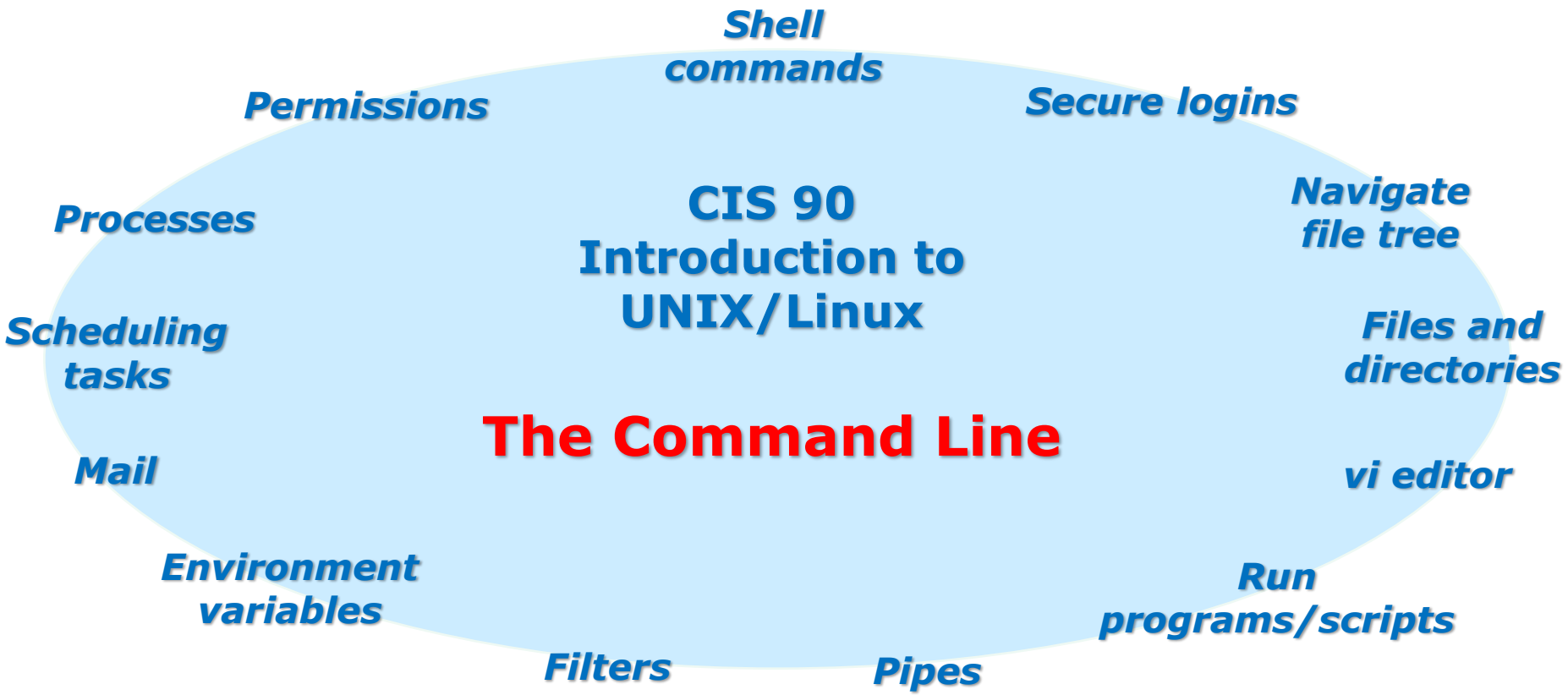
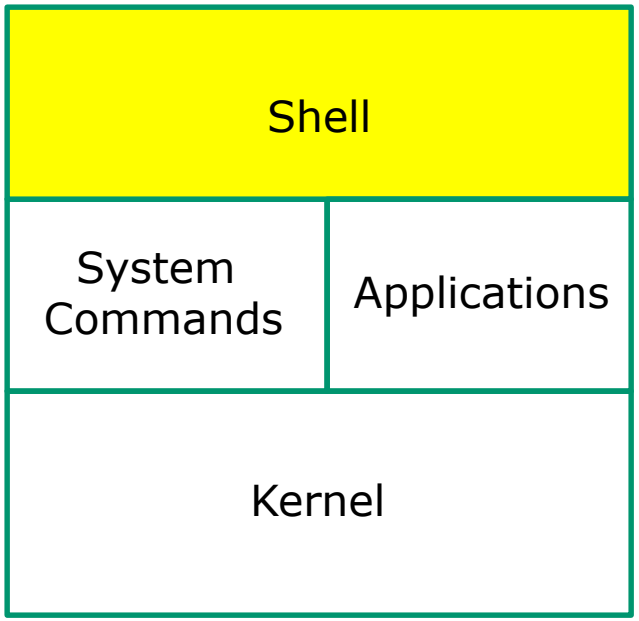


Student Learner Outcomes

1. Navigate and manage the UNIX/Linux file system by viewing, copying, moving, renaming, creating, and removing files and directories.
2. Use the UNIX features of file redirection and pipelines to control the flow of data to and from various commands.
3. With the aid of online manual pages, execute UNIX system commands from either a keyboard or a shell script using correct command syntax.

The Shell (six steps)

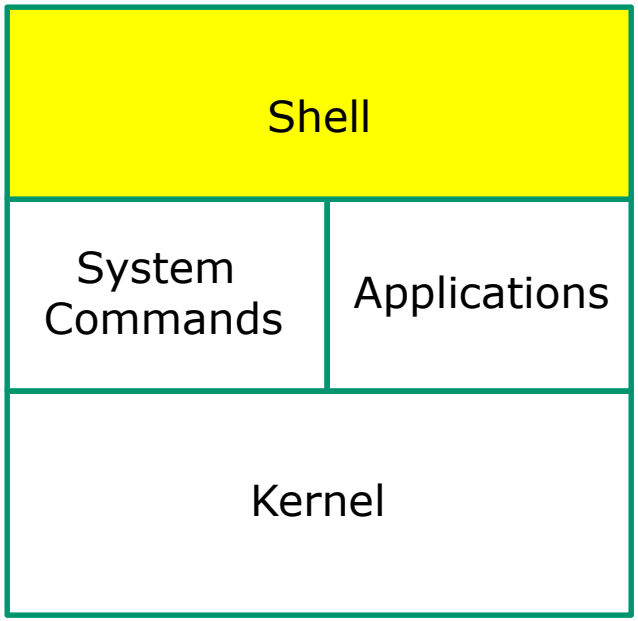
The Shell



- Allows users to interact with the computer via a "**command line**".
- **Prompts** for a command, parses the command, finds the right program and gets that program executed.
- Is called a "**shell**" because it hides the underlying operating system.

- Multiple shell programs are available: **sh** (Bourne shell), **bash** (Bourne Again shell), **csh** (C shell), **ksh** (Korn shell).
- The shell is a **user interface** and a **programming language** (scripts).
- GNOME and KDE desktops could be called **graphical shells**

Life of the Shell



This is what a shell does:

1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

Life of the Shell

Example:

```
/home/cis90/simben $ ls -lt proposal1 proposal2
-rw-r--r--. 1 simben90 cis90 1074 Aug 26  2003 proposal1
-rw-r--r--. 1 simben90 cis90 2175 Jul 20  2001 proposal2
/home/cis90/simben $
```

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

Lets take a deep dive into how a command gets executed.

Note it is always a team effort by both the shell and the command.

Life of the Shell

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

1) Prompt user for a command

Example:

The shell begins by outputting the prompt (which is based on the PS1 variable)

```
/home/cis90/simben $ ls -lt proposal1 proposal2
```

Then you type the command

FYI, you can mimic outputting the prompt yourself with these commands:

```
/home/cis90/simben $ echo $PS1
$PWD $
/home/cis90/simben $ echo $PWD $
/home/cis90/simben $
/home/cis90/simben $
```

to show value of PS1 variable

echo the output of the previous command

was output by the echo command above

was output by the shell (the same output)

Life of the Shell

2) Parse command user typed

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

Example:

```
ls -lt proposal1 proposal2
```

During the parse step the shell identifies all options & arguments, handles any metacharacters and redirection

- Command = ls
- 2 Options = l, t
- 2 Arguments = proposal1, proposal2
- No Redirection

Life of the Shell

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

3) Search path for the program to run

ls -lt proposal1 proposal2

Use this command to see the path directories (separated by :'s) on your path

```
/home/cis90/simben $ echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/cis90/simben/../bin:
/home/cis90/simben/bin:.
```

*The shell will search each directory in order for an **ls** command*

1st directory: /usr/local/bin *nope, not found here*
2nd directory: /usr/bin *bingo, found here!*
3rd directory: /usr/local/sbin
4th directory: /usr/sbin
5th directory: /home/cis90/simben/../bin
6th directory: /home/cis90/simben/bin
7th directory: .
.
.

Note: If the shell cannot find the command on the path it will output something like "command not found"

Try mimicking what the shell does to search for ls:

```
/home/cis90/simben $ ls /usr/local/bin/ls
ls: cannot access /usr/local/bin/ls: No such
file or directory

/home/cis90/simben $  ls /usr/bin/ls
/usr/bin/ls
```

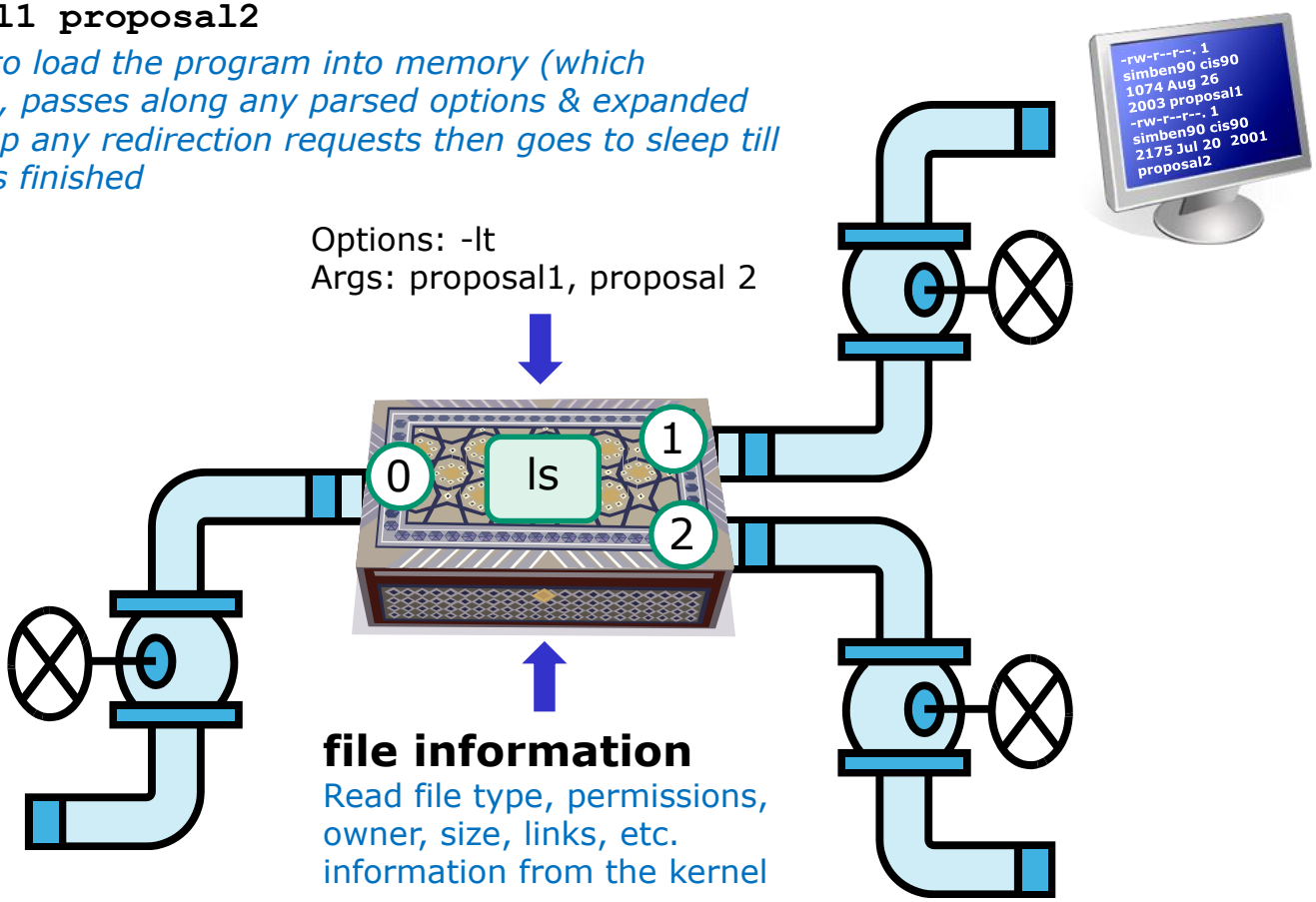
Life of the Shell

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

4) Execute the command

`ls -lt proposal1 proposal2`

Invokes the kernel to load the program into memory (which becomes a process), passes along any parsed options & expanded arguments, hooks up any redirection requests then goes to sleep till the new process has finished



Life of the Shell

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

5) Nap while the command (process) runs to completion

(The shell, itself a loaded process, goes into the sleep state and waits till the command process is finished)

```
/home/cis90/simben $ ls -lt proposal1 proposal2
-rw-r--r--. 1 simben90 cis90 1074 Aug 26  2003 proposal1
-rw-r--r--. 1 simben90 cis90 2175 Jul 20  2001 proposal2
```

The shell sleeps while the ls process outputs these two lines

Life of the Shell

6) And do it all over again
   … go to step 1

Shell Steps
1) Prompt
2) Parse
3) Search
4) Execute
5) Nap
6) Repeat

Life of the Shell

A `/home/cis90/simben $ ls -lt proposal1 proposal2`
`-bash: Ls: command not found`

What's wrong?
Who output the error?

B `/home/cis90/simben $ ls -lt proposal1 proposal5`
`ls: cannot access proposal5: No such file or directory`
`-rw-r--r--. 1 simben90 cis90 1074 Aug 26  2003 proposal1`

What's wrong?
Who output the error?

C `/home/cis90/simben $ ls -lw proposal1 proposal2`
`ls: invalid line width: proposal1`

What's wrong?
Who output the error?

D `/home/cis90/simben $ ls -lt proposal1proposal2`
`ls: cannot access proposal1proposal2: No such file or directory`

What's wrong?
Who output the error?

E `/home/cis90/simben $ ls-lt proposal1 proposal2`
`-bash: ls-lt: command not found`

What's wrong?
Who output the error?