Modules v4

Pushing forward user environment management

Xavier Delaruelle <xavier.delaruelle@cea.fr>

FOSDEM 2018
February 4th 2018, ULB, Bruxelles
I am Xavier Delaruelle

Work at CEA, a large research institute in France

Started out as an HPC system admin

Now operations manager of a computing center

Environment Modules project leader since July 2017
module in a nutshell

- A shell command
- That changes the environment state of the current shell
- By interpreting Tcl script files (called modulefiles)
- Which load or unload environment configurations

Aim
Give users the ability to handle their environment
Quick example

- **Activate a software**
  
  ```
  $ which gcc
  gcc not found
  $ module load gcc/6.1.1
  $ which gcc
  /apps/gcc/6.1.1/bin/gcc
  ```

- **Check what a modulefile does**
  
  ```
  $ module display gcc/6.1.1
  
  /apps/modfiles/compilers/gcc/6.1.1:
  
  prepend-path PATH /apps/gcc/6.1.1/bin
  ```

- **De-activate a software**
  
  ```
  $ which gcc
  /apps/gcc/6.1.1/bin/gcc
  $ module unload gcc
  $ which gcc
  gcc not found
  ```
Quick example

- **Activate a software**
  
  ```bash
  $ which gcc
  gcc not found
  $ module load gcc/6.1.1
  $ which gcc
  /apps/gcc/6.1.1/bin/gcc
  ```

- **Check what a modulefile does**
  
  ```bash
  $ module display gcc/6.1.1
  /apps/modfiles/compilers/gcc/6.1.1:
  prepend-path PATH /apps/gcc/6.1.1/bin
  ```

- **De-activate a software**
  
  ```bash
  $ which gcc
  /apps/gcc/6.1.1/bin/gcc
  $ module unload gcc
  $ which gcc
  gcc not found
  ```
Quick example

- **Activate a software**
  
  ```
  $ which gcc
  gcc not found
  $ module load gcc/6.1.1
  $ which gcc
  /apps/gcc/6.1.1/bin/gcc
  ```

- **Check what a modulefile does**

  ```
  $ module display gcc/6.1.1
  /apps/modfiles/compilers/gcc/6.1.1:
  prepend-path PATH /apps/gcc/6.1.1/bin
  ```

- **De-activate a software**

  ```
  $ which gcc
  /apps/gcc/6.1.1/bin/gcc
  $ module unload gcc
  $ which gcc
  gcc not found
  ```
Environment Modules project

- 1991: Concept and initial implementation of the module command laid down
- 1999: Modules ported to Linux, version 3.0 written in C
- 2002: Introduction of modulecmd.tcl, pure-Tcl implementation
- 2012: Publication of Modules 3.2.10, last C-version feature release
- 2017: Modules 4.0.0 released, modulecmd.tcl now acts as main module command
Environment Modules project

- 1991: Concept and initial implementation of the module command laid down
- 1999: Modules ported to Linux, version 3.0 written in C
- 2002: Introduction of modulecmd.tcl, pure-Tcl implementation
- 2012: Publication of Modules 3.2.10, last C-version feature release
- 2017: Modules 4.0.0 released, modulecmd.tcl now acts as main module command
The journey to Modules 4.0

- Fix and extend
  - Close most issues known on v3.2
  - modulecmd implementation change (C > Tcl)

- Test, test and test again
  - Extend non-regression test suites (from 250 to >5k tests)
  - Add continuous integration (Travis) [build passing]
  - And monitor code coverage (Codecov) [codecov 95%]
v3.2 > v4.0: New features

- Additional shells supported (fish, lisp, tcl and R)
- Non-zero exit code in case of error
- Output redirect
- Filtering avail output
- Extended support for module alias and symbolic version
- Hiding modulefiles
- Improved modulefiles location
- Module collection
- Path variable element counter
- Optimized I/O operations
- Sourcing modulefiles

v4.0 > v4.1: New features

- Virtual modules
- Extend module command with site-specific Tcl code
- Quarantine mechanism to protect module execution
- Pager support
- Module function to return value in scripting languages
- New modulefile commands (is-saved, is-used, is-avail, module-info loaded)
- New module sub-commands (append-path, prepend-path, remove-path, is-loaded, info-loaded)
- Use variable reference in MODULEPATH

v4.0 > v4.1: New features

- Virtual modules
- Extend module command with site-specific Tcl code
- Quarantine mechanism to protect module execution
- Pager support
- Module function to return value in scripting languages
- New modulefile commands (is-saved, is-used, is-avail, module-info-loaded)
- New module sub-commands (append-path, prepend-path, remove-path, is-loaded, info-loaded)
- Use variable reference in MODULEPATH

Virtual modules

- Associate a module name to a modulefile
- No more need to have a file to get a module available
- Dynamically define available modulefiles depending on the situation
Virtual modules (2)

- **module-virtual** associates a module name to a modulefile:

  ```bash
  $ cat /etc/modfiles/libraries/liba/.modulerc
  #%Module1.0
  module-virtual /1.0 .common
  module-virtual /2.0 .common
  $ cat /etc/modfiles/libraries/liba/.common
  #%Module1.0
  setenv TEST [module-info name]
  ```

- Appears or can be found with its virtual name:

  ```bash
  $ module avail liba
  1) liba/1.0
  $ module load liba/1.0
  $ module list
  Currently Loaded Modulefiles:
  1) liba/1.0
  ```

- The target modulefile is the script interpreted:

  ```bash
  $ echo $TEST
  liba/1.0
  ```
**Virtual modules (2)**

- **module-virtual** associates a module name to a modulefile
  
  ```bash
  $ cat /etc/modfiles/libraries/liba/.modulerc
  #%Module1.0
  module-virtual /1.0 .common
  module-virtual /2.0 .common
  $ cat /etc/modfiles/libraries/liba/.common
  #%Module1.0
  setenv TEST [module-info name]
  ```

- **Appears or can be found with its virtual name.**
  
  ```bash
  $ module avail liba
  """liba/1.0  liba/2.0"
  $ module load liba/1.0
  $ module list
  Currently Loaded Modulefiles:
  1) liba/1.0
  ```

- **The target modulefile is the script interpreted**
  
  ```bash
  $ echo $TEST
  liba/1.0
  ```
Virtual modules (2)

- **module-virtual** associates a module name to a modulefile
  - $ cat /etc/modfiles/libraries/liba/.modulerc
    - #%Module1.0
    - module-virtual /1.0 .common
    - module-virtual /2.0 .common
  - $ cat /etc/modfiles/libraries/liba/.common
    - #%Module1.0
    - setenv TEST [module-info name]

- Appears or can be found with its virtual name.
  - $ module avail liba
  - ----------- /etc/modfiles/libraries -----------
    - liba/1.0   liba/2.0
  - $ module load liba/1.0
  - $ module list
  - Currently Loaded Modulefiles:
    - 1) liba/1.0

- The target modulefile is the script interpreted
  - $ echo $TEST
    - liba/1.0
Roadmap

- A feature release cut every 4 months
- Bug fix releases in-between if necessary

**Next releases**

<table>
<thead>
<tr>
<th>Year</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-05</td>
<td>v4.2.0</td>
</tr>
<tr>
<td>2018-09</td>
<td>v4.3.0 (or v5.0.0)</td>
</tr>
<tr>
<td>2019-01</td>
<td>v5.0.0 (or v5.1.0)</td>
</tr>
</tbody>
</table>
Next releases

- Expectation for v4.2
  - Meta alias or Package
  - Improved conflict and prereq specifications
  - Automatic dependency management between modulefiles

- Expectation for v4.3
  - Available modulefiles cache system
Next releases

- Expectation for v4.2
  - Meta alias or Package
  - Improved conflict and prereq specifications
  - Automatic dependency management between modulefiles

- Expectation for v4.3
  - Available modulefiles cache system
Please load it yourself!

$ module load app
WARNING: app/2.0 cannot be loaded due to missing prereq.
HINT: the following module must be loaded first: liba libb

Should be handled without manual intervention

Leveraging existing prereq modulefile command

$ module load app
load liba/1.0
load libb/2.0
load app/2.0
$ module list
Currently Loaded Modulefiles:
  1) liba/1.0  2) libb/2.0  3) app/2.0
Please load it yourself!

```
$ module load app
WARNING: app/2.0 cannot be loaded due to missing prereq.
HINT: the following module must be loaded first: liba libb
```

Should be handled without manual intervention

Leveraging existing prereq modulefile command

```
$ module load app
load liba/1.0
load libb/2.0
load app/2.0
$ module list
Currently Loaded Modulefiles:
  1) liba/1.0  2) libb/2.0  3) app/2.0
```
Advanced automatic behaviors

- Changing requirement reloads dependent modules
- Unloading a module unloads automatically loaded dependencies
- Automatic unload of dependencies except if they have been explicitly loaded
Conclusion

- Environment Modules project is active again
- With tons of new features to implement
- Stay tuned
Thanks for your attention

- Website: http://modules.sourceforge.net/
- Code: https://github.com/cea-hpc/modules
- Documentation: https://modules.readthedocs.io
- Questions, feedback, new use-cases, want to participate: modules-interest@lists.sourceforge.net