Shell Enhancement On Sylix OS Design Document

Yifan Jiang
Shuxin Zheng
Zhiyang Yu

April 2021

Contents

1 Project Background and Description

SylixOS is an operating system designed with an open-source kernel and developed completely by the Chinese. With complete functions, it has a wide range of applications in the fields of national defense, aerospace, electric power, rail transit, and industrial automation.The core code autonomy rate is 100%, and the autonomy rate of all codes reaches 89.1%. The interface is compatible with POSIX standards.

SylixOS already has a set of command line implementation mechanisms, shell command lines, and simple shell scripting capabilities.SylixOs is also a Real-time operating system,which means the correctness of the system depends not only on the logical result of the system calculation, but also on the time at which this result is produced. In other words, all events during system design can be responded to within a specified time.

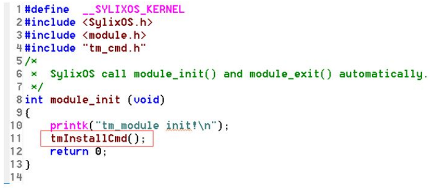
The goal of our project is to design and improve the command line mechanism and enhance the shell command line programming ability.

2 Implementation

For the design of the function, we plan to look for similar functions in other operating systems. We will learn how they realize the function and transfer these on SylixOs properly. During the process of project, we will try to make improvement on these existed implements to design a new implement that adapt to SylixOs better.

As for the implementation, we mainly modify and realize our design in two ways [1]. We can register a new command and implement its function.Then upload the new shell command into the kernel and rebuild(Figure.1). Or we can simply modify the exist shell command in **libsylixos/SylixOS/shell** directly(Figure.2) [2].

The details of technique route are as follows. For automatic completion, the design of current stage is brutal and simple. We use b+ tree to store all key words, corresponding parameters and help documents. Whenever shell detects changes in user input, search for matching results, refresh shell interface and show results list. Other functions like implementing branching and loop statement are viewed as new kinds of grammar rules in shell. To make new grammars work, the they could be registered as functions. Branching statement requires conditionally ignoring some input in shell, and use loop statements in function implementation could realize loop statement in shell. Finally, similar implementation as Linux OS could be applied to common functions like find and grep in Sylix OS.



```c
#define  __SYLIXOS_KERNEL
#include <SylixOS.h>
#include <module.h>
#include "tm_cmd.h"
/*
 *  SylixOS call module_init() and module_exit() automatically.
 */
int module_init (void)
{
    printk("tm_module init!\n");
    tmInstallCmd();
    return 0;
}

```

Figure 1: register a new command

2.1 auto-completion of command line keywords

- We can make SylixOS store an operation of pressing the tab. And then when a pressing action is detected after user has input some command. Auto-completion is triggered. Shell will first check whether the existed command contains ./,.,/. If then and the path is correctly, shell will display all files in that directory that contains keywords in circulation. Or the shell will simply do nothing.
- SylixOS will use the letters we saved in step 2 to search the file that fits best. The matching process may first find the file with shortest name.

For candidate files whose name is of same length, SylixOS will find file in lexicographical order

- We can use **help** command to search and match all commands that contains the keyword input by user.

2.2 Implement associative help for command line keywords and arguments

- After typing the full function name, as we introduced in the last section, the user can press the tab, the shell will show a possible parameter, and then press tab again. It will change to another possible parameter. Those parameters will show up in alphabetical order and circulation. .
- And the implementation is similar to the last section. We can use **help** command to search and match all commands that contains the keyword input by user.

2.3 Support find, grep, awk, sed utility commands

- For command find, it is used to find files in the specified directory. After user enter find + filename, the SylixOs can search all files in current path. To implement this, we can use **ls** or **bfs** to show all file in the folder, and search whether the file are in the dirctory.
- For command grep, it is used to find the strings in the file that match the conditions.After user enter grep + keyword, the SylixOs can search and print all files satisfying the condition in current path. we can using the keyword as a filter all files in current path. filter the input stream according to its parameter. After that, we print the information after filtering.
- For command sed, it is a line-oriented processing tool that uses the "line" as the processing unit, processing each line, and outputting the results to standard output. We can use fopen() to read a file, then modify it according to the rules given by the parameters.

3 Expected goals

- Realize the automatic completion function of command line keywords.
- Realize lenovo help for command line keywords and parameters.
- Implement pipeline operations between commands —, support output filtering and. split-screen display.
- support branch: if/elif/else

- support loop: while/for; break/continue
- support find,grep,sed

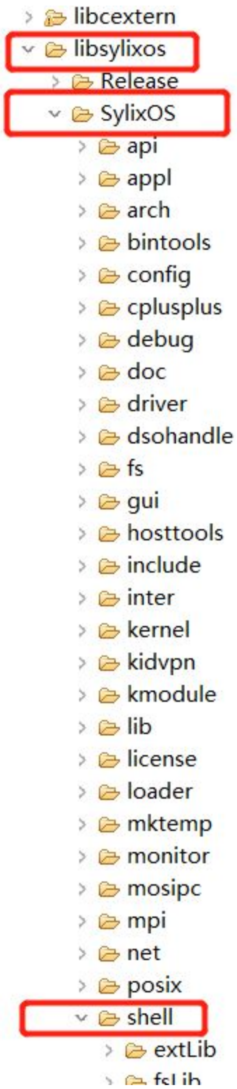
4 Division of labor

Our group design and code the project together.

References

[1] *SylixOS shell Enhanced development guidance document.docx.*

[2] *SylixOS shell user manual.*



```
> libcextern
v libsylixos
  > Release
  v SylixOS
    > api
    > appl
    > arch
    > bintools
    > config
    > cplusplus
    > debug
    > doc
    > driver
    > dsohandle
    > fs
    > gui
    > hosttools
    > include
    > inter
    > kernel
    > kidvpn
    > kmodule
    > lib
    > license
    > loader
    > mktemp
    > monitor
    > mosipc
    > mpi
    > net
    > posix
    v shell
      > extLib
      > fsLib
```

Figure 2: modify the exist shell comman