15 years of Lmod: A Retrospectve

Robert McLay

August 1, 2023

Outline

- ▶ The start of Lmod
- ▶ Why I used Lua?
- ▶ Where Lmod success comes from?
- ▶ Features added over time
- ▶ Lmod lessons learned
- ▶ Conclusions

TACC

The start of Lmod

- ▶ My friend a colleague Bill Barth asked:
- ▶ TACC has a software hierarchy with Tmod 3.2.10
- ▶ Impossible to change compilers or mpi modules
- ▶ Help?

Key Insight

▶ Remember the state of $MODULEPATH
▶ When it changes, change loaded modules
▶ Support the idea of inactive modules
▶ Lmod was born

Why implement in Lua?

- ▶ TCL is not my favorite language
- ▶ Worked in Lua to develop testing tools (more on this latter)
- ▶ Did consulting work in Python before TACC: Avoid conflicts
- ▶ If useful, the idea would be integrated into Tmod (Nope!)
- ▶ Happy accident, Lua is fast enough
- ▶ Lua is easier to protect from user environments (Esp. Python)

Reasons for Lmod Success

- ▶ Initially no real competition (Tmod 4 started in 2017)
- ▶ Working at TACC helped
- ▶ Good enough documentation
- ▶ Easy transition from Tmod to Lmod (Lmod reads TCL modulefiles)
- ▶ Many Features not provided by tools
- ▶ Unsolicited articles written by Jeff Layton about Lmod
- ▶ Many say: "It just works so I don't worry about it."
- ▶ Used by EasyBuild, OpenHPC and Spack
- ▶ Package for Mac Brew, Fedora, Debian

Hidden Reason for Lmod Success

- ▶ Testing Lmod has an extensive test suite (1400+ tests)
- ▶ It is reliable
- ▶ Can debug (via ml -D ... or ml -T) remotely

More on Testing

▶ The TM testing suite filters output to converts to canoncial names

▶ Path independent

▶ Tests both stderr and stdout output for each test

▶ Can repeatly run a single test file or just the ones that failed

▶ Lmod is reliable because of testing

▶ Github Actions run 3 types of test on 4 version of Lua on Ubuntu and macOS for every check-in.

Building trust with the user community

- ▶ Making it reliable (via Testing)
- ▶ Timely answering the email and github issues
- ▶ Book: Team Geek
- ▶ Learning to be polite when answering and re-answering question
- ▶ "You might consider ..."
- ▶ "Please test Lmod version ... when you get a chance to see if it works for you"
- ▶ Not getting upset when non-native english speakers sound insulting

Features over time

► Tab completion for bash and z-shell
► Support for N/V then C/N/V finally N/V/V (Lmod 7+)
► Module properties
► Spider cache (speed up "module avail" and "module load")
► ml
► sandbox (prevents modulefiles from calling internal routines)

Features part II

- ▶ pushenv, sticky modules, i18n error messages
- ▶ Hooks, /etc/lmod/lmod_config.lua
- ▶ Tracking of module usage via hooks
- ▶ Hidden modules
- ▶ depends_on()
- ▶ source_sh(): source a shell script inside a modulefile
- ▶ LMOD_QUARANTINE_VARS
- ▶ ...

Lmod lesson learned

- ▶ git worktrees
- ▶ Exploiting Lmod to help XALT
- ▶ Learned way more than I ever wanted to know about bash, zsh, tcsh shell startup procedures
- ▶ Want tcsh to die, die, die
- ▶ That default interactive non-login bash shell startup is borked (We patch bash to get to work)
- ▶ Can be difficult to decide what a user is reporting. Bug or not?
- ▶ Getting users to use the bugReport script when submitting a bug.

Lmod lesson learned (II)

▶ That `module() { eval "$(LMOD_CMD bash "$@")" }` works better than

▶ `module() { eval $(LMOD_CMD bash "$@") }` who knew?

▶ Cannot test every possibility, users will **ALWAYS** find a case I missed

▶ The the moduleTable has been incredible notion to store the module state.

▶ That other tools will use the spider cache output.

▶ No two site are run exactly the same way.

▶ There are ten or more ways that Lmod can be tailored.

▶ Communicating changes thru README.new

Conclusions

- ▶ Not every site works like TACC.
- ▶ That making Lmod available to the world has made it so much better.
- ▶ I have made many friends over the years.

Lmod Doc usage by City

1 ▮▮▮ 323
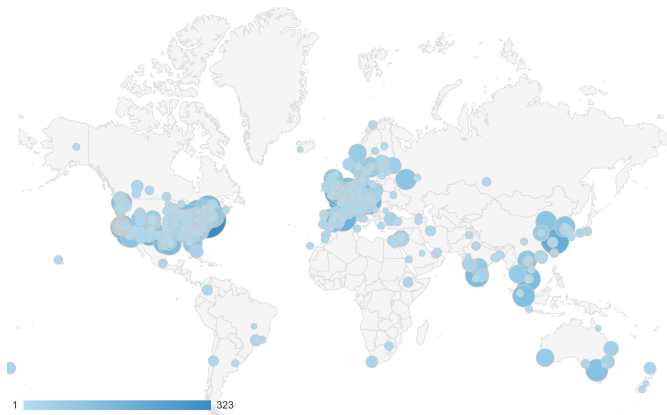
Future Topics

- Next Meeting will be Sept. 5th at 9:30 Central (14:30 UTC)
- Mathew Cawood will be running the meeting with a different zoom link