

Лабораторная работа 14

Именованные каналы

Арина Олеговна Аристова

2022, 3 June

RUDN University, Moscow, Russian Federation

Приобретение практических навыков работы с именованными каналами.

Изучите приведённые в тексте программы `server.c` и `client.c`.
Взяв данные примеры за образец, напишите аналогичные программы, внося следующие изменения:

1. Работает не 1 клиент, а несколько (например, два).
2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд).
Используйте функцию `sleep()` для приостановки работы клиента.
3. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек). Используйте функцию `clock()` для определения времени работы сервера. Что будет в случае, если сервер завершит работу, не закрыв канал?

Одним из видов взаимодействия между процессами в операционных системах является обмен сообщениями. Под сообщением понимается последовательность байтов, передаваемая от одного процесса другому.

В операционных системах типа UNIX есть 3 вида межпроцессорных взаимодействий: общеюниксные (именованные каналы, сигналы), System V Interface Definition (SVID — разделяемая память, очередь сообщений, семафоры) и BSD (сокеты).

Для передачи данных между неродственными процессами можно использовать механизм именованных каналов (named pipes). Данные передаются по принципу FIFO (First In First Out) (первым записан — первым прочитан), поэтому они называются также FIFO pipes или просто FIFO. Именованные каналы отличаются от неименованных наличием идентификатора канала, который представлен как специальный файл (соответственно имя именованного канала — это имя файла). Поскольку файл находится на локальной файловой системе, данное IPC используется внутри одной системы.

Создаю поддиректорию `~/work/os/lab14`, в ней создаю и заполняю файлы `common.h`, `service.c`, `client.c`, `client2.c`:

```
[aoaristova@fedora lab14]$ vi common.h
[aoaristova@fedora lab14]$ vi server.c
[aoaristova@fedora lab14]$ vi client.c
[aoaristova@fedora lab14]$ vi client2.c
[aoaristova@fedora lab14]$
```

Рис. 1: Создание и заполнение файлов `common.h`, `service.c`, `client.c`, `client2.c`.

Содержимое заголовочного файла `common.h` заполняю согласно описанию лабораторной работы:

```
/*
 * common.h - заголовочный файл со стандартными определениями
 */

#ifndef __COMMON_H__
#define __COMMON_H__

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

#define FIFO_NAME "/tmp/fifo"
#define MAX_BUFF 80
#endif /* __COMMON_H__ */
```

Рис. 2: Содержимое заголовочного файла.

Заполняю файл `server.c`, реализующий сервер. Использую функцию `clock` для определения времени работы сервера. Сервер работает не бесконечно, а прекращает работу через некоторое время, в моем случае, через 30 секунд:

Выполнение лабораторной работы

```
/*
 * server.c - реализация сервера
 *
 * чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли;
 * 2. запустить программу client на другой консоли.
 */

#include "common.h"
int
main()
{
    int readfd; /* дескриптор для чтения из FIFO */
    int n;
    char buff[MAX_BUFF]; /* буфер для чтения данных из FIFO */
    /* баннер */
    printf("FIFO Server...\n");

    /* создаем файл FIFO с открытыми для всех
     * правами доступа на чтение и запись
     */
    if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
    {
        fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-1);
    }
    /* откроем FIFO на чтение */
    if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
    {
        fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-2);
    }
    /* использую функцию clock для определения времени работы сервера */
    clock_t now=time(NULL), start=time(NULL);
```

Рис. 3: Содержимое файла реализации сервера, начало.

Выполнение лабораторной работы

```
clock_t now=time(NULL), start=time(NULL);
while(now-start<30)
{
    /* читаем данные из FIFO и выводим на экран */
    while((n = read(readfd, buff, MAX_BUFF)) > 0)
    {
        if(write(1, buff, n) !=n)
        {
            fprintf(stderr, "%s: Ошибка вывода" (%s)\n",
                __FILE__, strerror(errno));
        }
    }
    now=time(NULL);
}
printf("server timeout, %li - second passed\n", (now-start));
close(readfd); /* закроем FIFO */
/* удалим FIFO из системы */
if(unlink(FIFO_NAME) < 0)
{
    fprintf(stderr, "%s: PkPuPiPsP·PjPsPMPSPs CPrPrP"PsPec,сь FIFO (%s)\n",
        __FILE__, strerror(errno));
    exit(-4);
}
exit(0);
0
```

Рис. 4: Содержимое файла реализации сервера, конец.

Заполняю файлы `client.c` и `client2.c`. Использую функцию `sleep` для приостановки работы клиента. Клиенты передают текущее время с некоторой периодичностью, в моем случае раз в пять секунд:

Выполнение лабораторной работы

```
/*
 * client.c - реализация клиента
 *
 * чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли;
 * 2. запустить программу client на другой консоли.
 */

#include "common.h"
#define MESSAGE "Hello Server!!!\n"
int
main()
{
    int msg, len, i; /* дескриптор для записи в FIFO */
    long int t;
    for(i=0; i<20; i++)
    {
        /* используем функцию sleep для приостановки работы клиента */
        sleep(3);
        t = time(NULL);
        printf("FIFO Client...\n");

        /* получим доступ к FIFO */
        if((msg = open(FIFO_NAME, O_WRONLY))<0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }
        /* передадим сообщение серверу */
        len = strlen(MESSAGE);

        if(write(msg, MESSAGE, len) != len)
        {
            fprintf(stderr, "%s: Ошибка в записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }
        /* закроем доступ к FIFO */
        close(msg);
    }
    exit(0);
}
```

Рис. 5: Содержимое файла реализации клиента 1.

Выполнение лабораторной работы

```
/*
 * client.c - реализация клиента
 *
 * чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли;
 * 2. запустить программу client на другой консоли.
 */

#include "common.h"
#define MESSAGE "Hello Server!!!\n"
int
main()
{
    int msg, len, l; /* дескриптор для записи в FIFO */
    long long int t;
    char message[10];
    for(count = 0; count<=5; ++count)
    {
        /* используем функцию sleep для приостановки работы клиента */

        sleep(5);
        t = (long long int) time(0);
        sprintf(message, "%lli", t);
        /* получим доступ к FIFO */
        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }

        /* передадим сообщение серверу */
        msglen = strlen(MESSAGE);
        if(write(writefd, MESSAGE, msglen) != msglen)
        {
            fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }

        /* закроем доступ к FIFO */
        close(writefd);
        exit(0);
    }
}
```

Рис. 6: Содержимое файла реализации клиента 2.

Создаю и заполняю Makefile:

```
[aoaristova@fedora lab14]$ vi Makefile  
[aoaristova@fedora lab14]$ make  
gcc server.c -o server
```

Рис. 7: Создание и заполнение Makefile

```
all: server client

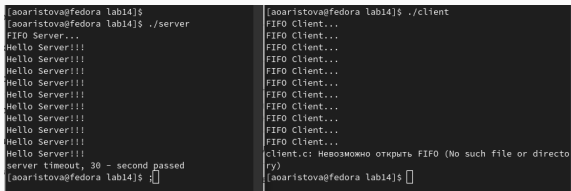
server: server.c common.h
    gcc server.c -o server

client: client.c common.h
    gcc client.c -o client

clean:
    -rm server client *.o
```

Рис. 8: Содержание Makefile.

Затем выполняю программу. На одной консоли запускаю программу server, а на другой консоли запускаю программу client.



```
[aoaristova@fedora lab14]$  
[aoaristova@fedora lab14]$ ./server  
FIFO Server...  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
Hello Server!!!  
server timeout, 30 - second passed  
[aoaristova@fedora lab14]$  
[aoaristova@fedora lab14]$ ./client  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
FIFO Client...  
client.c: Невозможно открыть FIFO (No such file or directory)  
[aoaristova@fedora lab14]$
```

Рис. 9: Запуск программ на двух консолях.

В ходе выполнения лабораторной работы я приобрела практические навыки в работы с именованными каналами.