**北京邮电大学软件学院**

**2016-2017学年第2学期实验报告**

**课程名称： 操作系统**

**实验名称： 进程通讯**

**实验完成人：**

**姓名：刘浩博 学号：2015212086 成绩：**\_\_\_\_\_\_\_\_

**姓名：陈润泽 学号：2015212105 成绩：**\_\_\_\_\_\_\_\_

**指导教师：**\_\_\_\_\_\_\_ 陈晋鹏\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**日 期： 2017 年 4 月 16 日**

1. **实验目的**

（1）消息缓冲队列、共享存储区机制进行进程间的通信；

（2）理解通信机制。

1. **实验内容**

1.使用消息缓冲队列来实现 client 进程和 server 进程之间的通信

server 进程先建立一个关键字为 SVKEY（如 75）的消息队列，然后等待接收类型为 REQ （例如 1）的消息；在收到请求消息后，它便显示字符串“serving for client”和接收到 的 client 进程的进程标识数，表示正在为 client 进程服务；然后再向 client 进程发送应 答消息，该消息的类型是 client 进程的进程标识数，而正文则是 server 进程自己的标识 ID。client 进程则向消息队列发送类型为 REQ 的消息（消息的正文为自己的进程标识 ID） 以取得 sever 进程的服务， 并等待 server 进程发来的应答；然后显示字符串“receive reply from”和接收到的 server 进程的标识 ID。

2. 使用共享存储区来实现两个进程之间的进程通信

进程 A 创建一个长度为 512 字节的共享内存，并显示写入该共享内存的数据；进程 B 将共享内存附加到自己的地址空间，并向共享内存中写入数据。

1. **实验环境**

在Linux和MacOS下，采用以VSCode +GCC+GDB为开发环境的C语言实验环境。其中VSCode 作为编辑器，GCC作为编译器，GDB作为调试器。

1. **实验过程**

分析了两个小实验的目标和相关函数的使用方法，然后进行分工：

刘浩博负责实验2以及实验报告的编写；

陈润泽负责实验1以及调试修改实验代码。

1. **实验结果**

实验均能达到预想目标，与实验指导手册给出的结果基本吻合。

1. **附件**

**6.1 源代码**

**Lab2.h:**

**#ifndef OSLABS\_LAB2\_H**

**#define OSLABS\_LAB2\_H**

**#define LAB2\_1 0**

**#define LAB2\_2 1**

**#include <sys/types.h>**

**#include <sys/msg.h>**

**#include <sys/ipc.h>**

**#include <sys/shm.h>**

**#include <unistd.h>**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**#define LAB2\_MSGKEY 166**

**#define LAB2\_SHMKEY 166**

**#define LAB2\_MTXT\_LEN 1024**

**#define LAB2\_SHMEM\_LEN 1024**

**struct msgform {**

**long mtype;**

**char mtext[LAB2\_MTXT\_LEN];**

**};**

**#endif //OSLABS\_LAB2\_H**

**Lab2Server.c:**

**#include "Lab2.h"**

**#ifdef \_POSIX\_VERSION**

**#if LAB2\_1**

**int main(int argc, char \*\*argv) {**

**int count = 0;**

**while (count++ < 100) {**

**int msgqid, pid, i;**

**msgqid = msgget(LAB2\_MSGKEY, 0777 | IPC\_CREAT); /\*创建 75#消息队列\*/**

**struct msgform rcvmsg;**

**msgrcv(msgqid, &rcvmsg, sizeof(struct msgform), 1, 0); /\*接收消息\*/**

**puts(rcvmsg.mtext);**

**}**

**exit(0);**

**}**

**#elif LAB2\_2**

**int main(int arc, char \*\*argv) {**

**int shmid = shmget(LAB2\_SHMKEY, LAB2\_SHMEM\_LEN, 0777 | IPC\_CREAT);**

**char \*scr = shmat(shmid, NULL, SHM\_R | SHM\_W);**

**int x = 0, y = 0;**

**char ch;**

**while ((ch = (char) getchar()) != EOF) {**

**system("clear");**

**switch (ch) {**

**case 'w':**

**y = y > 0 ? y - 1 : y;**

**printf("[Up]");**

**break;**

**case 's':**

**y = y < 15 ? y + 1 : y;**

**printf("[Down]");**

**break;**

**case 'a':**

**x = x > 0 ? x - 1 : x;**

**printf("[Left]");**

**break;**

**case 'd':**

**x = x < 63 ? x + 1 : x;**

**printf("[Right]");**

**break;**

**default:**

**break;**

**}**

**for (int i = 0; i < LAB2\_SHMEM\_LEN; ++i) {**

**if (64 \* y + x == i) {**

**scr[i] = '@';**

**}**

**else {**

**scr[i] = '%';**

**}**

**}**

**}**

**}**

**#endif**

**#endif**

**Lab2Client.c:**

**#include "Lab2.h"**

**#ifdef \_POSIX\_VERSION**

**#if LAB2\_1**

**int main(int argc, char \*\*argv) {**

**int msgqid, pid, i;**

**char str[LAB2\_MTXT\_LEN];**

**int count = 0;**

**while (count++ < 100) {**

**msgqid = msgget(LAB2\_MSGKEY, 0777);**

**pid = getpid();**

**struct msgform sndmsg;**

**sndmsg.mtype = 1;**

**scanf("%s", str);**

**if (strcmp(str, "quit") == 0) {**

**exit(1);**

**}**

**sprintf(sndmsg.mtext, "[Client][%d]%s", getpid(), str);**

**msgsnd(msgqid, &sndmsg, sizeof(struct msgform), 0);**

**printf("[Send][%d]%s\n", getpid(), str);**

**}**

**exit(0);**

**}**

**#elif LAB2\_2**

**int main(int argc, char \*\*argv) {**

**int shmid = shmget(LAB2\_SHMKEY, LAB2\_SHMEM\_LEN, 0777);**

**char \*scr = shmat(shmid, NULL, SHM\_R);**

**while (1) {**

**system("clear");**

**for (int i = 0; i < LAB2\_SHMEM\_LEN; ++i) {**

**putchar(scr[i]);**

**if (i % 64 == 63) putchar('\n');**

**}**

**sleep(1);**

**}**

**return 0;**

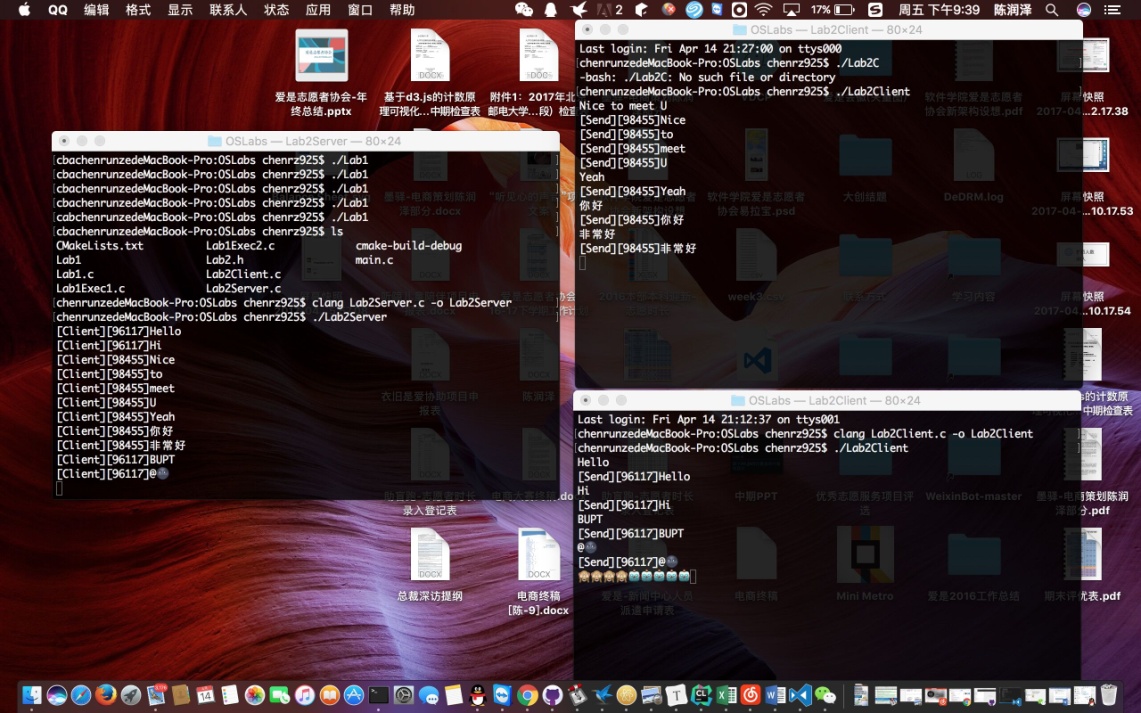
**}**

**#endif**

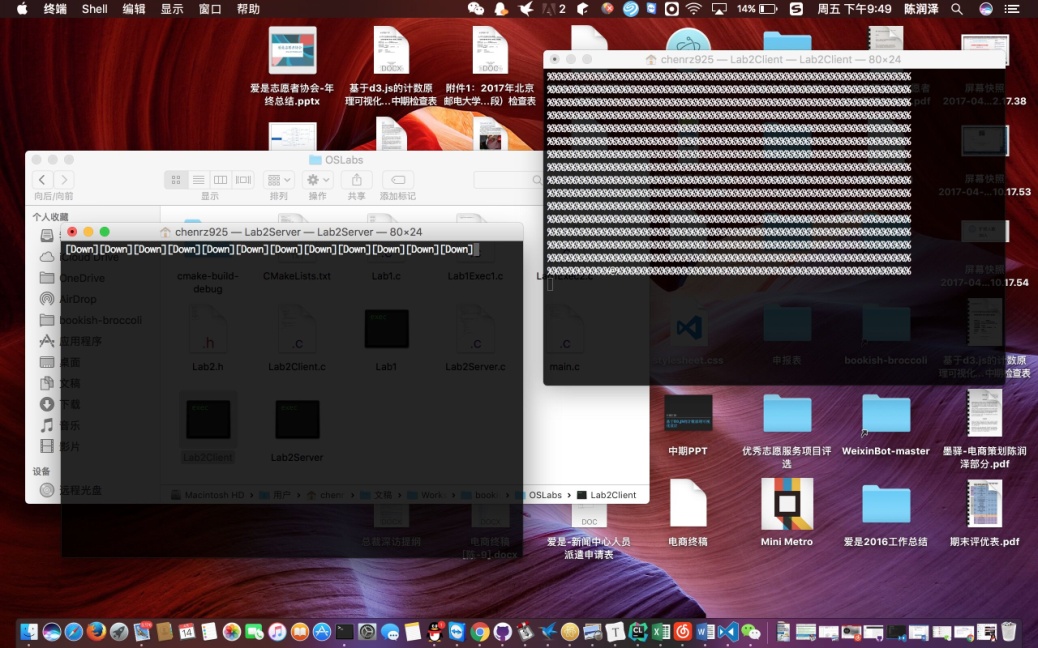
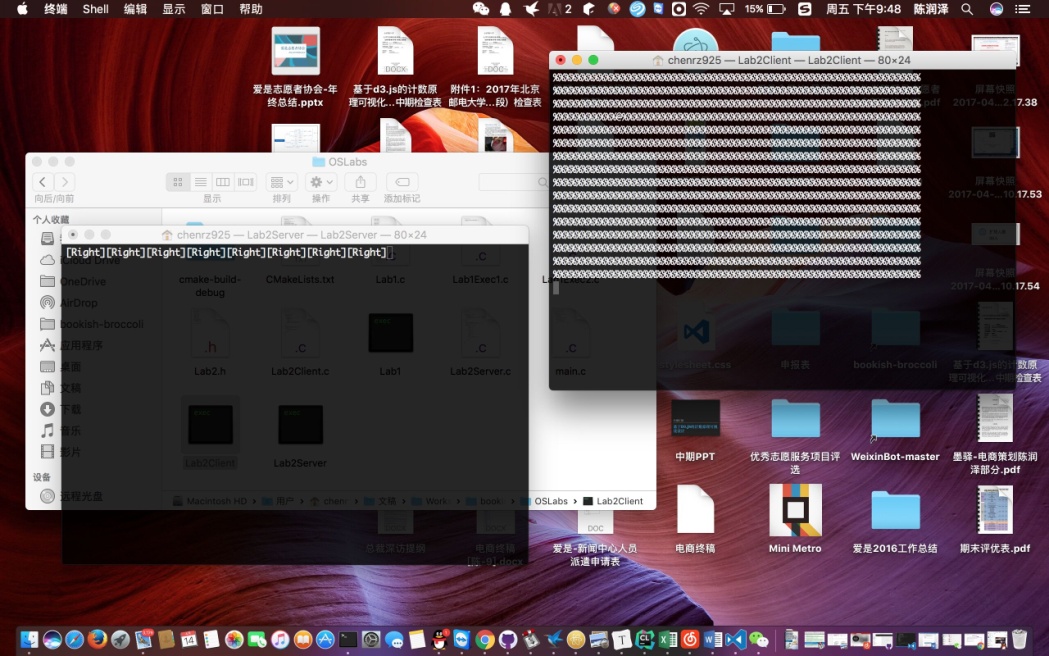
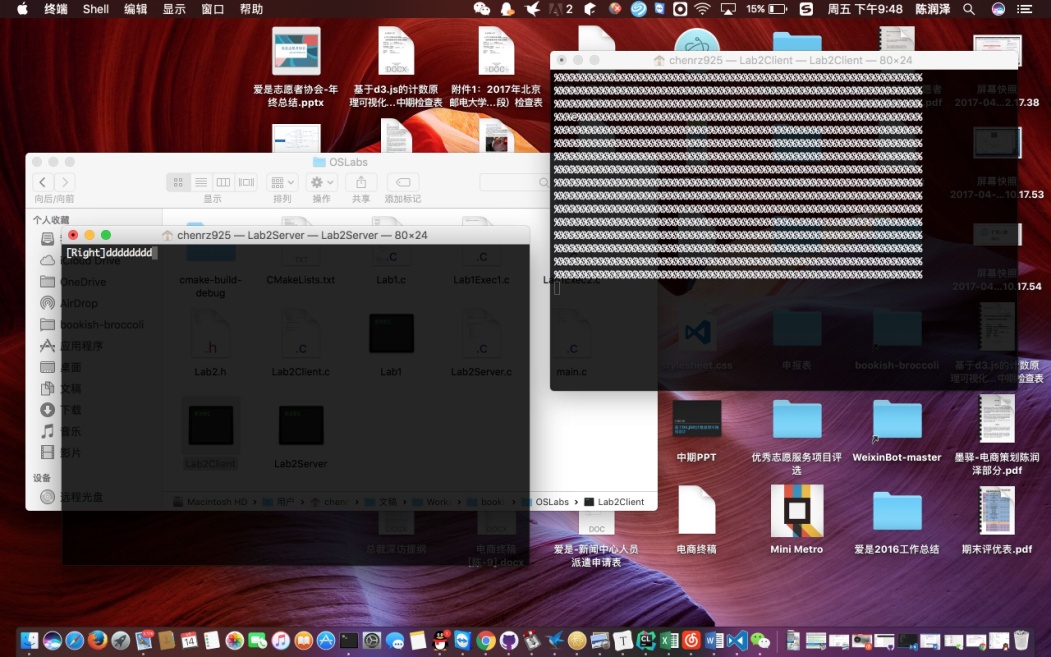
**#endif**

**6.2 实验结果截图**

**1 两个Client端将数据发送到Server端**

****

**2 通过在Server端键入一系列wasd指令 控制Client端光标的移动**

****