/\*

\* lookup6 : does no looking up locally, but instead asks

\* a server for the answer. Communication is by shared memory

\*

\* Synchronization is by two semaphores:

\* One excludes other clients while we talk to server

\* The other is used BOTH to alert the server and to await his reply

\* Client raises a semaphore by 2. Server drops by 1 before and

\* after doing lookup

\* The same key (which is passed as resource) is used BOTH

\* for shared memory & semaphores.

\*/

#include <sys/ipc.h>

#include <sys/shm.h>

#include <sys/sem.h>

#include <sys/types.h>

#include <sys/mman.h>

#include <stdlib.h>

#include <string.h>

#include "dict.h"

int lookup(Dictrec \* sought, const char \* resource) {

static int shmid,semid;

long key = strtol(resource,(char \*\*)NULL,0);

static Dictrec \* shm;

struct sembuf grab = {0,-1,SEM\_UNDO}; /\* mutex other clients \*/

struct sembuf release = {0,1,SEM\_UNDO}; /\* release mtx to other clients \*/

struct sembuf alert = {1,2,SEM\_UNDO}; /\* wake up server \*/

struct sembuf await = {1,0,0}; /\* wait for server \*/

static int first\_time = 1;

//有兩個信號量集 分別叫信號量0 與信號量1, EX:grab = {0,-1,SEM\_UNDO} 0表示信號量0,-1表示執行後-1,SEM\_UNDO

if (first\_time) { /\* attach to shared memory & semaphores \*/

first\_time = 0;

/\* Connect to shared memory.

\* Fill in code. \*/

if((shmid=shmget(key,sizeof(Dictrec),0666))==-1){

perror("shmget");

exit(1);

}

/\* Get shared memory virtual address.

\* Fill in code. \*/

shm = shmat(shmid,NULL,0);

if(shm==MAP\_FAILED){

perror("shmat");

exit(1);

}

/\* Get semaphore.

\* Fill in code. \*/

if((semid=semget(key ,2,0))==-1){

perror("semget");

exit(1);

}

}

/\* Reserve semaphore so other clients will wait.

\* Fill in code. \*/

if(semop(semid,&grab,1)==-1){

perror("semop2");

exit(1);

}

strcpy(shm->word,sought->word);

/\* Alert server. Bump semaphore 1 up by 2.

\* Fill in code. \*/

if(semop(semid,&alert,1)==-1){

perror("semop2");

exit(1);

}

/\* Wait for server to finish. Server will have set sem 1 to zero.

\* Fill in code. \*/

if(semop(semid,&await,1)==-1){

perror("semop2");

exit(1);

}

/\* Done using the server. Release to other clients.

\* Fill in code. \*/

if(semop(semid,&release,1)==-1){

perror("semop1");

exit(1);

}

if (strcmp(shm->text,"XXXX") != 0) {

strcpy(sought->text,shm->text);

return FOUND;

}

return NOTFOUND;

}