Operating Systems Laboratory (Gx) (2017-18)

Implementation of a primitve shell

A *shell* is an interface program between the **Operating System** and a **User**. That is, a shell takes commands from the user and executes those commands through invocation of appropriate system calls as depicted in the following figure.
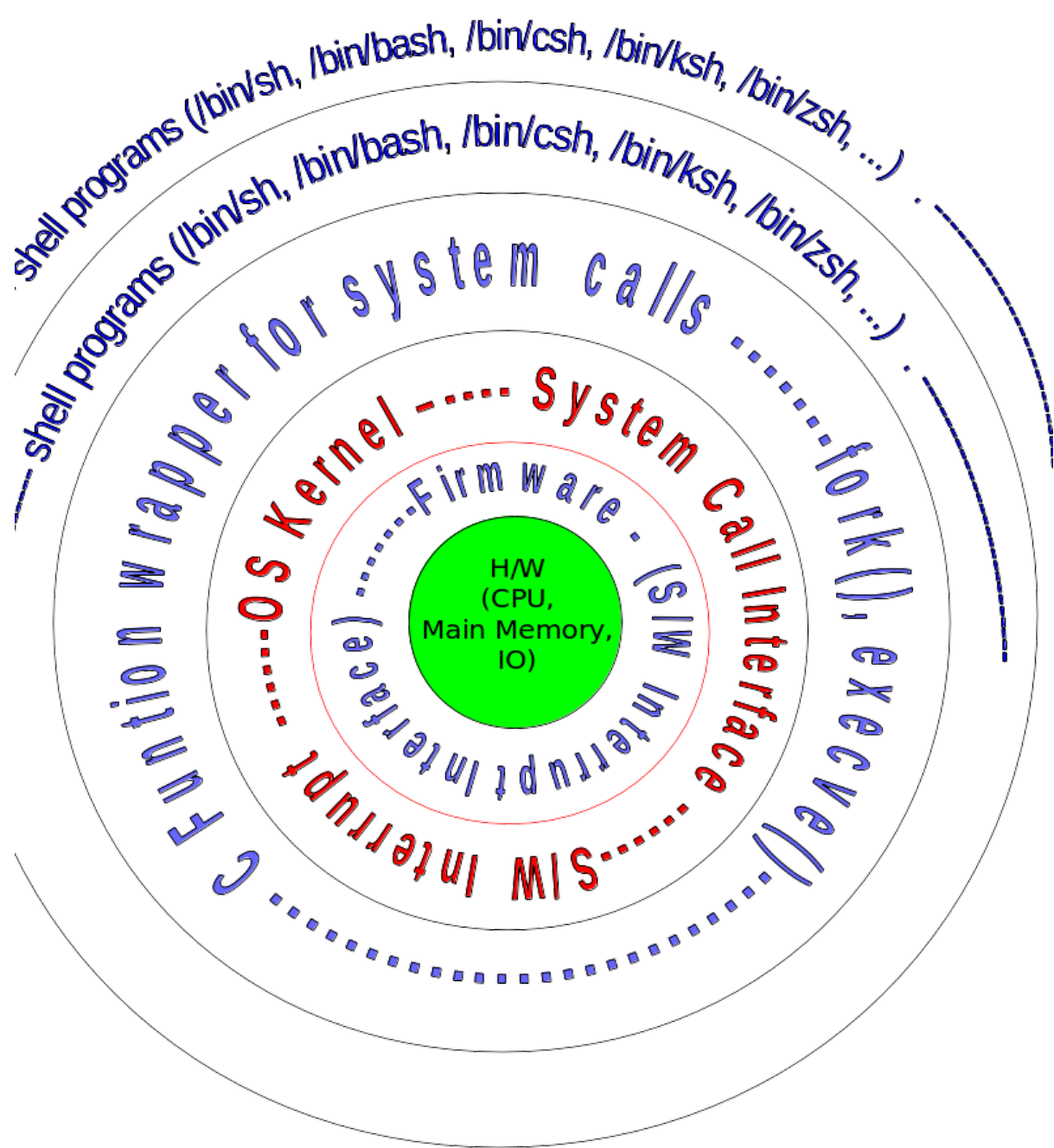
You are familiar with Bourne Shell (/bin/sh) or Bourne Again Shell (/bin/bash). You may refer to the manual (guide) on bash to have an idea about the language (shell script) in which user may issue commands to it. The present assignment is to construct a trivial shell (say, *mysh*).

In general the commands a shell can execute are of 2 types - **internal commands** and **external commands**. Commands like **cd**, **pwd**, etc., are internal commands. Please note that the shell does not need any external executable file for the internal commands. Whereas, external commands (eg., **ls**, **vi**,

...) are essentially executable files and are managed by the shell by first creation of child process (*fork()*)and then making the child process execute (execve()) the file.

The trivial shell should support internal commands like *cd, pwd, clear, exit*, etc. It should support any external command with or without command line arguments. For example, myshell should be able to execute both "*ls*" as well as "*ls -lt*" as external commands. Multiple commands can be can be connected by "*;*" (semicolon, sequential execution) "*&&*" (logical and), "*||*" (logical or). That is *"command1 ; commnd2"* or *"command1 && commnd2"* or *"command1 || commnd2"* are all valid commands for "*mysh*". You may refer to the guides (associated herewith) for the semantics (meaning) of these operators.

The algorithm for "mysh" should be quite straight forward.

> *do*
>
> *show the shell prompt*
>
> *read a line in the string variable cmd*
>
> *parse cmd into subcommands /* in "command1 ; command2" command1 and command2 are subcommands */*
>
> *parse the subcommands for command line arguments*
>
> *if a subcommand is internal then do what is necessary for it. Else if there is an executable for the command then fork() and let the child process execute the executable*
> *while (true);*

Please note that "*mysh*" can be executed interactively, that is, upon execution it accepts commands from the *stdin*. "*mysh*" can also be executed in batch mode (for example, *mysh file.sh*) when **mysh** will read the commands from the file "*file.sh*" and execute them.

📄 abs-guide.pdf
📄 Bash-Beginners-Guide.pdf
📄 bash.pdf

Submission status

Attempt number	This is attempt 1.
Submission status	No attempt
Grading status	Not graded
Due date	Tuesday, 27 March 2018, 4:35 PM
Time remaining	2 hours 24 mins
Last modified	-
Submission comments	➕ Comments (0)

Add submission

Make changes to your submission