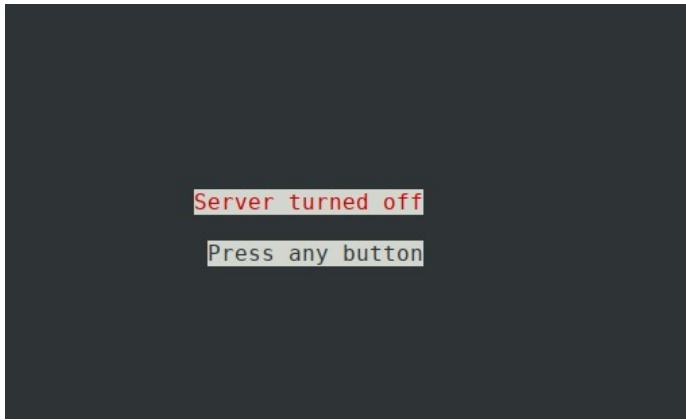

	Loaded Loaded file :: text.txt Port publisher :: 4040 Port pull :: 7523 Connecting port start listening Pull port start listening
4	show 1a

Для программы-клиента

№	Результаты
1	pavel@mahina:~/Documents/labki/3sem/osi/kursach\$./client Server address :: l You ip :: l Turn safe mode?(low speed)y/n :: n Safe mode off Can't connect to server
2	pavel@mahina:~/Documents/labki/3sem/osi/kursach\$./client Server address :: localhost You ip :: localhost Turn safe mode?(low speed)y/n :: n Safe mode off pub=4040::push=7523 Connected Loading file text.txt Done (далее запускается UI)

5. Демонстрация функций и способностей программы

Описание действия	Результат у сервера	Результат у клиента
Подключение	New user connected 0::localhost	Запуск UI
Ввод двух символов и переноса строки	localhost::1 Text Update 1 ----- localhost::1 Text Update 1a ----- localhost::1 Text Update 1a ----- ---saved---	

Подключение второго пользователя	New user connected 0::localhost 1::localhost	
Отключение второго пользователя	localhost::-1	
Сохранение файла у пользователя	Отсутствуют	 <p>Нажата комбинация ctrl+o pavel@mahina:~/Documents/labki/3sem/osi/ kursach\$ ls text.txt#copy text.txt</p>
Отключение сервера	!/exit ---saved--- pavel@mahina:~/Documents/labki/3sem/osi/kursach\$	
Отправка всех сообщений при выходе у клиента	Отсутствуют	

5. Листинг

Все исходные файлы находятся в репозитории на github:

<https://github.com/poeblo/DocLiveEditor>

6. Вывод

В данной курсовой работе я применил все знания, набранные за семестр: работа с потоками, файлами и сокетами. Для создания UI использовалась библиотека ncurses, которая была самостоятельно изучена. Благодаря данной курсовой работе я закрепил все набранные за семестр знания.