HACETTEPE UNIVERSITY COMPUTER SCİENCE ENGINEERING DEPARTMENT

BIL-235 LABORATORY

3. EXPERIMENT REPORT

Name :Serdar
Surname :GÜL
Number :20421689

Subject :Operation Systems
Language :ANSI C and Linux Kernal
System :Linux
Platform :Eclipse Europa Ganymede
Advisors :Dr. Ahmet Burak CAN , R.A. Ali Seydi KEÇELİ
Due Date :15.11.2008

SOFTWARE USAGE

It is used so easily.At first you open the command line application (Linux terminal).After this you will open the directory of the Project and you will just write commands and the program will do everything for you..You can see the results in the kernal...

This program has 3 different usage types and takes 3 parameters except the name of the program..

First one is the type parameter.Second is the first directory’s name.And the third is the second directory’s name.We can give some examples for the usage of the program that..

```
Dirsync –d ./serdar ./bil235
Dirsync –i ./serdar ./bil235
Dirsync –u ./serdar ./bil235
```

Their meanings will be explained in the deeper levels of the report..

So we can say that is has a basic usage…

ERROR MESSAGES

“The parameters are wrong” : If user enters less than 4 arguments to the command line program will give this error message to the user.

“The parameters are too much” : If the user enters too much parameters there are not necessary , program will give this error message to the user…

“stat error on .. ” : If the program can not open the file in the one of the given directories , program will give this error message..

SOFTWARE DESIGN NOTES

PROBLEM

In this experiment our main problem is the usage of the ANSI C programming language with the kernal commands of the LINUX operating system.We can cope with a lot of problems indeed.

-User can give wrong parameters
-User can give less parameters to the command line while executing the program.(This program will run in 4 arguments for the command line)
-User can give more parameters.(User can give unnecessary parameters to the command line..)
-User may give the non-existed directories as a parameter.
-User can write a type that is not suitable for the experiment.
-User can write the directories with the wrong style…

In this experiment we will have 2 directories and an option which determines the operation will be done to the directories..

Dirsync –d ./serdar ./bil235
Dirsync –i ./serdar ./bil235
Dirsync –u ./serdar ./bil235

In this commands –d means that we will find the differences between the given directories and we will print the tree structure of the directories.And also that we will print the values of the files which are in the directories.(name,size,date and state)

-u means that we will find the files and directories which are not exist in the both of the given directories and copy them to the directory which has not…

And –i means that we will delete the old files and directories in the given directories.For example we will copy a text file from first directory to the second directory.We will delete the text file from the first directory because the common one is in the second directory…

And some of the commands are forbidden in this experiment like

Fopen
Fclose
Printf
System
Listdir etc..

So we will use the other commands and if needed we will make some functions which has done the job which the forbidden commmands make…

It will be a big problem to solve the experiment and print the results to the user…

SOLUTION

At first we will take the parameters from the command line.If user wants to run the program with files , we should open the directories.

We will control the second parameter of the program…

If the parameter is –d we will the operations explained above.

And we will wander it recursively to cope with some of the problems for the design of the experiment.

We will have a print design that will be like in the experiment sheet.And sizes and names can be different so we will control the name and size to make a good and smooth result to the command prompt.

And also that our rinting result will show the relationship between the directories and their files.

And we will print the properties of the files to the command prompt..

If the second parameter will be –u we will wander the directories recursively and at first we will copy the files and directories which are not exist from first to second.

After that we will recursively wander the other directory and we will copy the directories and their files which are not exist to the first directory.

With using the same method twice we will union both of the directories.

If the second parameter will be –i we will wander the directories recursively and at first we will remove the files and directories which are not common in the first directory..

After that we will recursively wander the other directory and we will remove the directories and their files which are not common in the first directory.

With using the same method twice we will intersect both of the directories.

SYSTEM CHARTS

INPUT	PROGRAM	OUTPUT
Directories and option	dirsync.c	command prompt

MAIN DATA STRUCTURES

I use the ready structures of the some of the C libraries and I use a lot of variables to cope with the experiment..

For example I use the dirent and stat data structures that…

Stat is for the file and it has some of the variables we can use that..For example

- st_mode -- file permissions (user, other, group) and flags
- st_ino -- file serial number
- st_dev -- file device number
- st_nlink -- file link count
- st_uid -- the owner's user ID
- st_gid -- the owner's group ID
- st_size -- file size in bytes (for regular files)

- st_atime -- the last access time
- st_mtime -- the last modification time
- st_ctime -- the file's creation time

And also that dirent is for the directories and it has some variables that

```
struct dirent {
        long            d_ino;
        __kernel_off_t  d_off;
        unsigned short  d_reclen;
        char            d_name[256];
};
```

And I use some character arrays fort he path of the directories some controls for the loops and recursive functions too..

For example

```
DIR *dir;
struct dirent *entry;
int count;
int b,c;
char path[1025];
struct stat info;
char *mode , *firstdir , *seconddir ;
```

ALGORITHM

Take the command from the command line
Take the second argument as an type
If parameter is –d
Open the directories
If one of them can not be opened
Give an error message and terminate the program
Recursvely traverse the both directories and print the properties and the hierarchies of the directories
Else If parameter is –u
Open the directories
If one of them can not be opened
Give an error message and terminate the program
Recursvely traverse the both directories and copy the non existed files and the directories to both of them.
Else If parameter is –i
Open the directories
If one of them can not be opened
Give an error message and terminate the program
Recursvely traverse the both directories and remove the non common files and the directories from both of them.

Algorithm was so short because I explained detailly in the solution part of the program...

EXECUTION FLOW BETWEEN SUBPROGRAMS

There are some functions that are used combi nely.These are;

void traverse(char *,int,int) : it recursively traverse the directory and print the hierarchy and properties of the files to the command prompt.

```
void traverse(char *fn, int indent,int counter) {
DIR *dir;
struct dirent *entry;
int count;
int b,c;
char path[1025];
struct stat info;

    for (count=0; count<indent; count++) printf(" ");
        //printf("%s\n", fn);

    if ((dir = opendir(fn)) == NULL)
        perror("opendir() error");
    else
    {
        while ((entry = readdir(dir)) != NULL) {
            if (strcmp(entry->d_name, ".") &&
strcmp(entry->d_name, "..")) {
                strcpy(path, fn);
                strcat(path, "/");
                strcat(path, entry->d_name);

                if(indent == 0)
                {
                    printf("|-- ");
                }
```

```
                    else
                    {
                        printf("|");
                        for(c = 0 ; c < indent ; c++)
                            printf("\t");
                        printf("|-- ");
                    }

                    //printf("%s\n", entry->d_name);
                    b = strlen(entry->d_name);
                    if(entry->d_type != DT_DIR)
                    {
                        printf("%s",entry->d_name);
                        if (stat(entry->d_name, &info) <
0)
                        {
                            if(b > 11)
                            {
                                for(c = 4 ; c > indent ;
c--)
                                    printf("\t");
                                printf(" %d",
info.st_size);
                                if(info.st_size >
1000000)
                                {
                                    printf("\t %s\n",
ctime(&info.st_mtime));
                                }
                                else
                                {
                                    printf("\t\t %s\n",
ctime(&info.st_mtime));
                                }
                            }
                            else
                            {
                                for(c = 5 ; c > indent ;
c--)
                                    printf("\t");
                                printf(" %d",
info.st_size);
```

```
                                if(info.st_size >
1000000)
                                {
                                    printf("\t %s\n",
ctime(&info.st_mtime));
                                }
                                else
                                {
                                    printf("\t\t %s\n",
ctime(&info.st_mtime));
                                }
                            }
                        }
                    }
                    else
                    {
                        printf("%s/\n",entry->d_name);
                        counter++;
                    }


                    if (stat(path, &info) != 0)
                        fprintf(stderr, "stat() error on
%s: %s\n", path,
                                strerror(errno));
                    else if (S_ISDIR(info.st_mode))
                    {
                        //printf("%s\n",path);
                        traverse(path,
indent+1,counter);
                    }
                }
            }
            closedir(dr);
        }

}// end of the traverse
```

Void u (char *,char *) : Recursvely traverse the both directories and copy the non existed files and the directories to both of them.

Void i (char *,char *) : Recursvely traverse the both directories and remove the non common files and the directories to both of them.

void f_copy(char* path1, char* path2, char* name):it is used to copy a file from one directory to the secodn direstory...

And our main function is

```
int main(int argc , char *argv[])
{
    char *mode , *firstdir , *seconddir ;

    if(argc < 4)
    {
        printf("Parameters are wrong.\n");
        exit(-1);
    }
    else if (argc > 4)
    {
        printf("There are too much parameters.\n");
        exit(-5);
    }
    else
    {
        mode = argv[1];
        firstdir = argv[2];
        seconddir = argv[3];
        if(strcmp(mode,"-d") == 0)
        {
            printf("%s\t\t\t\t\t size\t\t time
stamp\t\t\t state",firstdir);
            traverse(firstdir,0,0);
            printf("\n%s\t\t\t\t\t size\t\t time
stamp\t\t\t state",seconddir);
            traverse(seconddir,0,0);
        }
        else if(strcmp(mode,"-i") == 0)
```

```
        {
            i(argv[2], argv[3]);
            i(argv[3], argv[2]);
        }
        else if(strcmp(mode,"-u") == 0)
        {
            u(argv[2], argv[3]);
            u(argv[3], argv[2]);
        }
        else
        {
            printf("This mode is not defined in this
program.\n");
            exit(-2);
        }
    }
    return 0;
}// end of the main function
```

SOFTWARE TESTING NOTES

SOFTWARE RELIABILITY

It is a good program that has too little bugs.If In sometimes it can give wrong solutions because I don’t have enough time to test it

But except a few wrongs program will run in

Except this , this program runs correctly and it has lots of controls that user can give wrong expression.
It controls these events.

-User can give less parameters to the command line while executing the program.(This program will run in 3 arguments for the files and 4 arguments for the commands which are written to the command line)

-User can give more parameters.(User can give unnecessary parameters to the command line..)

-User can give a wrong input or output file..

SOFTWARE EXTENDIBILITY AND UPRADIBILITY

This program can be extended with some small changes and we can use it with the operations we can cope with another data structures like vectors , arraylists and we can make our own library to this programming language.

PERFORMANCE CONSİDERATIONS

This program runs rapidly and with some small changes we can make it faster then all.

And it gives the true results which are expected from us in a very little time.

COMMENTS

I think that this experiment will be so useful for us now and after these days.

For example I learn C language and linux kernal again and I don’t forget it after this experiment.

And it was so useful to us to finding great ideas for the programming and it causes us to study much more than.

Briefly it was a so useful and good experiment.

And I take the Bil-341 course,because of this , program gives false results in sometimes rarely .So I apologise from you.And while you were giving the not for the experiment if you think about this I will be so appreciated to you..

Regards..

RESOURCES

E-books

Ileri C Programlama by MURAT TASBASI
C Programlama Dili
How To Program With C/C++ by DEITEL
PC Assembly Language by PAUL CARTER
The Art of Assembly Language

Web Sites
www.programlama.com