TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU

TEXAS
The University of Texas at Austin

# How Lmod processes TCL modulefiles

Robert McLay

November 8, 2022

# Outline


- Original Idea was just to support Lua based modules
- Hard to get sites to translate their existing modules (including TACC!)
- So how to support TCL modulefiles?
- What strategies are possible?
- What technique does Lmod use
- Why it is never going to be perfect.

# How could TCL modulefiles be supported?

- Could write a TCL interpreter in Lua
- Ugh!, too much work, Hard to maintain, Hard to get right
- Could rewrite much of the Lmod lua code in TCL
- Ugh! (same reasons!)
- Try something else

# Something good enough

- I tried something that I could do
- Create (really steal) a pure TCL module interpreter
- Convert it to work for Lmod.
- Called tcl2lua.tcl

# Where tcl2lua.tcl cam from

- It was the freely available pure TCL env. module code
- This was before Xavier took over Tmod
- It was simple enough to understand and convert.

# TCL is one of my least favorite languages

- Its parser is very line-oriented
- So it is very picky about what goes where
- My intuition about the language is almost always wrong.

# Surprising help from stackoverflow.com

- I have had many questions about how TCL works
- I am always shocked that my TCL question would get answered
- Grateful, but amazed!

# Questions for stackoverflow.com

- How does the puts command work?
- How break works?
- How the child interpreter works

# How tcl2lua.tcl works

- Let the TCL interpreter evaluate regular TCL commands
- Convert the TCL setenv, prepend-path etc
- $\Rightarrow$ into Lua setenv(), prepend_path() strings
- This lua output is then interpreted by Lmod as a Lua module
- This works very well most of the time
- However ...

# When things go awry

- Suppose you have TCL modules **Centos** and **B**

```
Centos::

    #%Module
    setenv SYSTEM_NAME Centos

And B::

    #%Module
    module load Centos

    if  $env(SYSTEM_NAME) == "Centos"
       # do something
```

# Converting the TCL B into Lua

```
load("Centos")
LmodError("can't read ënv(SYSTEM_NAME)̈: no such variable")
```

- Trouble: the TCL **load** command $\Rightarrow$ `load("Centos")`
- Cannot get the TCL load command to be evaluated before the TCL if block

# The best solutions would be:

- In this case use TCL code to get the name of the Linux OS
- Or use lsb_release inside the **B** modulefile
- Or translate the TCL B module into Lua
- Note that you can have both a TCL B/1.0 module **and** a B/1.0.lua module in the same directory
- Lmod will always chose the Lua modulefile over the TCL one.
- Tmod will ignore the *.lua file (no #%Module start line)

# The B translated into Lua

```
load("Centos")
if (os.getenv("SYSTEM_NAME") == "Centos") then
  -- Do something
end
```

- Note that the Centos module DO NOT need to be translated into Lua.

# Next time

- Lmod 8.0+ brought many improvements to TCL module support
- TCL break command
- Optional Integration of TCL interpreter into Lmod (saves time)
- Support for is-loaded
- Support for is-avail and why I resisted supporting that for TCL modulefiles for so long
- Special features of setenv and pushenv in TCL
- How tcl2lua.tcl supports the **puts** command
- New bugs found when integrating the TCL interpreter into Lmod

# Future Topics

- Next Meeting: December 6th 9:30 US Central (15:30 UTC)