System Programming

Assignment II

Linux Dictionary Device

Beycan KAHRAMAN
040020337
29.12.2006
Turgut UYAR

1. INTRODUCTION

We are writing a linux dictionary device, that could store words in Turkish or English, which also stores their meanings in other language.

We may use Turkish-English or English-Turkish Dictionary Device both. So one may add the same word and its definition by two ways.

```
For Example:            (English Word & Its Definition)
                  i.    Open English-Turkish Dictionary
                        Add Word-Definition to Dictionary
                  ii.   Open Turkish-English Dictionary
                        Add Definition-Word to Dictionary
```

We are also expected to write a read(...) function that could read from device as a character device, that means it will reads n bytes from current position from the device.

2. ENVIRONMENT

I have started building the project in Visual-C++ 6.0 in Microsoft operating system, because it is easier to debug the main parts.
Here, I have created an double pointered linked list(linked_list.cpp) and string function tester which will supplies me necessary string functions that are not in asm/string library(str_func_tester.c).

After being sure about their performance and stability, I have transported my work to Linux environment. I have used Kubuntu 6.10 Edgy Eft Stable Linux to implement my work.

```
My Kernel Version:      2.6.15-27-386
My Gcc Version:         4.0.3
```

First of all, I tried to run scull device example. As I get errors about could not find development files for kernel, I have downloaded kernel src, header files, build, etc. with Adept (Package Manager). So that I could manage to run the device.

After adding the created functions to 040020337.c I could manage to create dictionary device that could work.

I get only two warning in readFromXxMemory functions as 'uyarı: ISO C90 kod ve bildirimlerin karışımına izin vermez' on line 196 and 221. This warnings occured from

```
int currentLen = strlen(newTmpTr->stringTr);
```

line that I could not solve.

3. FUNCTIOONS and DECLARATIONS

I have used a linked list with two next pointer that is standart and there is no need to explain its work. linked_list.cpp has tested in both Linux and Windows environments.

There is only one distinct function to read n characters from linked list. It is given below.

```
int readFromTrMemory(char* in, int count){
      if(!newTmpTr)     //  dic_wd_letTr =  dictionary_word_letterTr
            return 0;
      int currentLen = strlen(newTmpTr->stringTr);
      int leftHere = currentLen - dictionary_word_letterTr + 1;
      if(count == leftHere){
            strTcat(in, newTmpTr->stringTr, dic_wd_letTr, currentLen);
            strcat(in, "\n");
            count = 0;
            dictionary_word_letterTr = currentLen;
            return leftHere;
      }else if(count > leftHere){
            strTcat(in, newTmpTr->stringTr, dic_wd_letTr, currentLen);
            strcat(in, "\n");
            count -= leftHere;
            dictionary_word_letterTr = 0;
            newTmpTr = newTmpTr->nextTr;
            return leftHere + readFromTrMemory(in, count);
      }else{      // c = count
            strTcat(in, newTmpTr->strTr, dic_wd_letTr, dic_wd_letTr + c);
            dictionary_word_letterTr += count;
            return count;
      }
}
```

In this functions I have calculated the return value for Turkish dictionary read() functions. It returns n charachters from list where endline charachters have placed between them.

Besides readFromTrMemory() function I have created strTcat(...) and seperateString(...) functions. (I have tested them in str_func_tester.c file) The first funciton, strTcat(...), cats the second charachter array to the first charachter array from cth element to dth element.

```
void strTcat(char *b, char *a, int c, int d){
      int len = strlen(b), i;
      for(i=0; i<d-c; ++i)
            b[i+len] = a[i+c];
      b[i+len] = '\0';
}
```

seperateString(...) function is seperates the coming string from user, that will be added to the dictionary.

```
For example:      Coming String     = “sun:gunes”     results
                  First String      = “sun”
                  Second String     = “gunes”
```

```
void seperateString(char* in, char* first, char* second){
      int i=0, j;

      while(in[i] != ':' && in[i] != '\0' && i < MAXWORDLEN-2){
            first[i] = in[i];
            ++i;
      }

            first[i] = '\0';
      if(in[i] == '\0'){
            second[0] = '\0';
      }else{
            j = i+1;
            while(in[j] != '\0' && in[j] != ':' && j-i < MAXWORDLEN-1){
                  second[j-i-1] = in[j];
                  ++j;
            }
            second[j-i-1] = '\0';
      }
}
```

4. EXECUTION

To create the dictionary device and test it with dic_tester.c; firstly sure that the kernel header and build files be in your /lib/modules/ $(uname -r)/.. directory.

After this be sure none of the devices to use major device number 250 as our dictionary uses this major value default.

Lastly run the commands given below to add dictionary device to the kernel and test it with the tester application.

```
----------------------------------------------------
To Create the Dictionary Device:
----------------------------------------------------
cd ....AssignmentII

make -C /lib/modules/$(uname -r)/build M=`pwd` modules
insmod ./040020337.ko
cat /proc/devices | grep dictionary  // we use dictionary as device name

mknod /dev/dic0 c 250 0
mknod /dev/dic1 c 250 1         crete devices

----------------------------------------------------
Removing the Dictionary Device:
----------------------------------------------------

rmmod 040020337
rm /dev/dic0
rm /dev/dic1


----------------------------------------------------
Running dic_tester.c:
----------------------------------------------------

gcc dic_tester.c -o testDic
./testDic
```

CONCLUSION

After finishing assignment II, I have analysed that writing a device driver has very easy and standart steps. One, who has learn the standart code running under the devices, may easily write device drivers for linux. As a result, I am delighted to finish this assignment and see that it works.