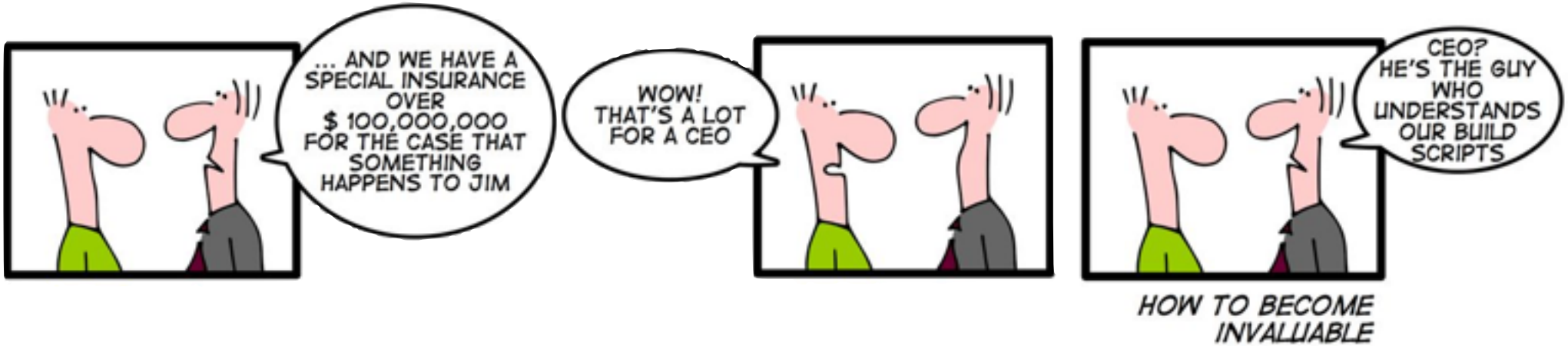
Getting Scientific Software Installed
Tools & Best Practices



http://geekandpoke.typepad.com/geekandpoke/2010/05/how-to-become-invaluable.html

SC'15 Birds-of-a-Feather session
November 17th 2015

Kenneth Hoste (HPC-UGent) - kenneth.hoste@ugent.be

Robert McLay (TACC) - mclay@tacc.utexas.edu

SC'15 BoF session
Getting Scientific Software Installed
Tools & Best Practices
Outline

- lightning talks
  - **Lmod** (Robert McLay, TACC)
  - **EasyBuild** (Ewan Higgs, HPC-UGent)
  - **Spack** (Todd Gamblin, LLNL)
- **show-of-hands** and a couple of key topics
- **open discussion**
  - what are the major issues (for you)?
  - which tools are you using, and would you recommend?

Show of hands (setup)

- Go to **socrative.com**

  (use your laptop, smartphone, tablet, ...)
- Click ‘Student Login’
- Enter the room number: **570181**
- Participate!

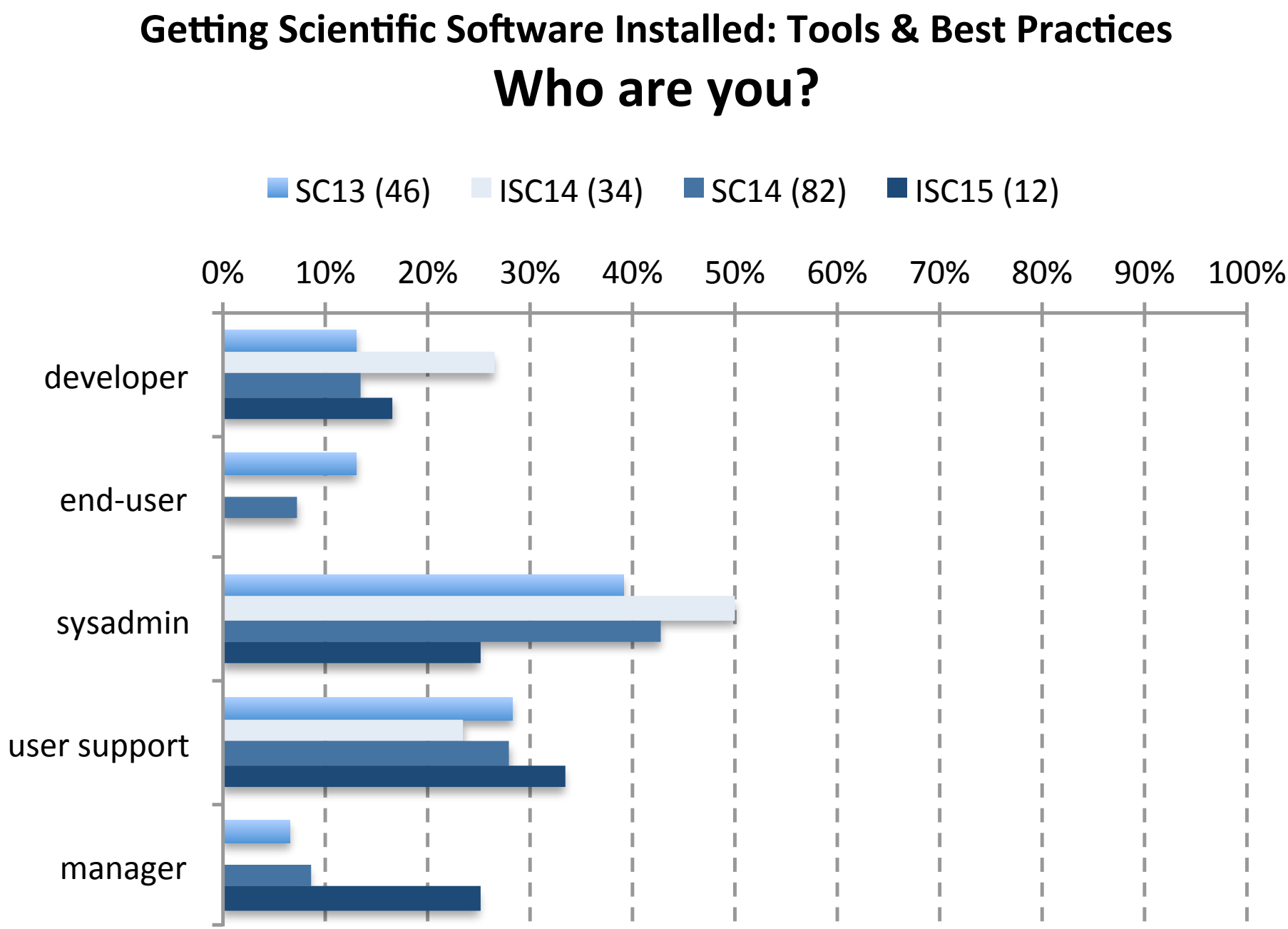
Who are you?

socrative.com
student login: 570181

- scientific software developer
- researcher / end-user of scientific software
- system administrator
- member of user support team
- manager
- other?

Who are you?

socrative.com
student login: 570181



Which modules tool do you use?

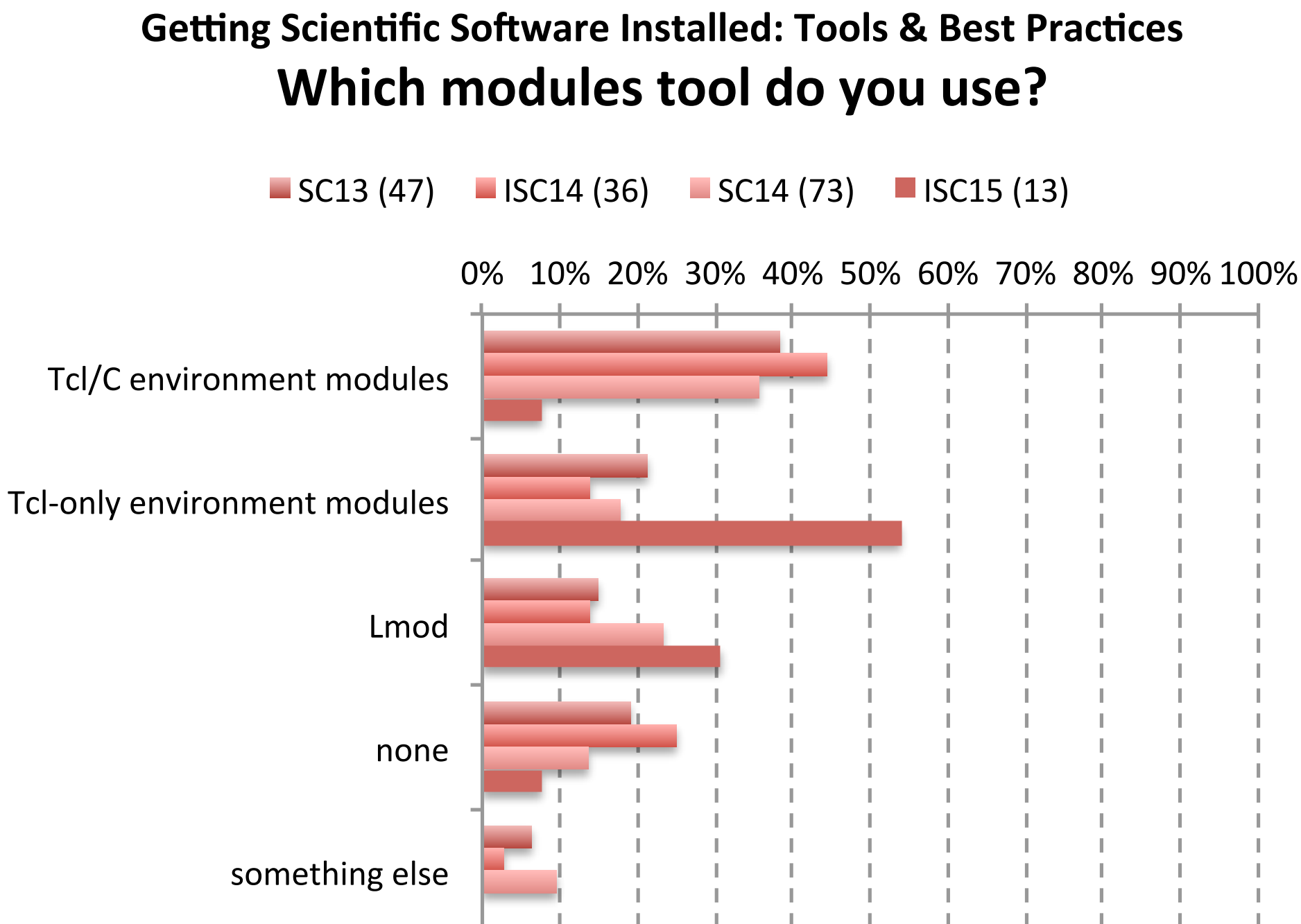
socrative.com
student login: 570181

(tip: if you're not sure, check the output of "type module" or "which module")

- Tcl/C environment modules
  - 'modulecmd' command
- Tcl environment modules
  - 'modulecmd.tcl' script
- **Lmod**
- no modules tool
- something else?

Which modules tool do you use?

socrative.com
student login: 570181



Module naming scheme?

socrative.com
student login: 570181

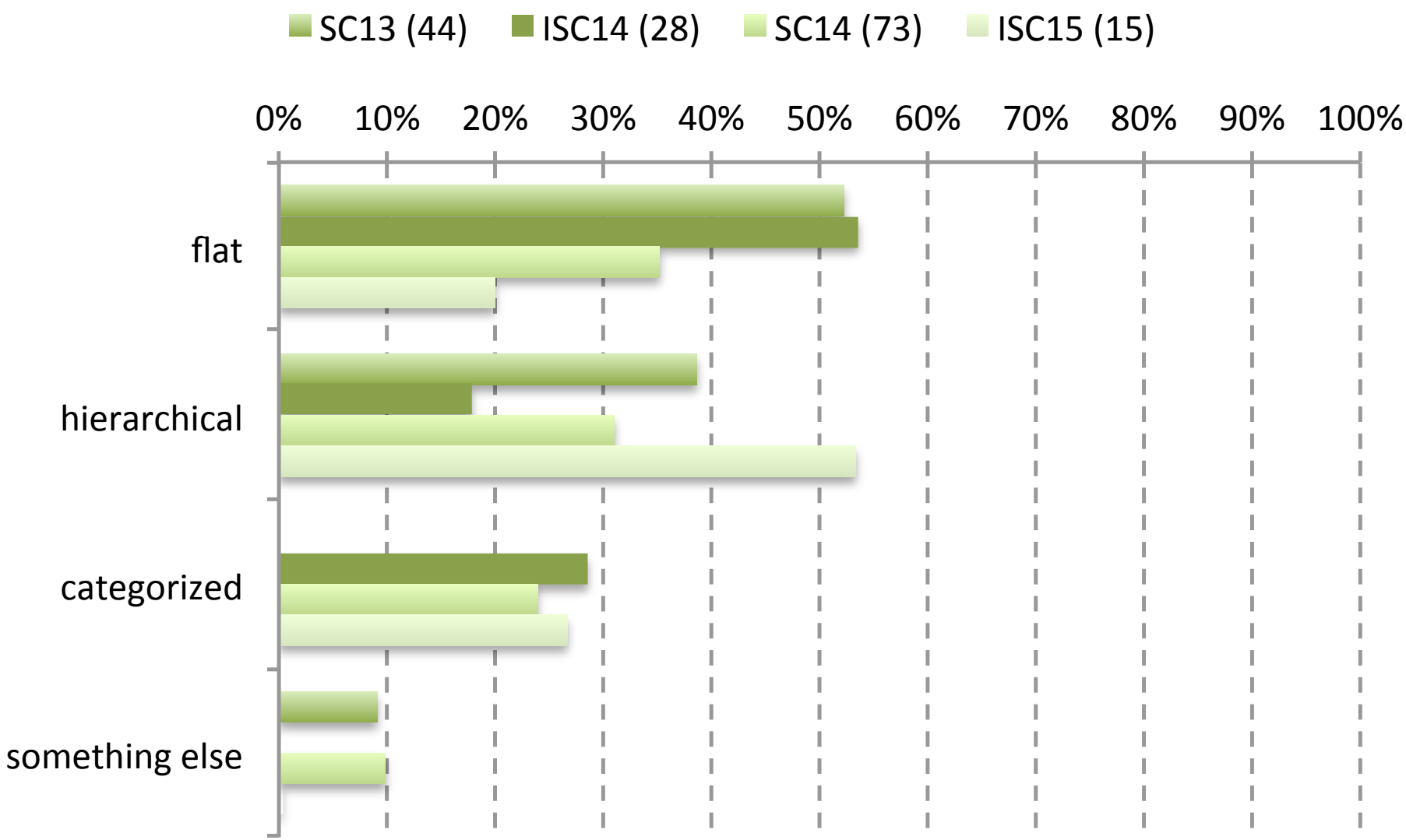
- flat scheme
  - ‘module list’ show *all* the available modules
- flat scheme, with modules grouped by category
- **hierarchical / tree scheme**
  - ‘module list’ only shows compilers
  - ‘module load <compiler>’ first
  - then ‘module load <MPI>’, ‘module load <software>’
- something else?

Module naming scheme?

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

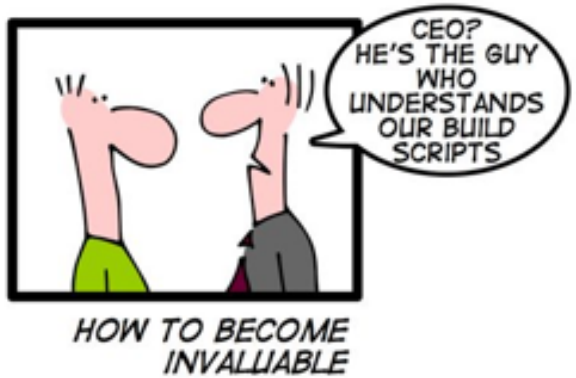
Which modules naming scheme do you use?



Tools for installing scientific software

socrative.com
student login: 570181

- self-written scripts (of any form)
- existing build tool(s)
  - **EasyBuild**, **Spack**, Portage, Ports, HomeBrew, ...
- packages
  - RPMs, .deb, ...
- well-documented build procedures
- 'that guy' (Jim)



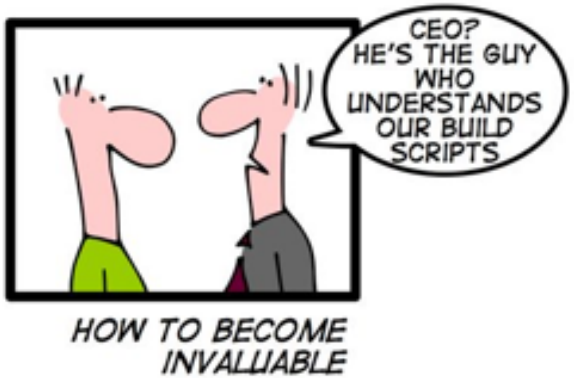
http://geekandpoke.typepad.com/geekandpoke/2010/05/how-to-become-invaluable.html

socrative.com
student login: 570181

Tools for installing scientific software

socrative.com
student login: 570181

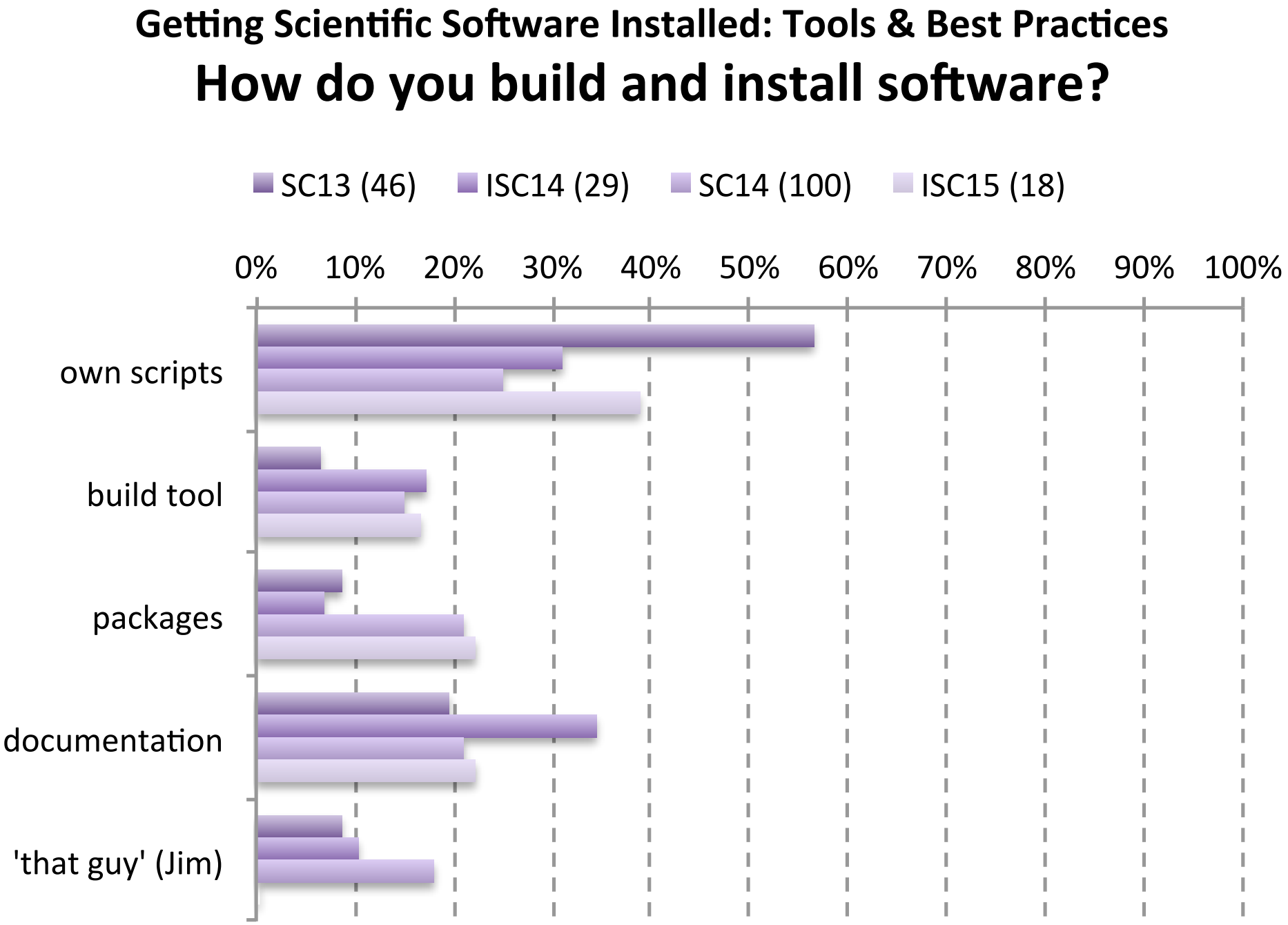
- self-written scripts (of any form)
- existing build tool(s)
  - **EasyBuild**, **Spack**, Portage, Ports, HomeBrew, ...
- packages
  - RPMs, .deb, ...
- well-documented build procedures
- ‘that guy’ (Jim)



socrative.com
student login: 570181

Tools for installing scientific software

socrative.com
student login: 570181



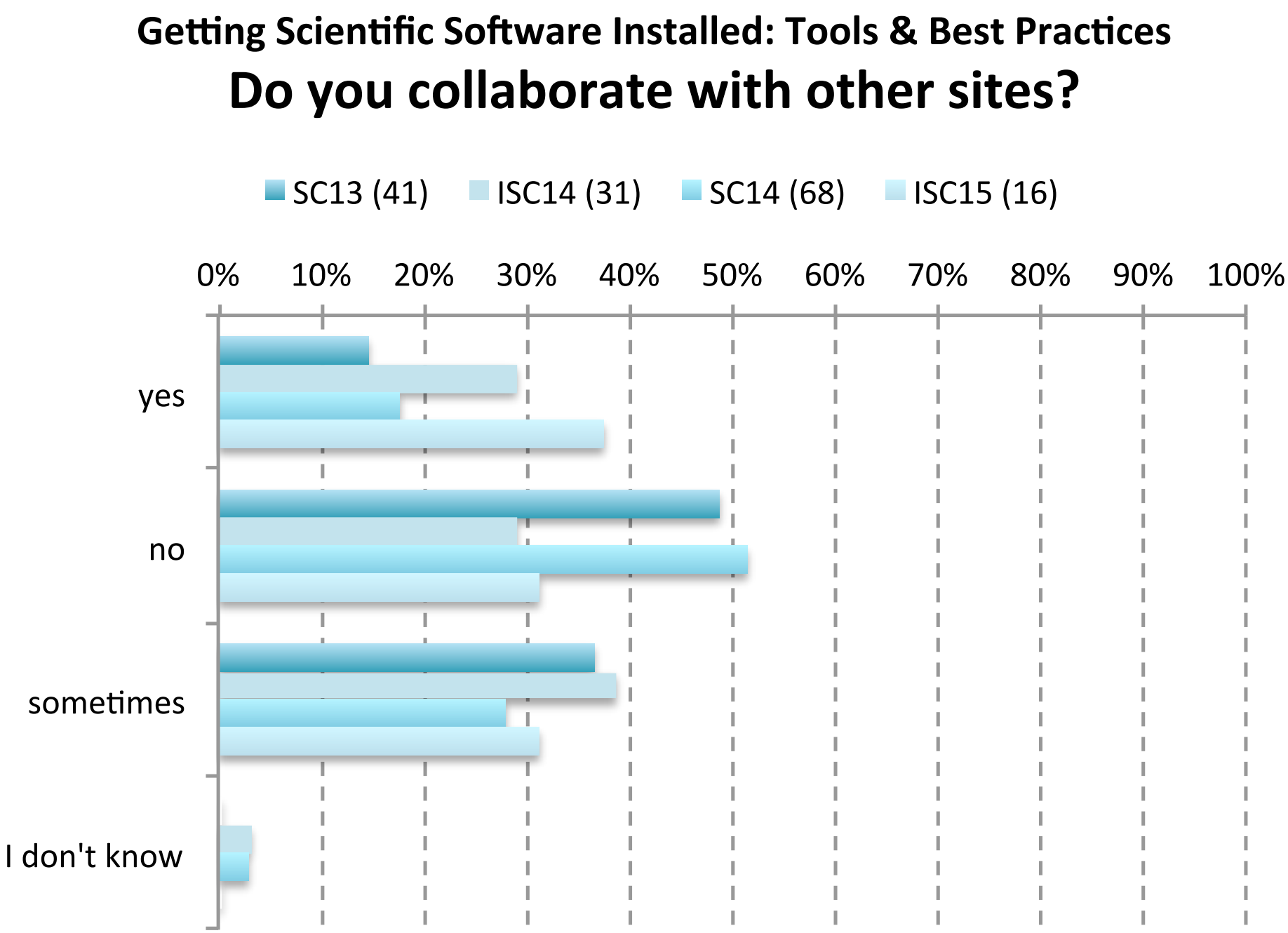
Best practices

socrative.com
student login: 570181

- **collaboration with other HPC sites** (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181



Best practices

socrative.com
student login: 570181

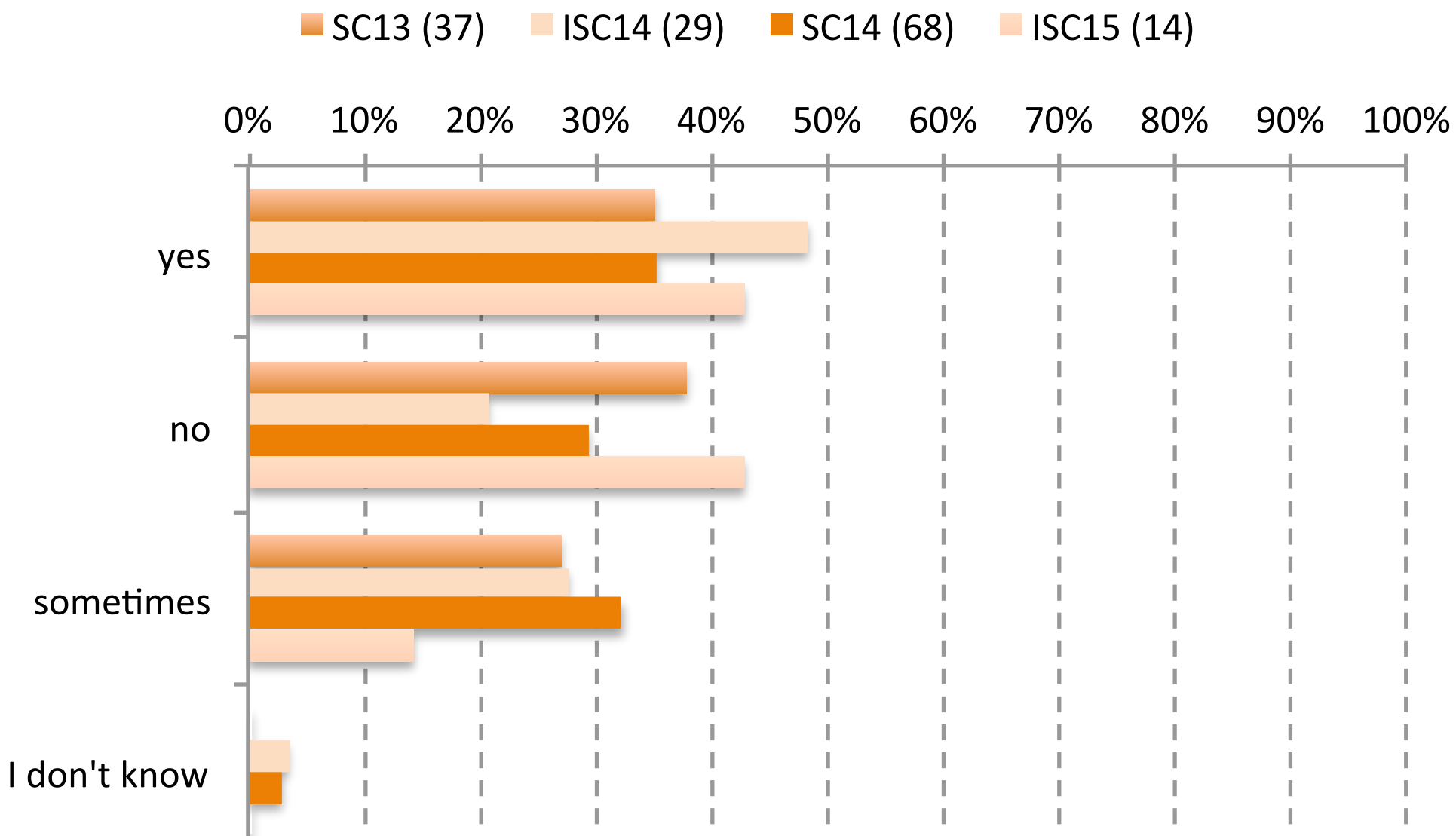
- collaboration with other HPC sites (w.r.t. installing scientific software)
- **automation of builds**
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Do you automate software builds?



Best practices

socrative.com
student login: 570181

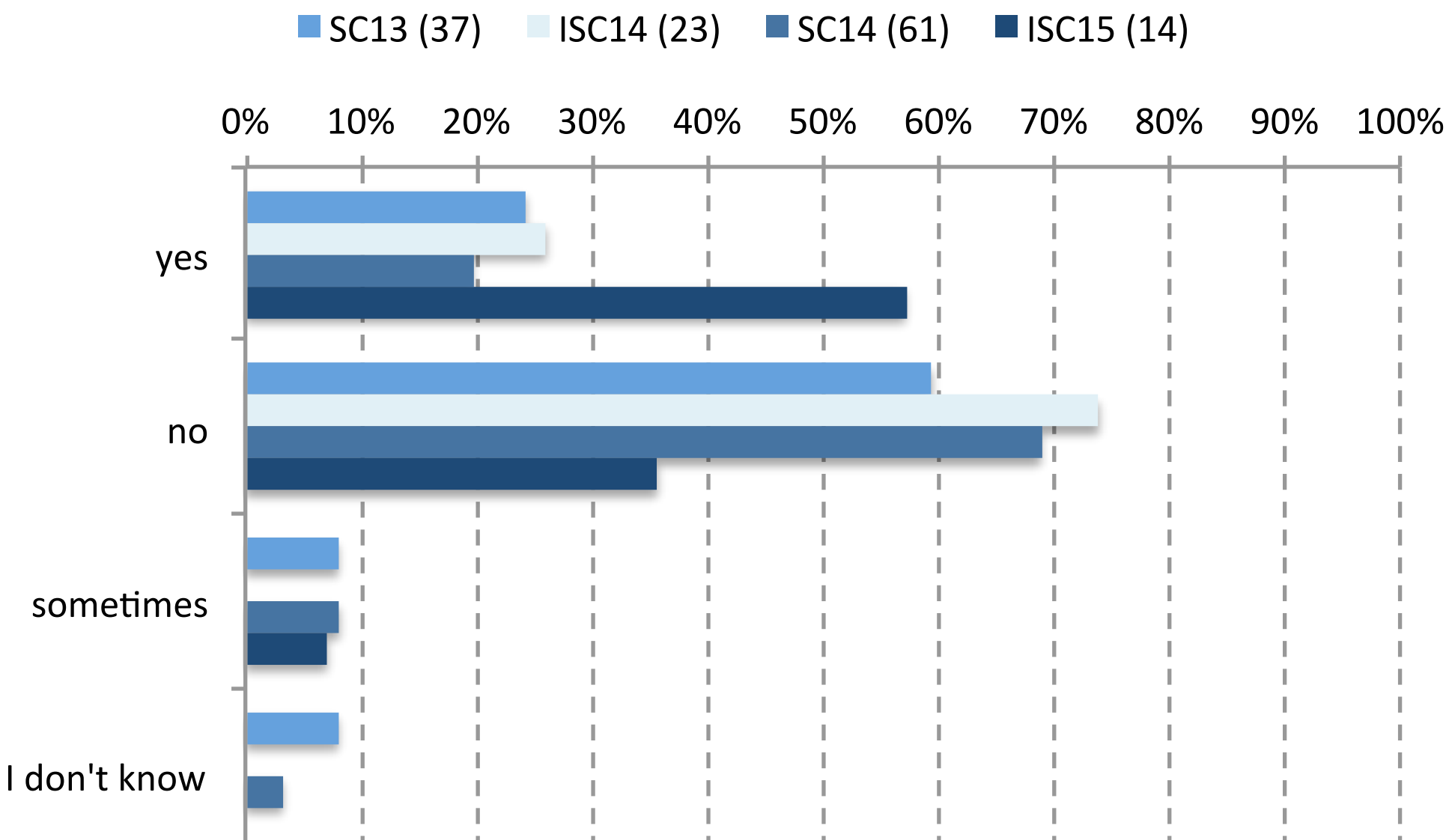
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- **auto-generated module files**
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Are module files generated automatically?



Best practices

socrative.com
student login: 570181

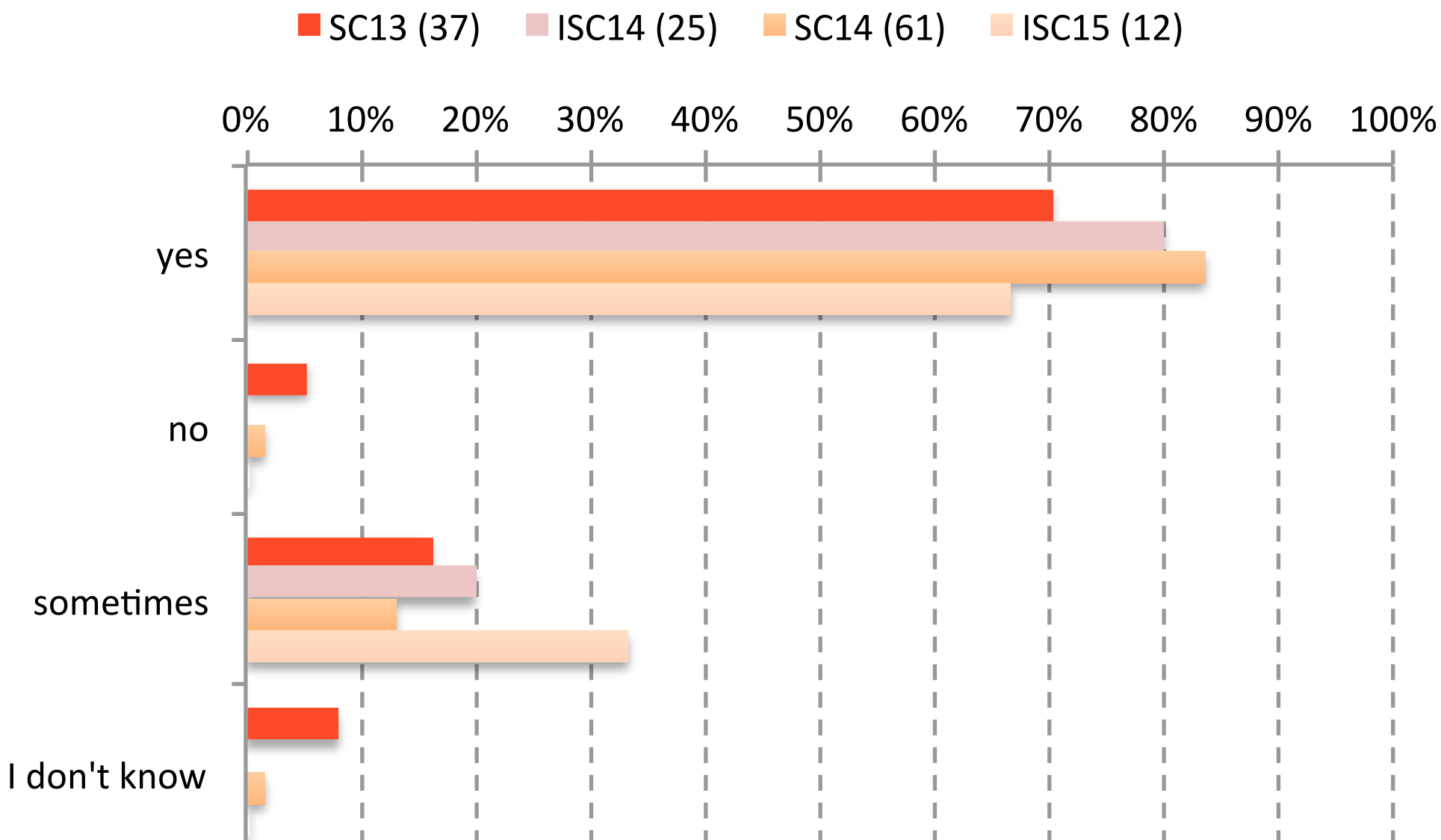
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- **providing multiple builds of the same software**
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Do you provide multiple builds per application?



Best practices

socrative.com
student login: 570181

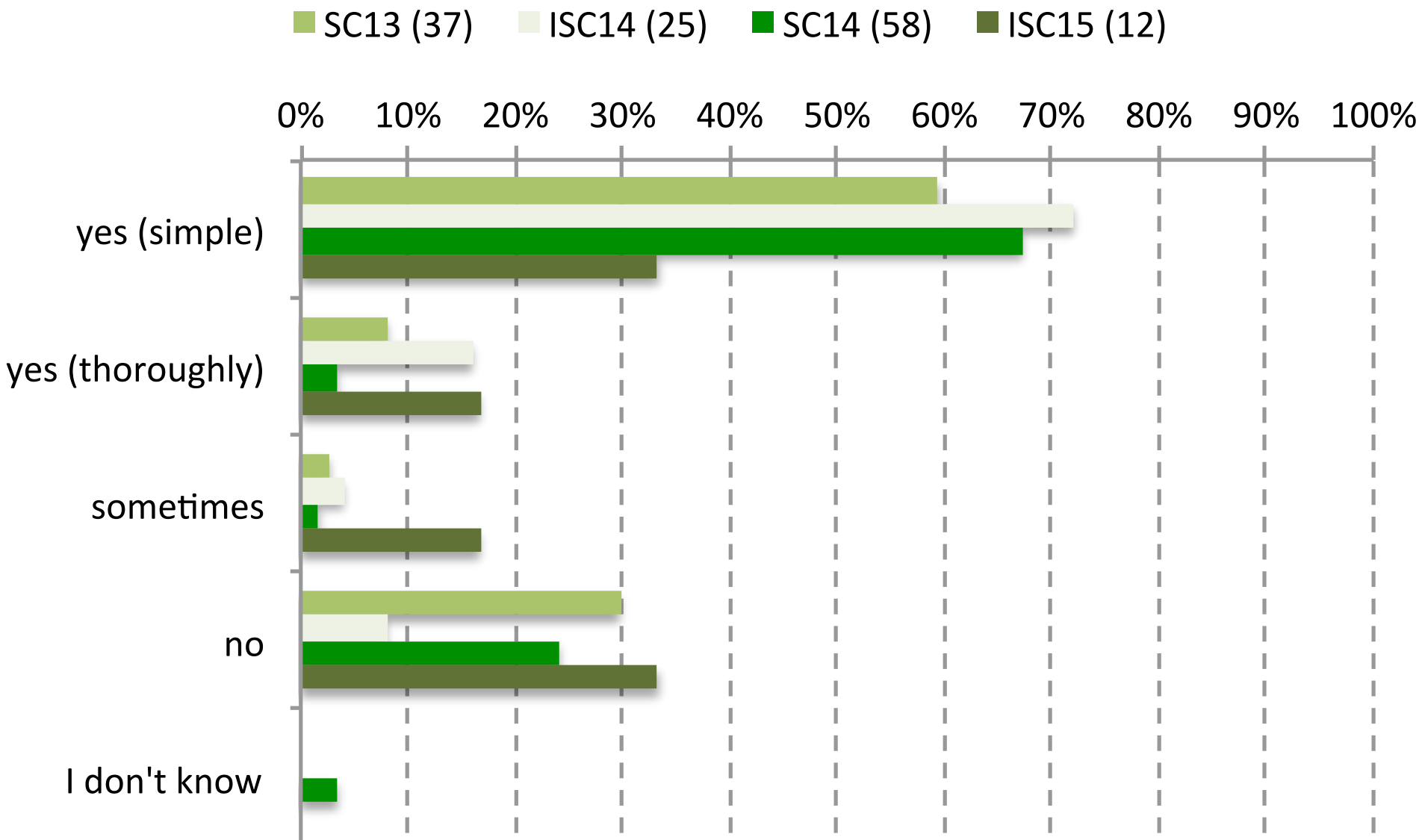
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- **testing of the software installation**
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Do you test the software builds (correctness)?



Best practices

socrative.com
student login: 570181

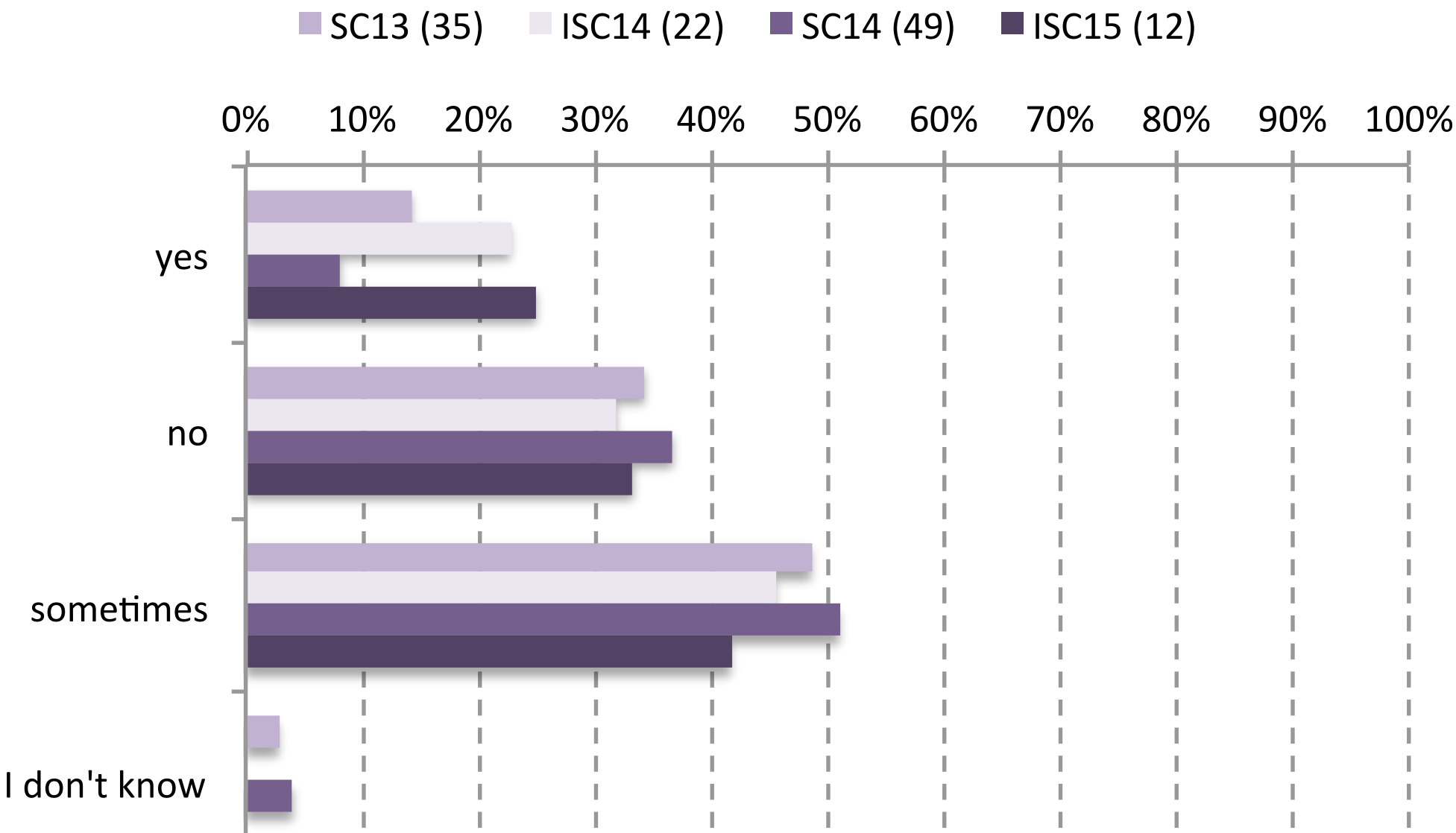
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- **performance evaluation** (post-build)**,** performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Do you evaluate performance of the software?



Best practices

socrative.com
student login: 570181

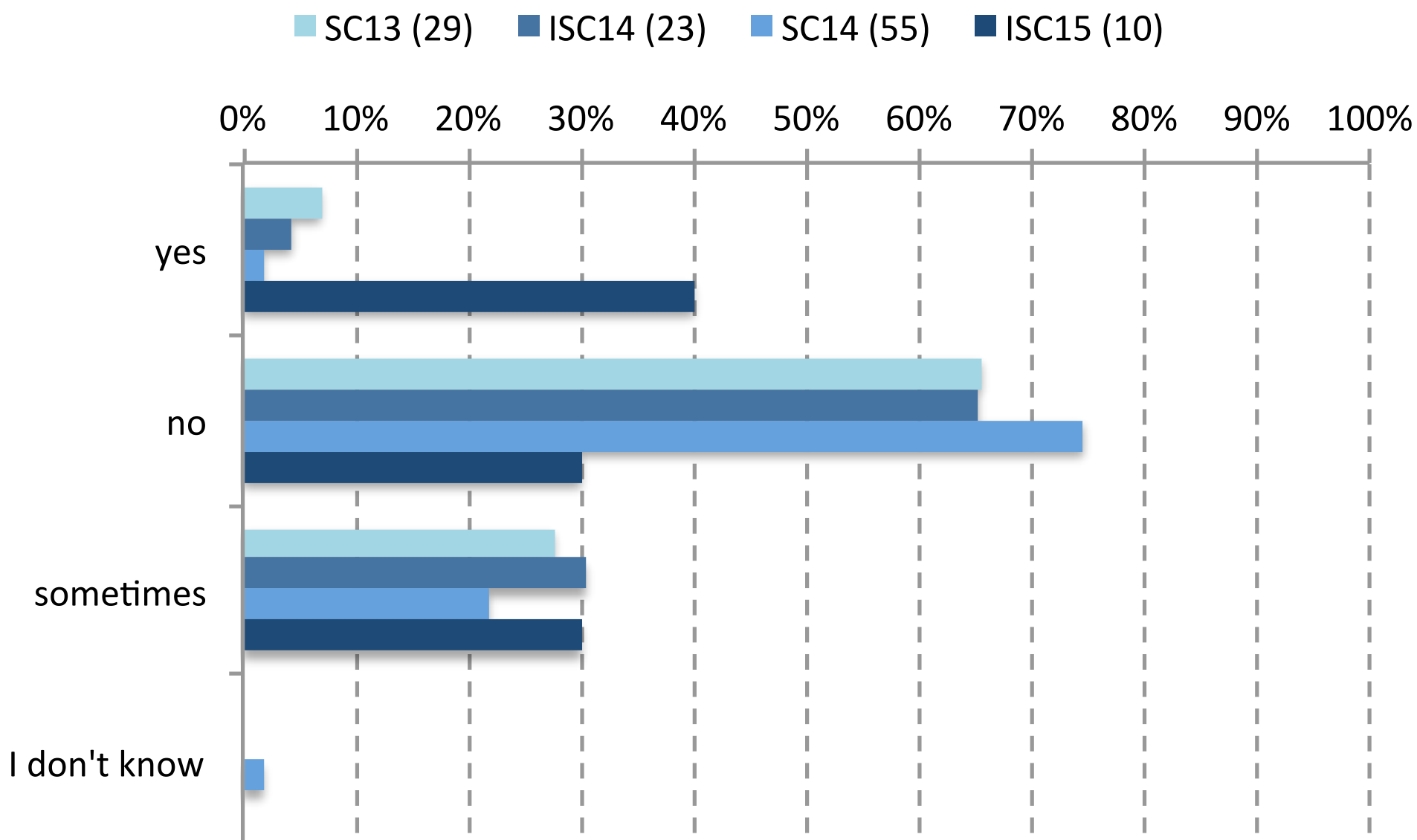
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build)**, performance monitoring** (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Do you monitor performance of the software?



Best practices

socrative.com
student login: 570181

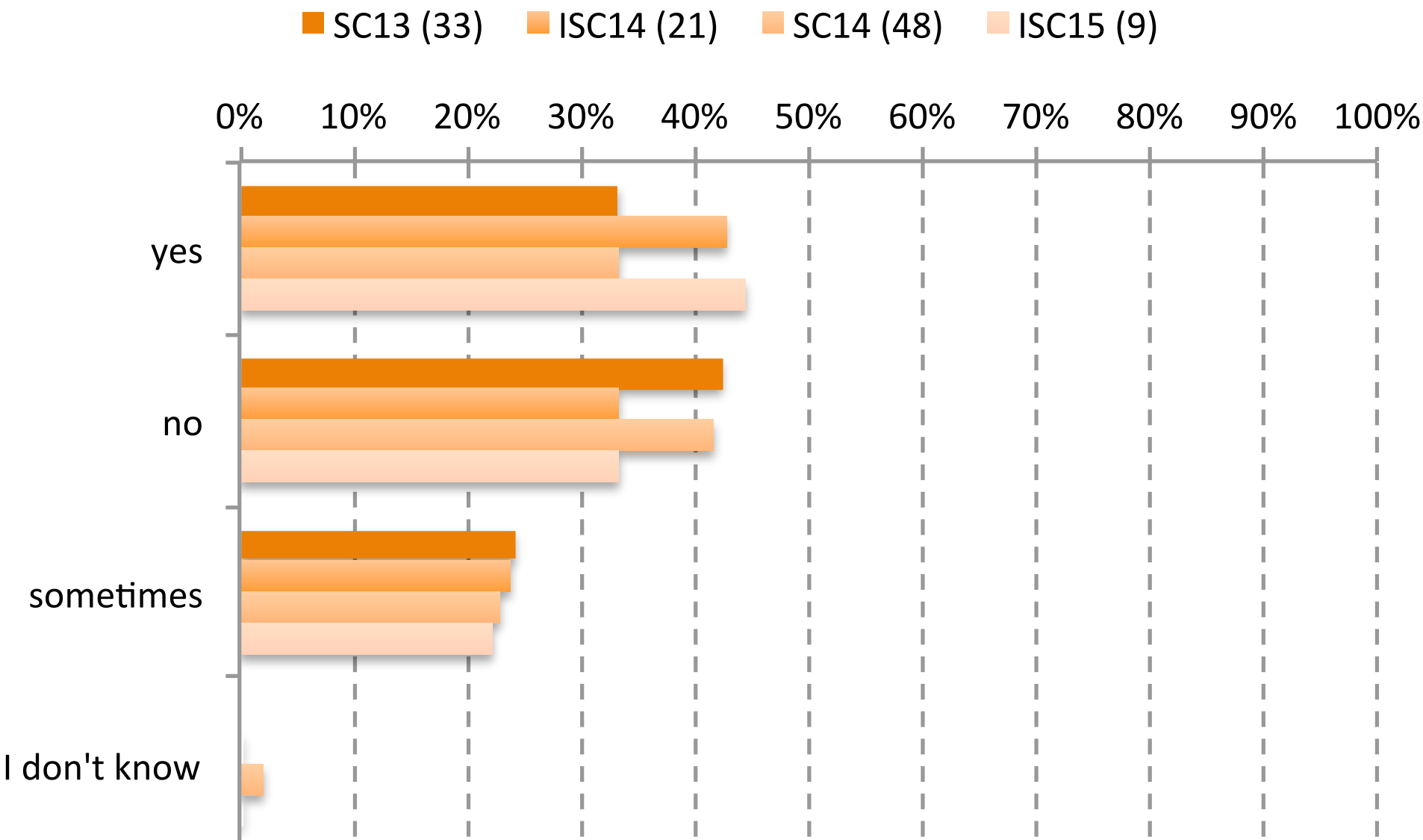
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- **keeping track of build 'metadata'**
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- keeping archive of source/installation files
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181

Getting Scientific Software Installed: Tools & Best Practices

Do you keep track of build metadata?



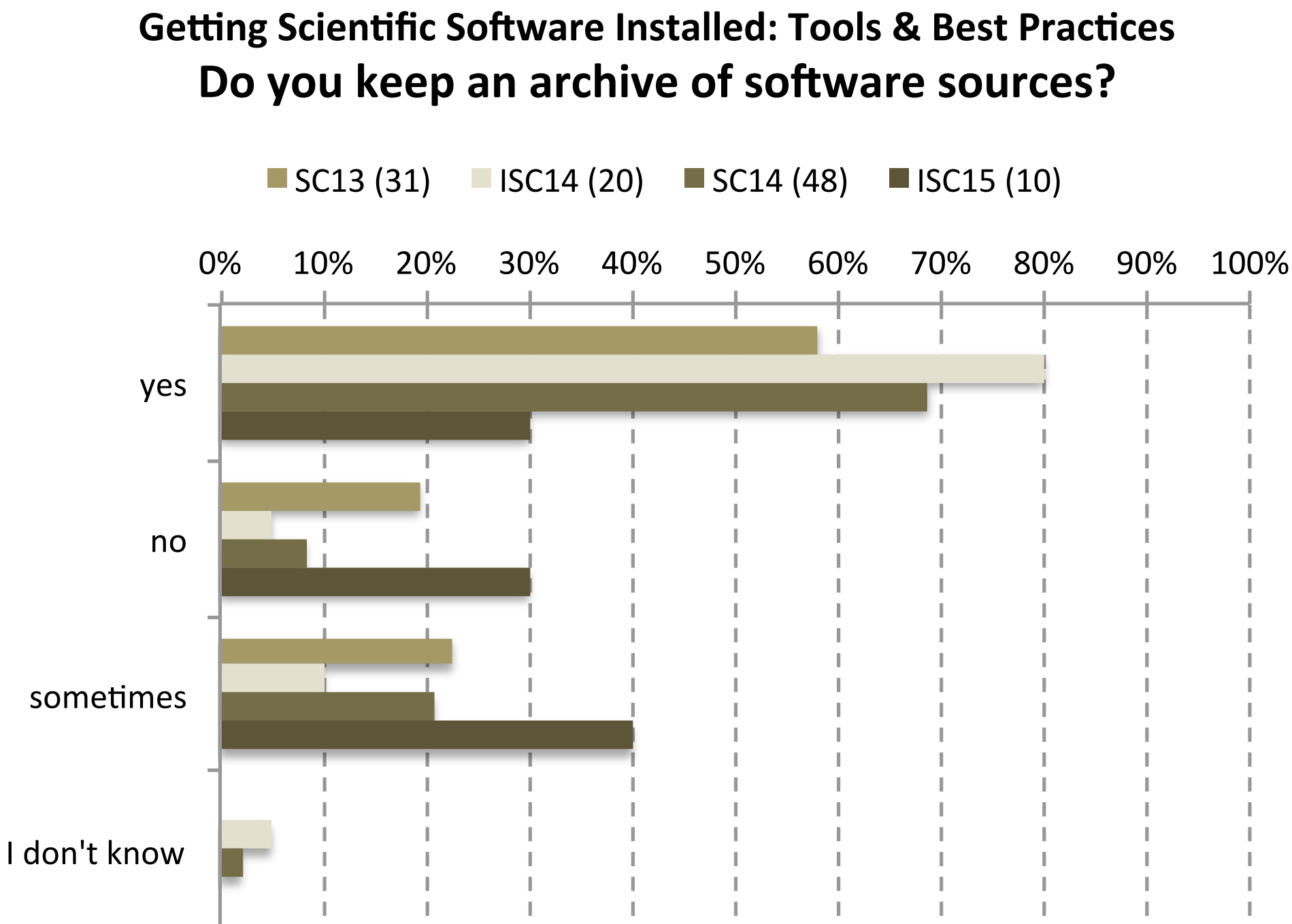
Best practices

socrative.com
student login: 570181

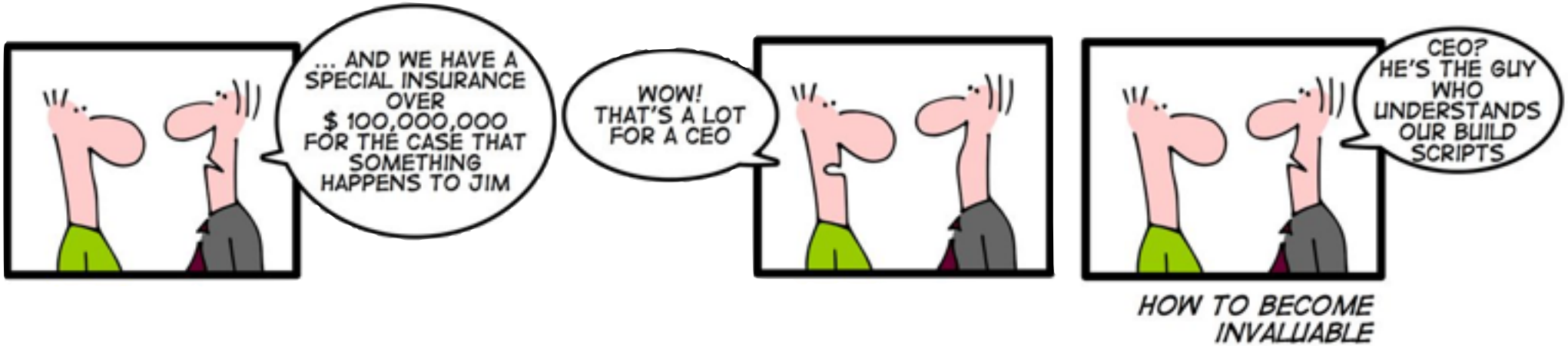
- collaboration with other HPC sites (w.r.t. installing scientific software)
- automation of builds
  - in some way or another (not Jim)
- auto-generated module files
- providing multiple builds of the same software
  - different versions, building with different compilers/MPI libraries
- testing of the software installation
  - simple: make sure everything is there (binaries, libraries, header files, ...)
  - thoroughly: using well-defined tests, verification of results, ...
- performance evaluation (post-build), performance monitoring (over time)
- keeping track of build ‘metadata’
  - build procedure, build log, patch files, build time, built by, dependencies, ...
- **keeping archive of source/installation files**
  - to remedy disappearing upstream sources

Best practices

socrative.com
student login: 570181



Getting Scientific Software Installed
Tools & Best Practices



http://geekandpoke.typepad.com/geekandpoke/2010/05/how-to-become-invaluable.html

SC'15 Birds-of-a-Feather session
November 17th 2015

Kenneth Hoste (HPC-UGent) - kenneth.hoste@ugent.be

Robert McLay (TACC) - mclay@tacc.utexas.edu