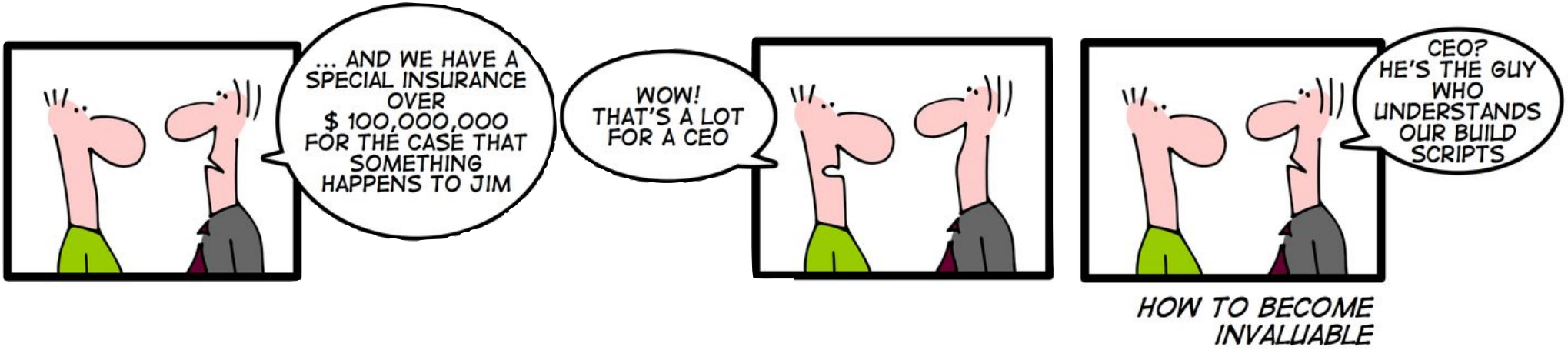
Getting Scientific Software Installed
Tools & Best Practices



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

SC 14 Bird-of-a-Feather session

19 November 2014

andy.georges@ugent.be and jens.timmerman@ugent.be

HPC-UGent in a nutshell

- Central ICT department team at UGent (Belgium)
  - 9 members: 1 manager, ~3 user support staff, ~5 sysadmins
  - tasks: hardware, system administration, software deployment, user support, training, …
- Varied machine park
  - 6 Tier-2 clusters: 34 - 170 nodes, 8 - 32 cores, 2 - 4 GiB/core
  - 1 Tier-1 cluster: 528 nodes, 16 cores, 4 GiB/core
  - Total over 1000 worker nodes
- Varied user base with +1400 accounts, wide range of research domains
- Flemish Supercomputer Centre (VSC) member
  - virtual as a collaboration between Flemish university associations

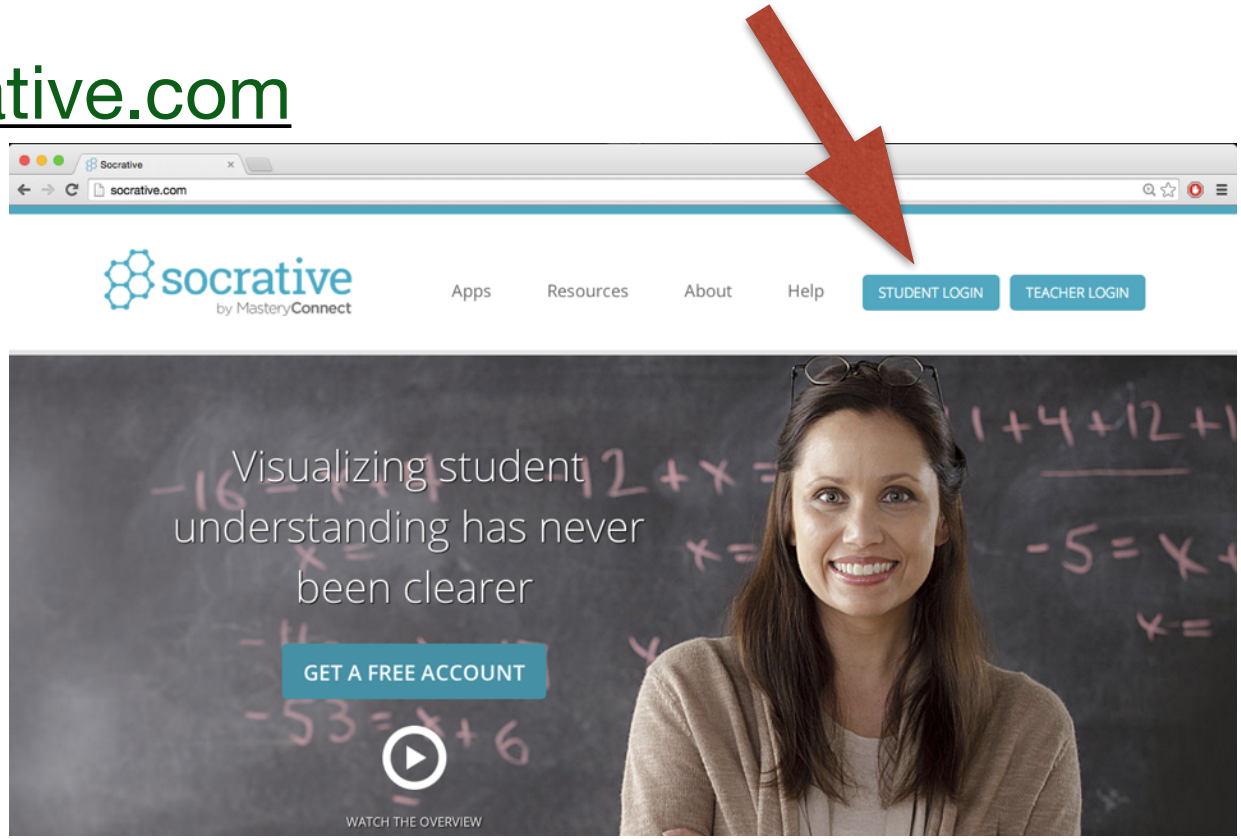
UNIVERSITEIT GENT

Today's outline

- Lightning talks (5 minutes each)
  - Spack: a flexible package manager for HPC— Todd Gamblin (LLNL)
  - Lmod: A Modern Environment Module System— Robert McLay (TACC)
  - EasyBuild — Jens Timmerman (UGent)
- Show of hands for a couple key topics
- Open discussion
  - What are major issues do(es) you(your team) have?
  - Which tools are you using and would you recommend them?

Show of hands - setup

- Visit http://socrative.com
  - use your laptop, phone, tablet, …



- Choose ‘Student Login’
- Room number 570181
- Participate!
- First question to see if things are working properly.

UNIVERSITEIT GENT

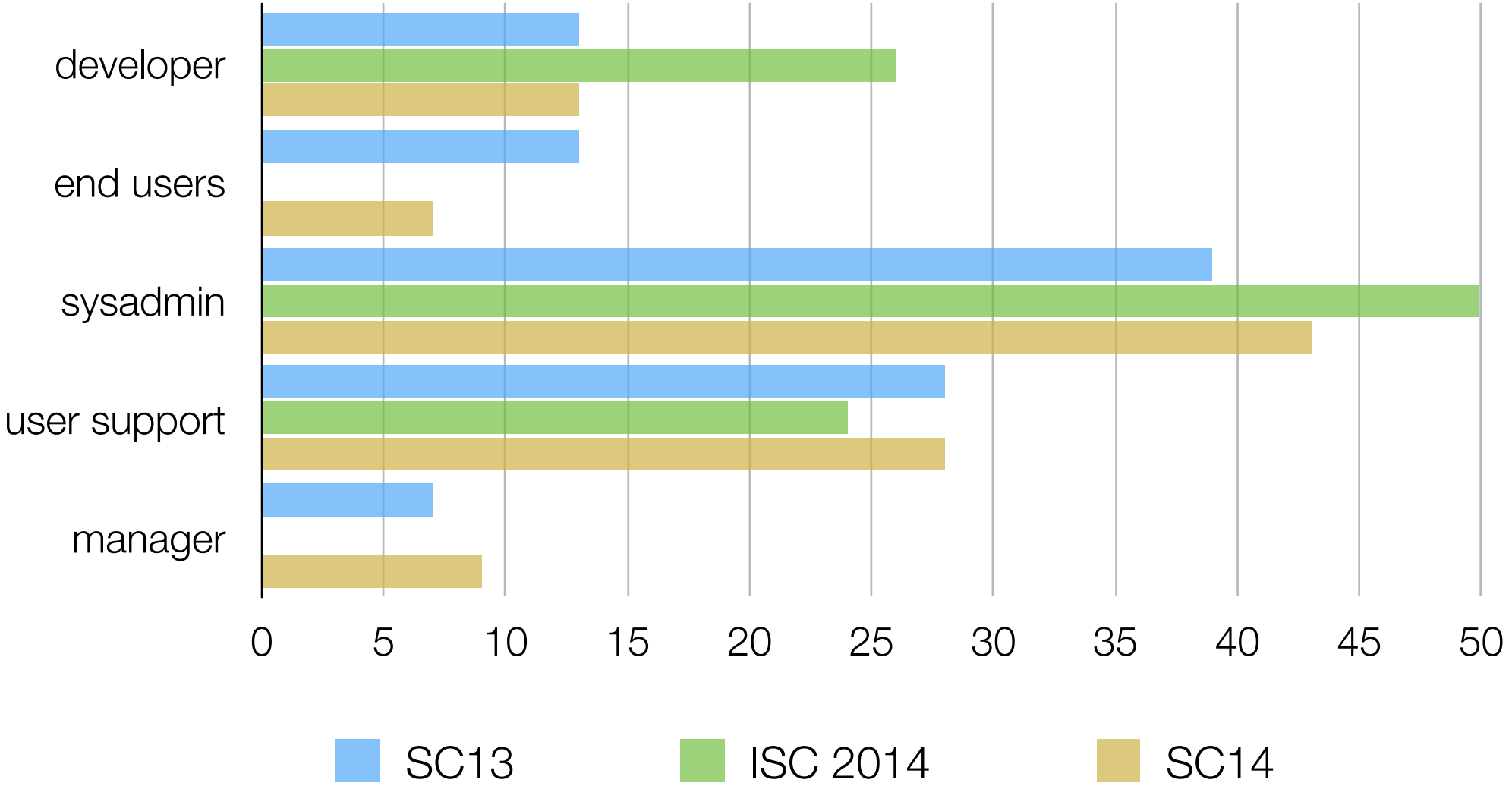
socrative.com - 570181

Who are you?

- scientific software developer
- researcher or end-user of scientific software
- system administrator
- user support team member
- manager
- other, i.e., none of the above

socrative.com - 570181

Who are you?



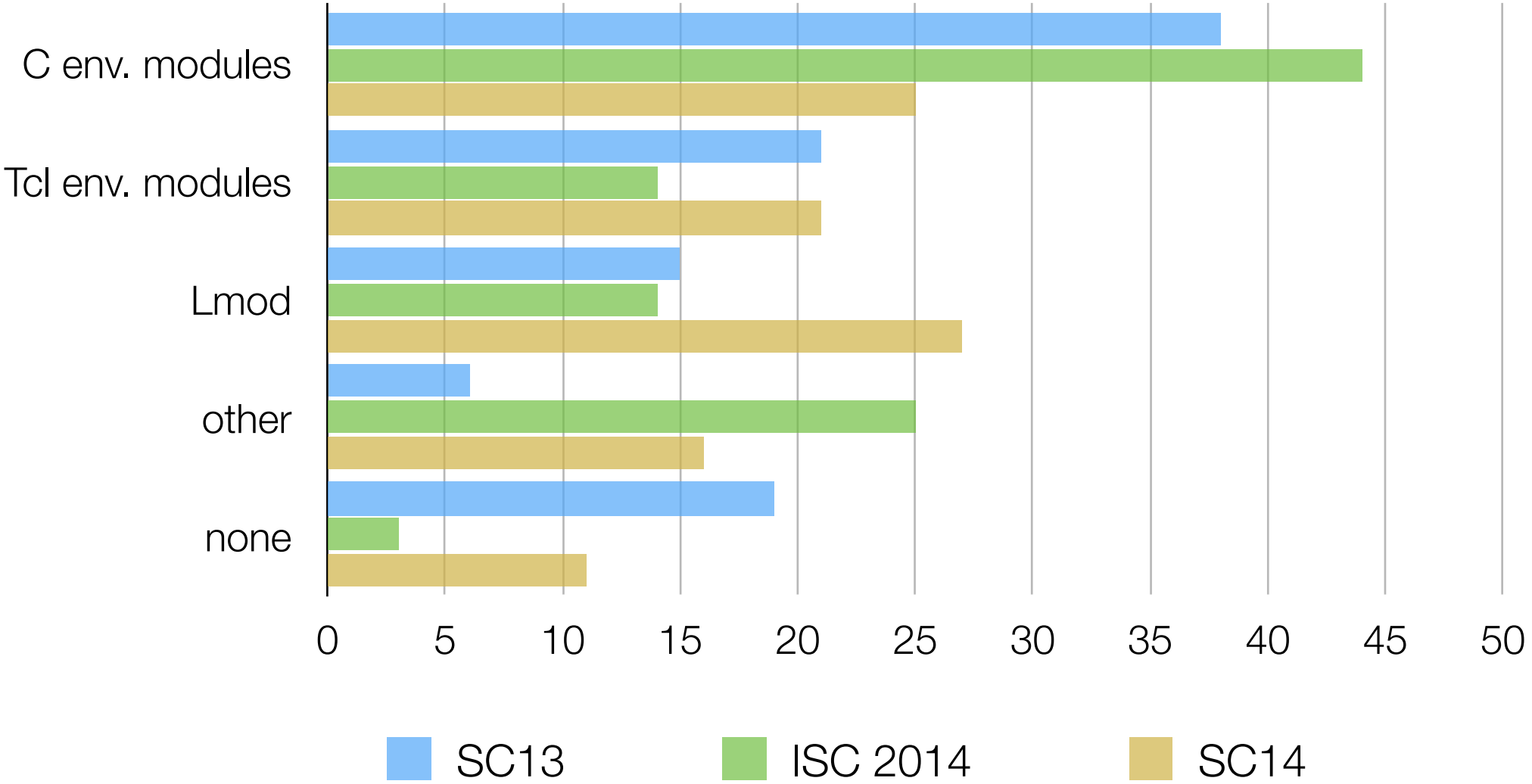
Which modules tool do you use?

- Tcl/C environment modules (uses `modulecmd`)
- Tcl environment modules (uses `modulecmd.tcl`)
- Lmod
- no modules tool
- something else entirely

If you are in doubt, check the output of `which module` or `type module`

socrative.com - 570181

Which modules tools do you use?

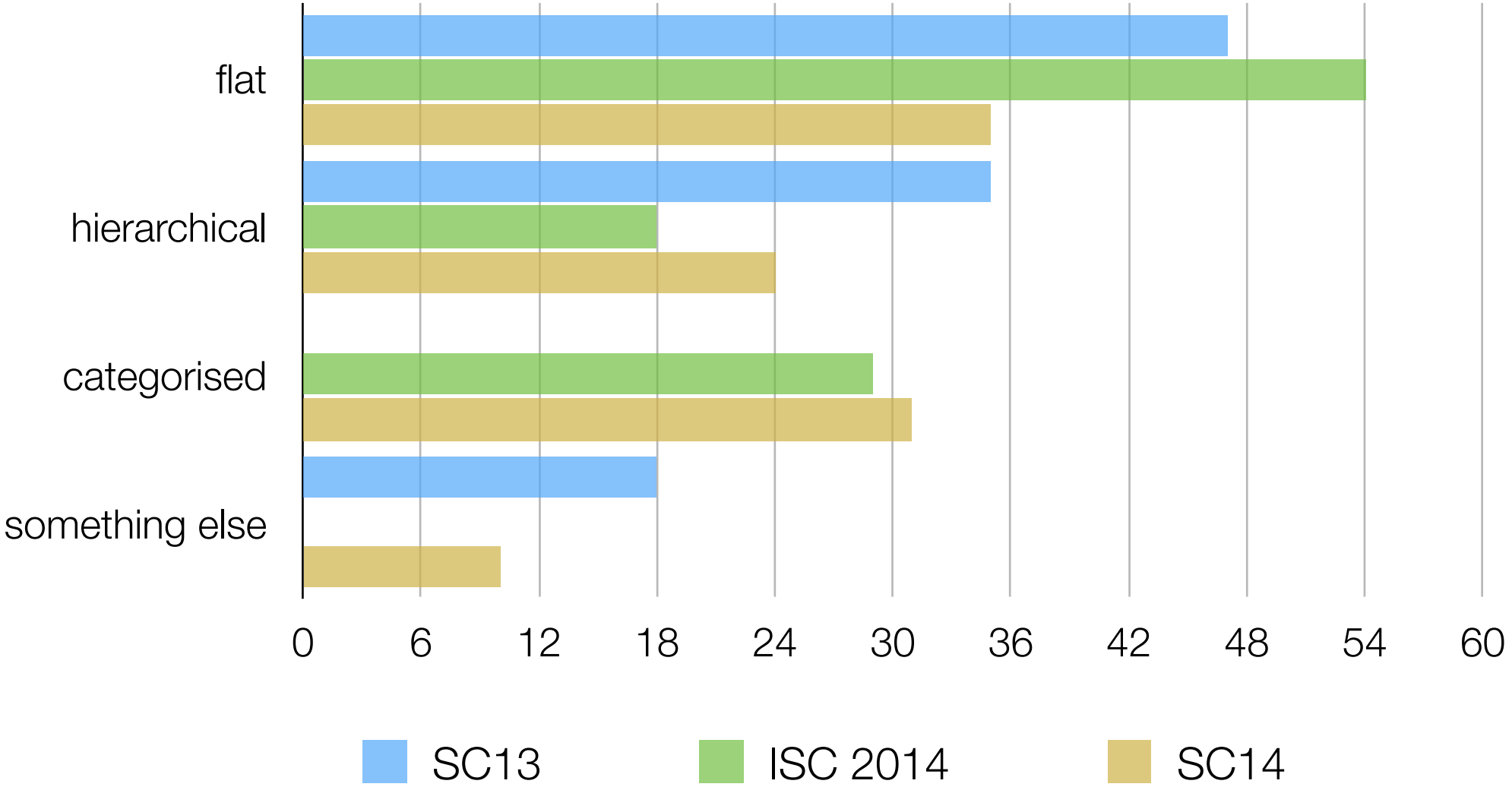


Which module naming scheme do you use?

- a flat scheme: `module list` shows all the modules
- a flat scheme with modules grouped per category
- a hierachical or tree scheme, e.g.,
  - module list shows only compilers
  - `module load <compiler>` first
  - `module load <MPI>` second
  - `module load <software>` last
- something else?

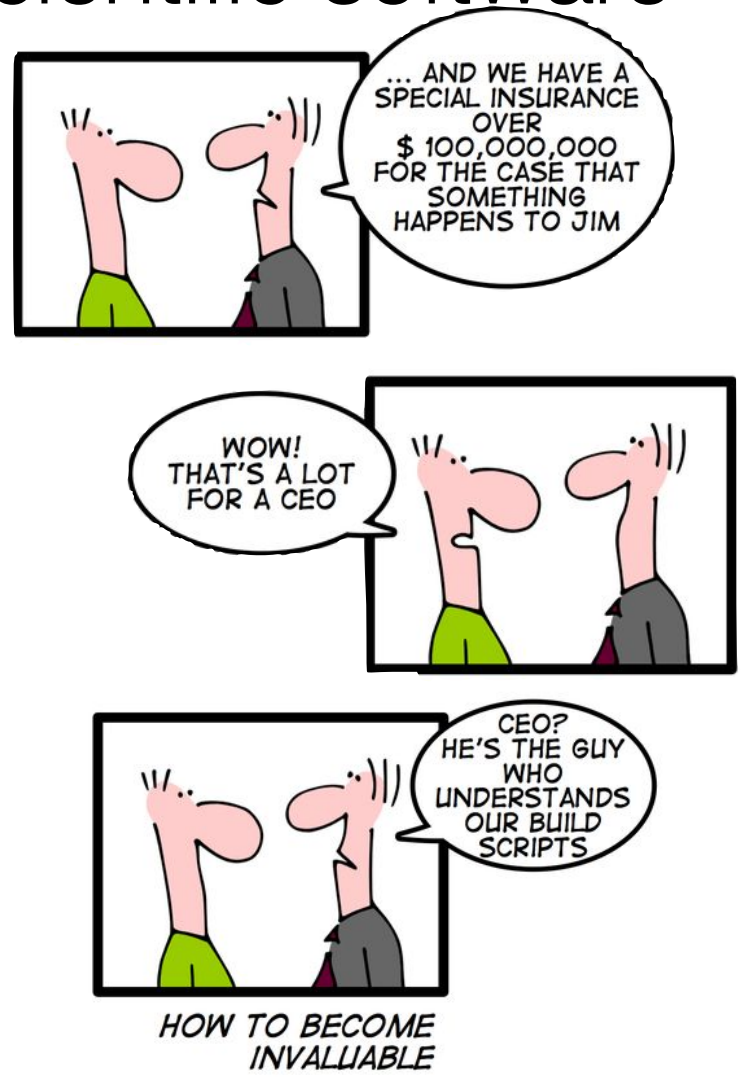
socrative.com - 570181

Using which modules naming scheme?



Tools for building/installing scientific software

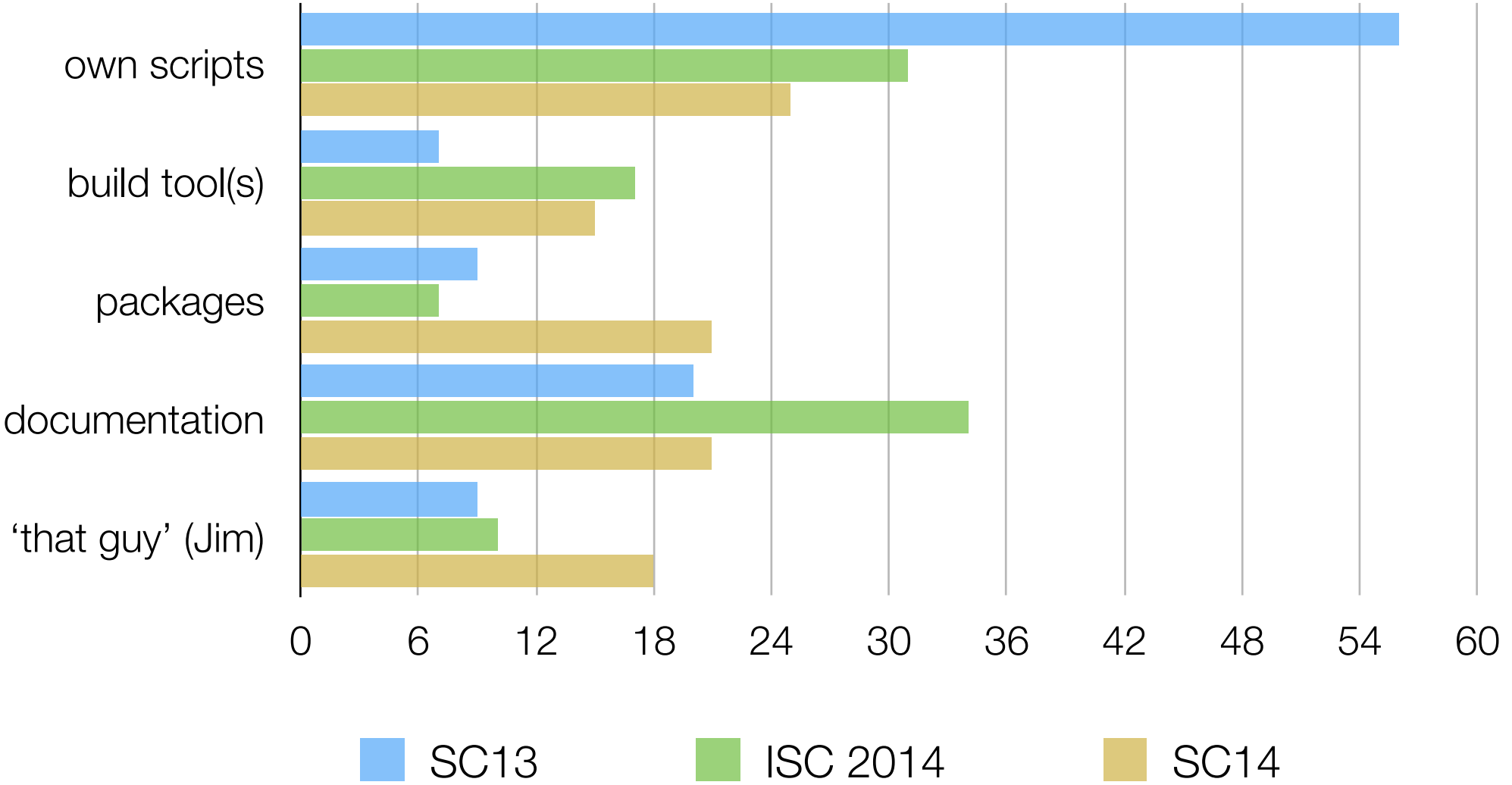
- self written scripts in any form
- existing build tools (homebrew, portage, ports, Spack, EasyBuild, iBS, nix, …)
- packages (RPM, .deb, …)
- well documented build procedures
- ‘that guy’ (Jim)
- other?



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

socrative.com - 570181

How do you build and install software?

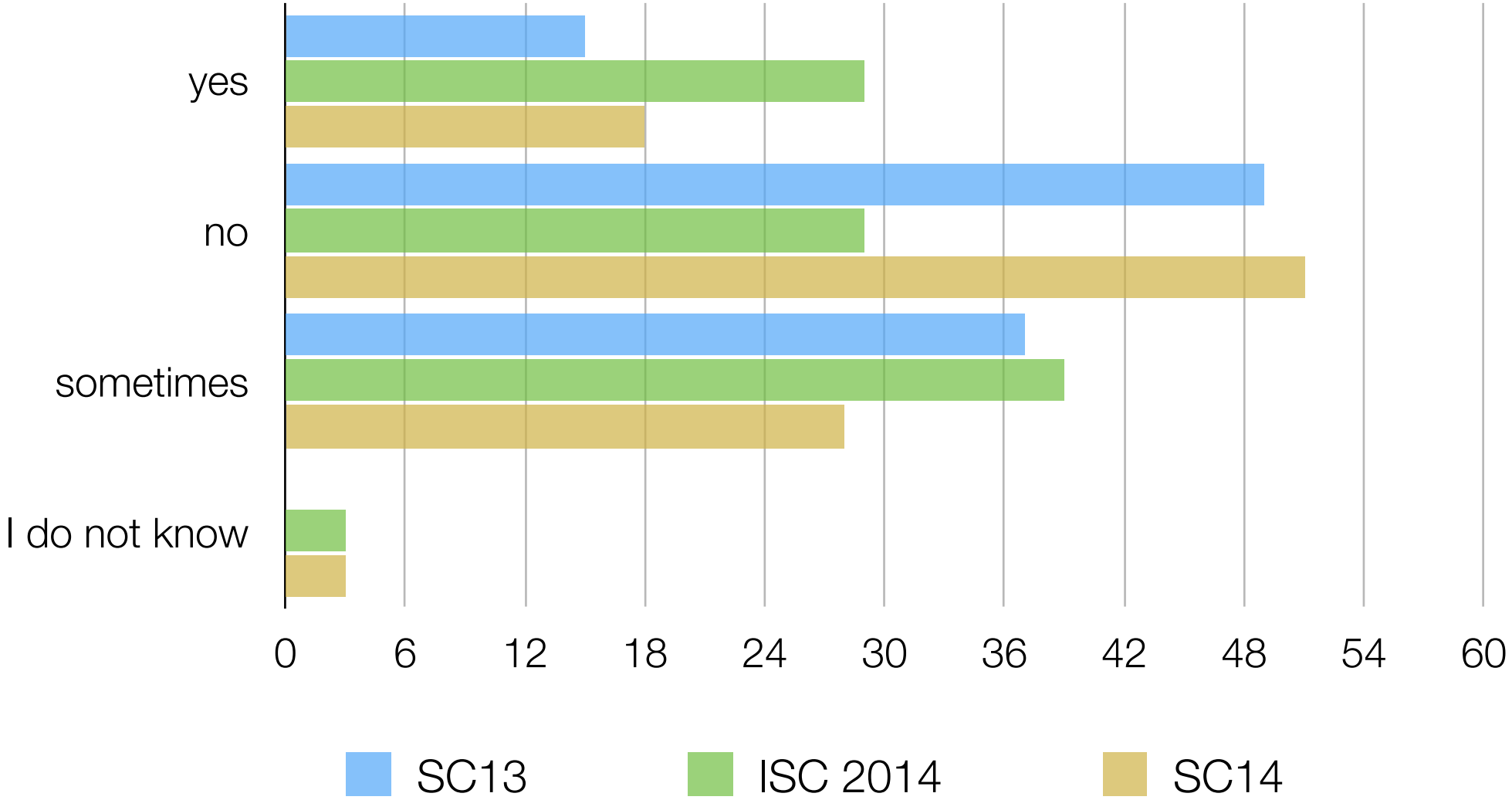


Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
  - simple: verifying existence of binaries, libraries, …
  - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you collaborate with other sites?



UNIVERSITEIT GENT

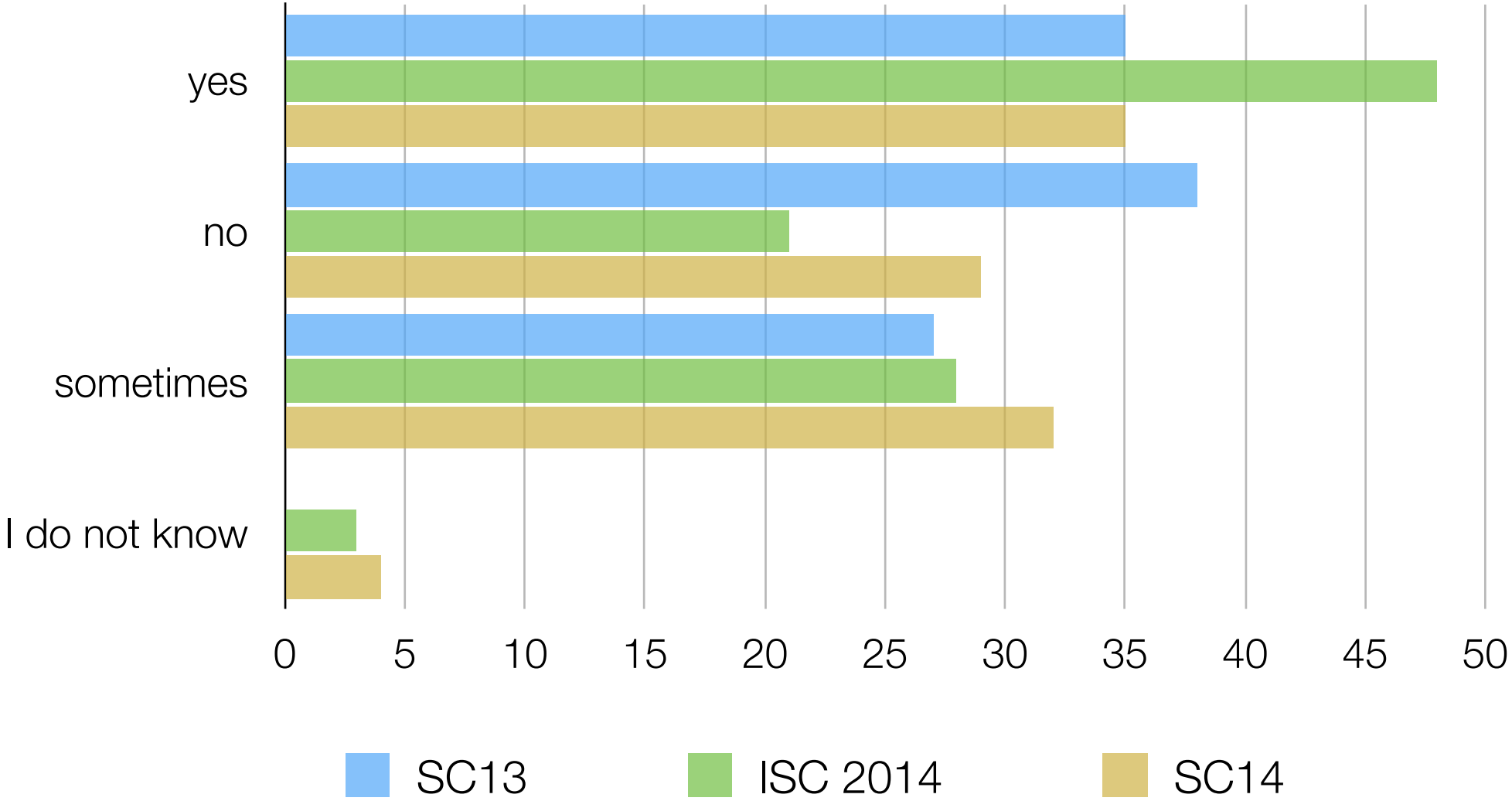
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
    - simple: verifying existence of binaries, libraries, …
    - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you automate software builds?



UNIVERSITEIT GENT

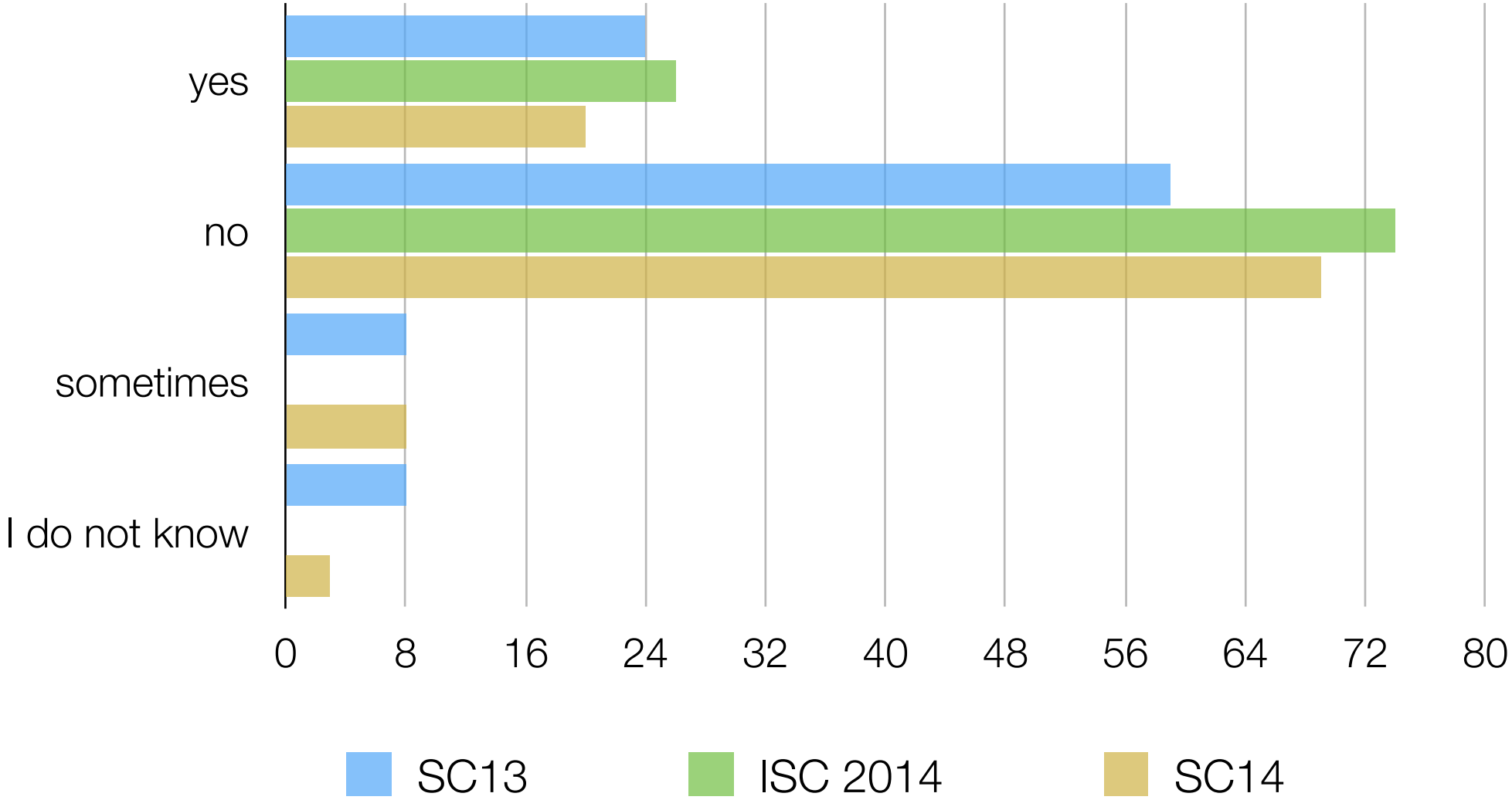
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
    - simple: verifying existence of binaries, libraries, …
    - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Are module files automagically generated?



UNIVERSITEIT GENT

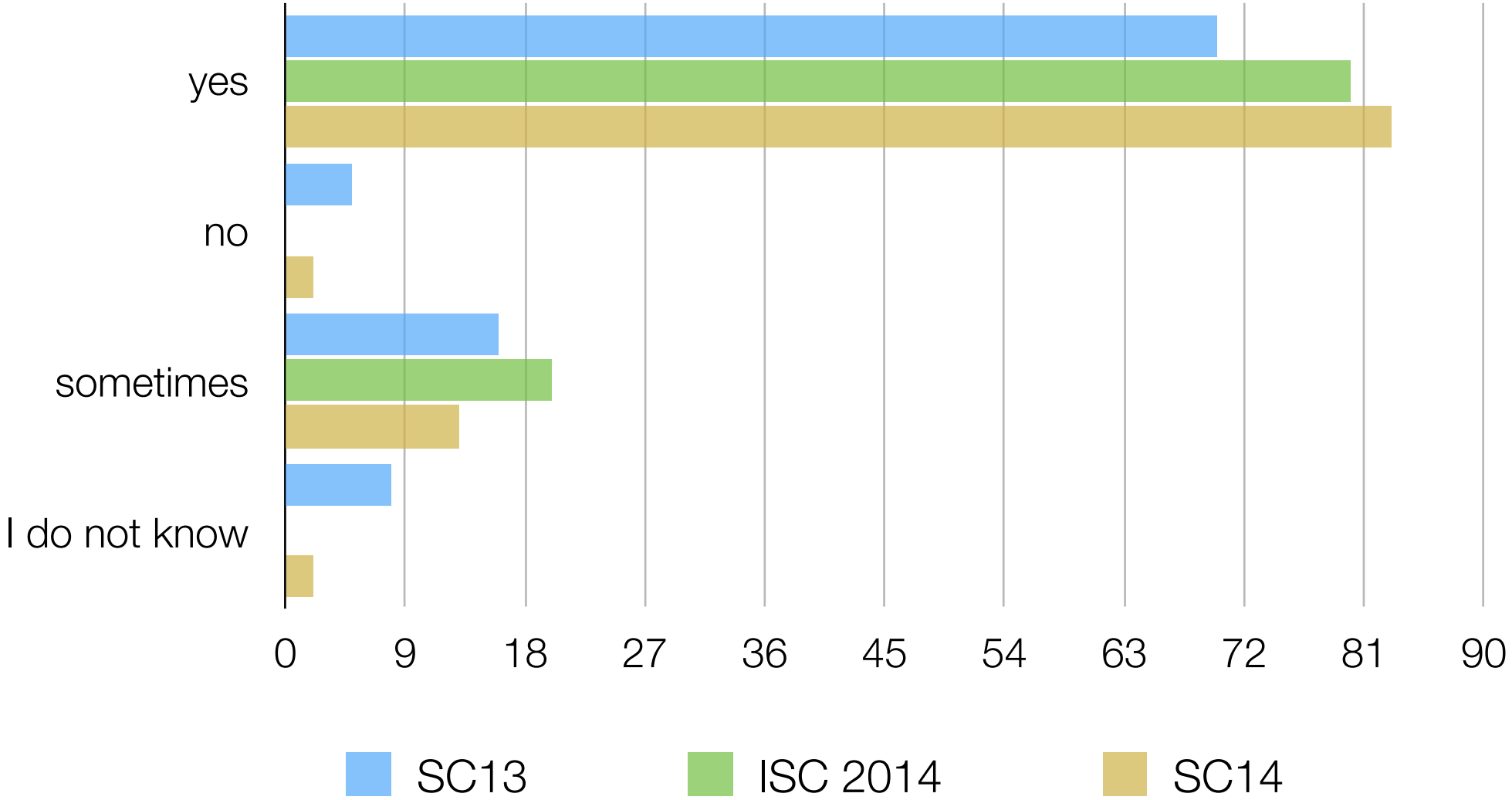
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
  - simple: verifying existence of binaries, libraries, …
  - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Providing multiple builds per application?



UNIVERSITEIT GENT

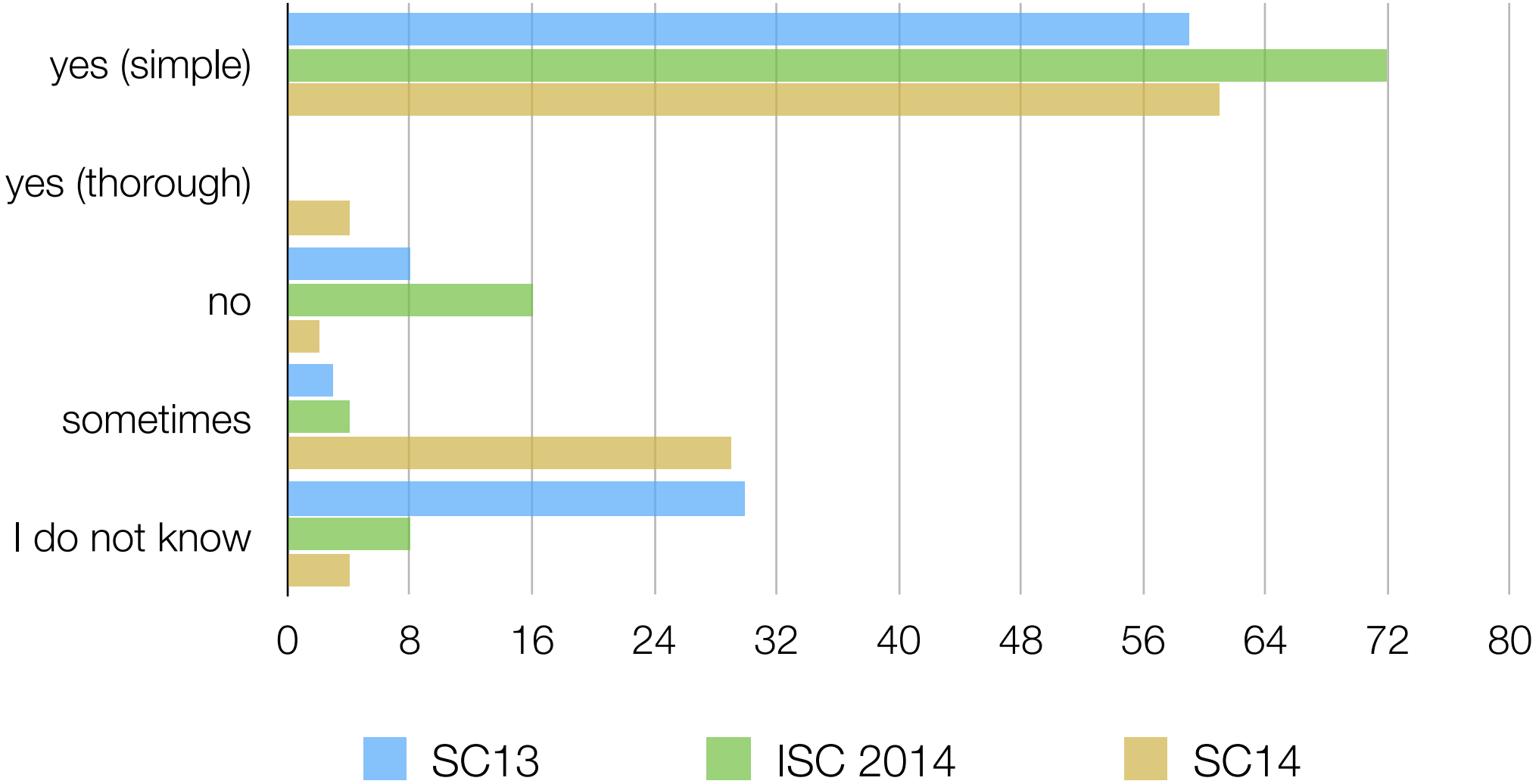
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
  - simple: verifying existence of binaries, libraries, …
  - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you test correctness of software builds?



UNIVERSITEIT GENT

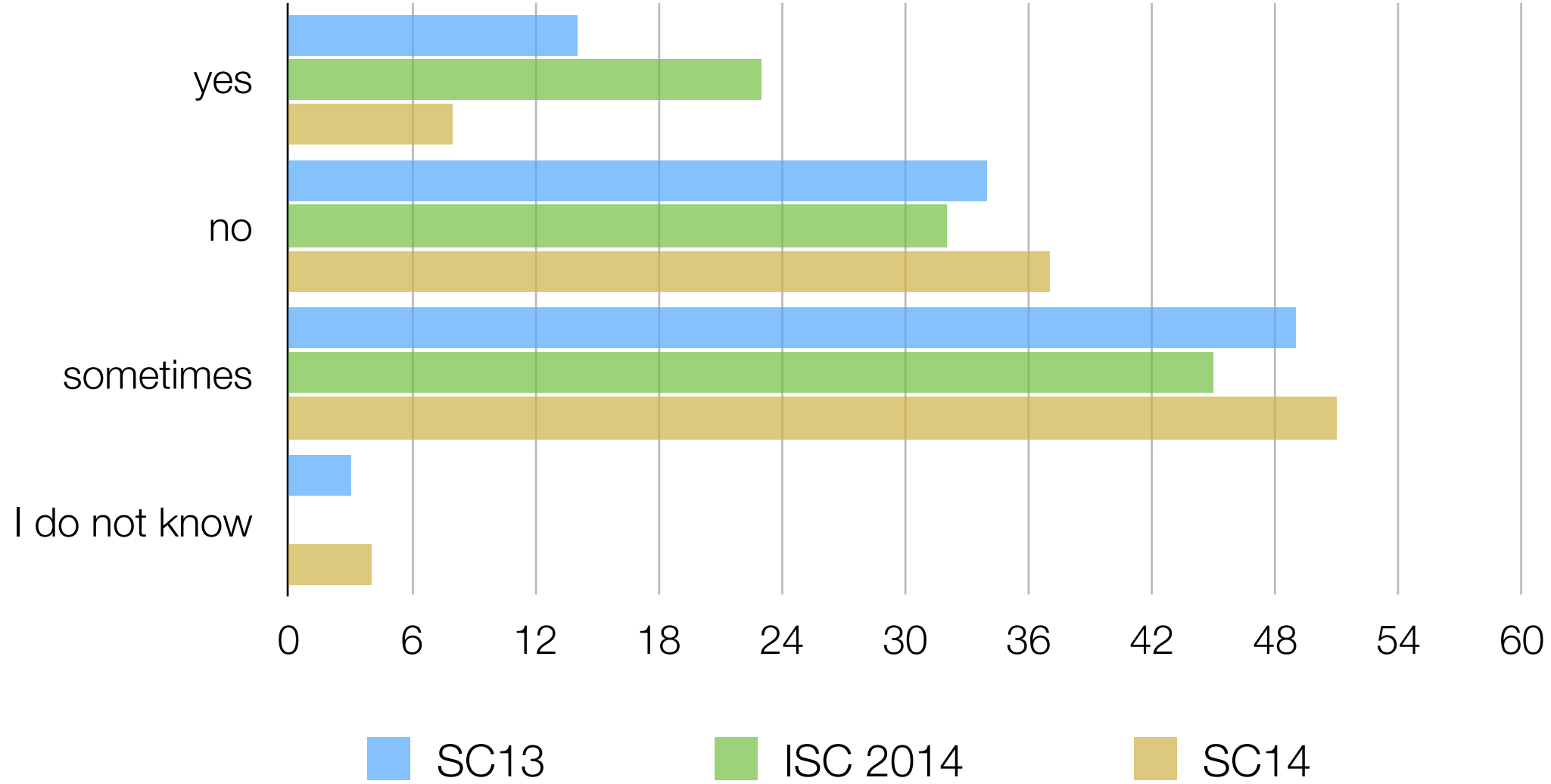
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
    - simple: verifying existence of binaries, libraries, …
    - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you evaluate the software performance?



UNIVERSITEIT GENT

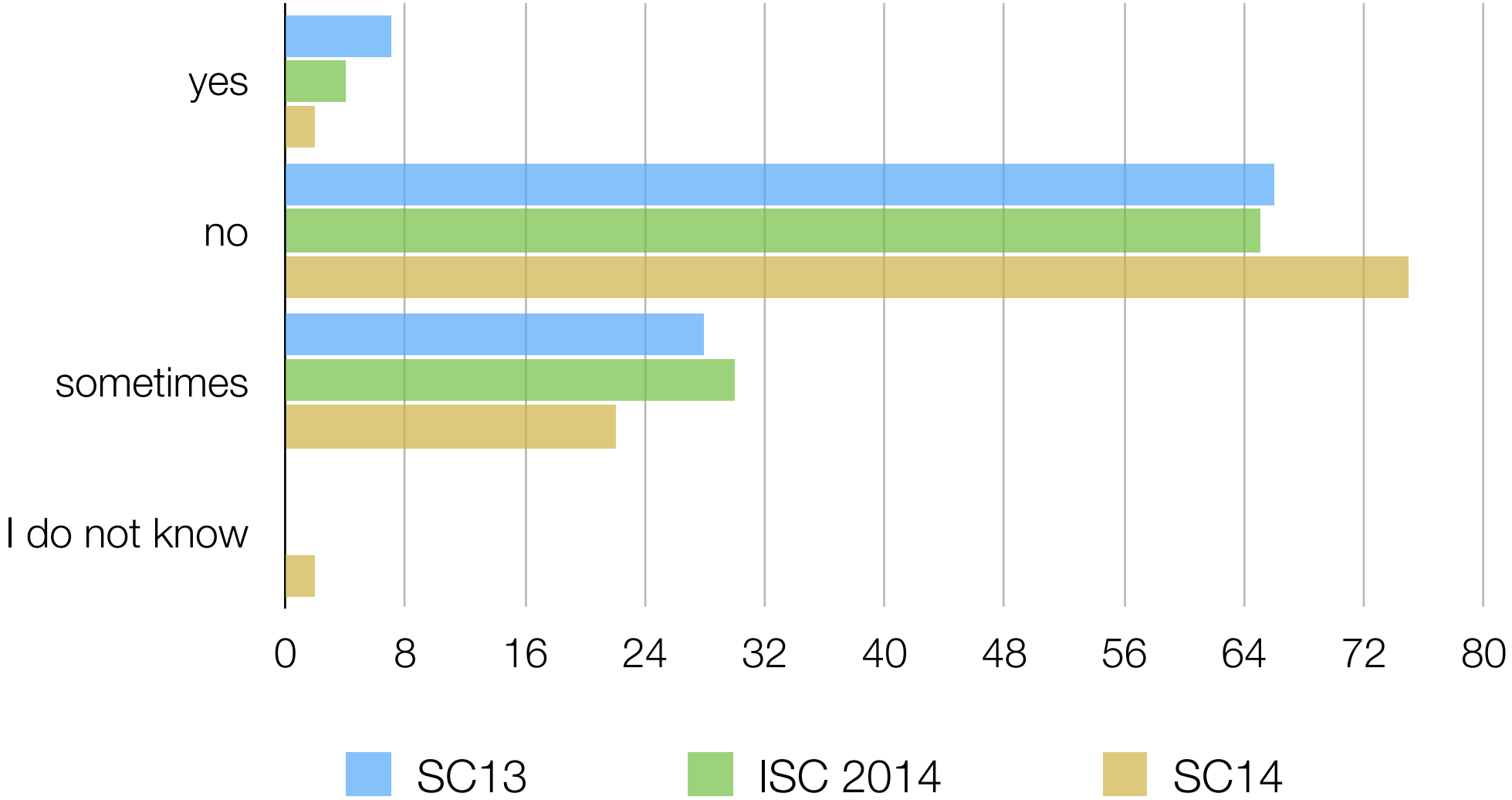
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
    - simple: verifying existence of binaries, libraries, …
    - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you monitor software performance?



UNIVERSITEIT GENT

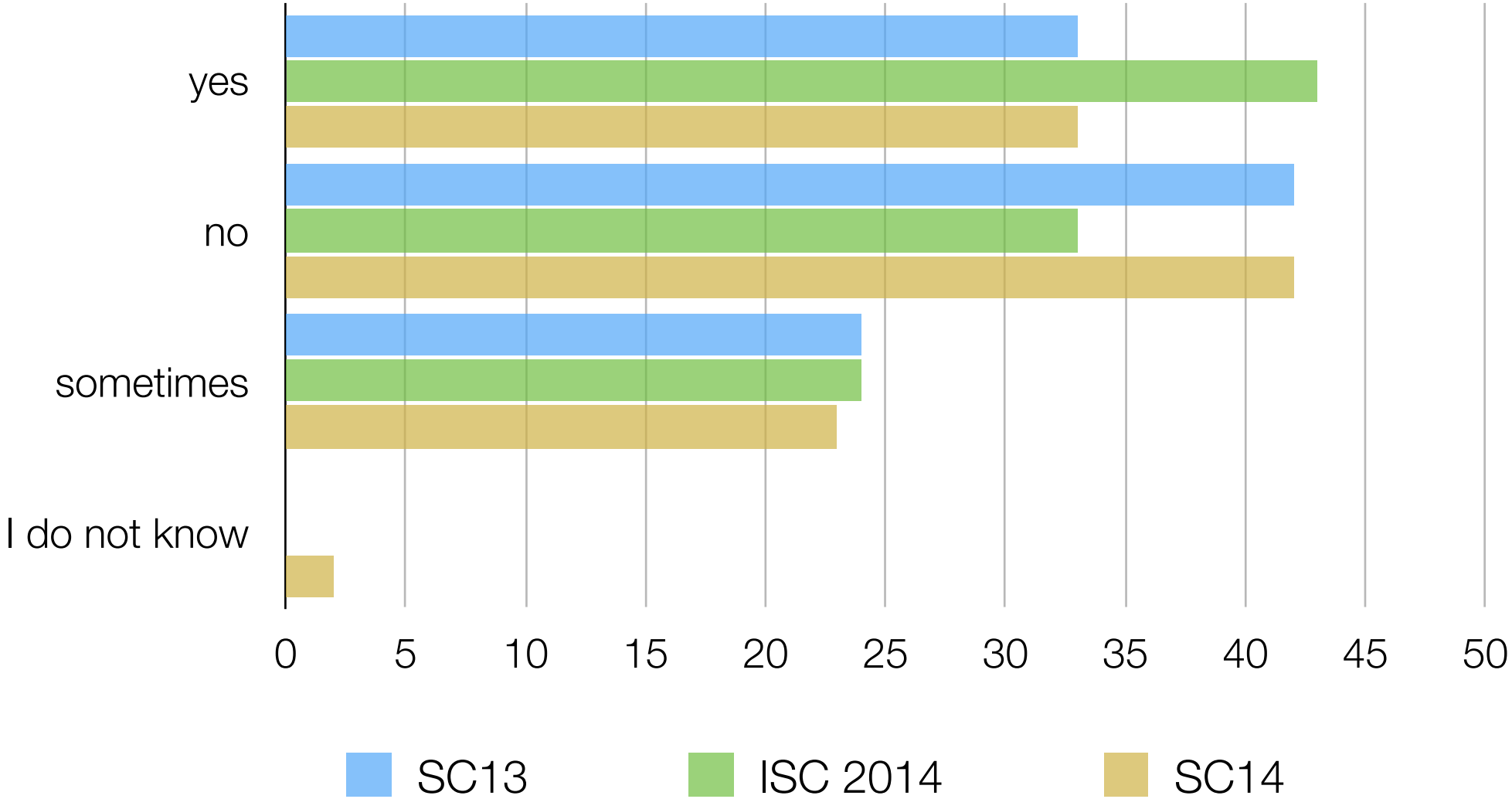
socrative.com - 570181

Which best practices do you use?

- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
    - simple: verifying existence of binaries, libraries, …
    - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you keep track of build metadata?

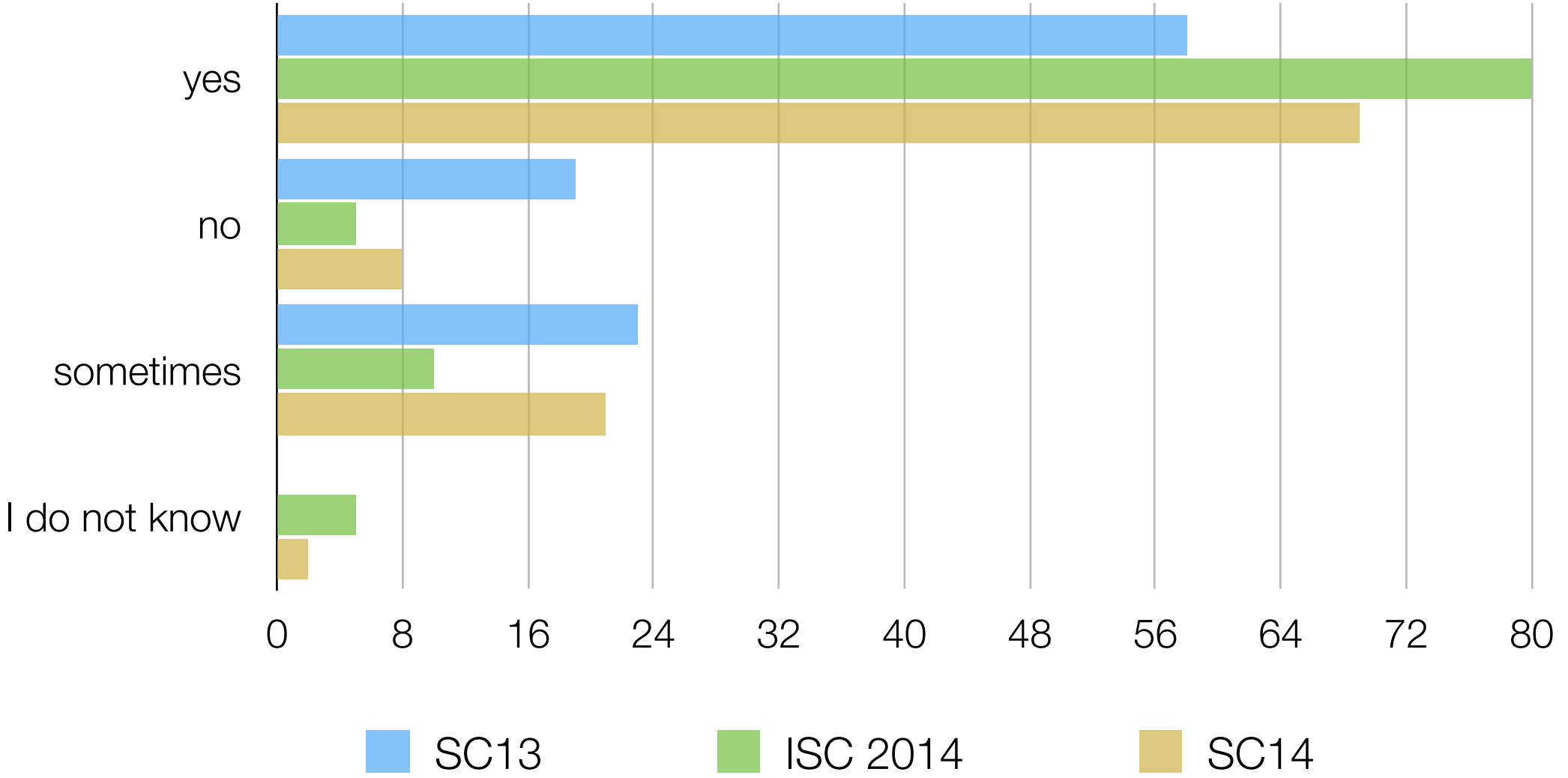


Which best practices do you use?

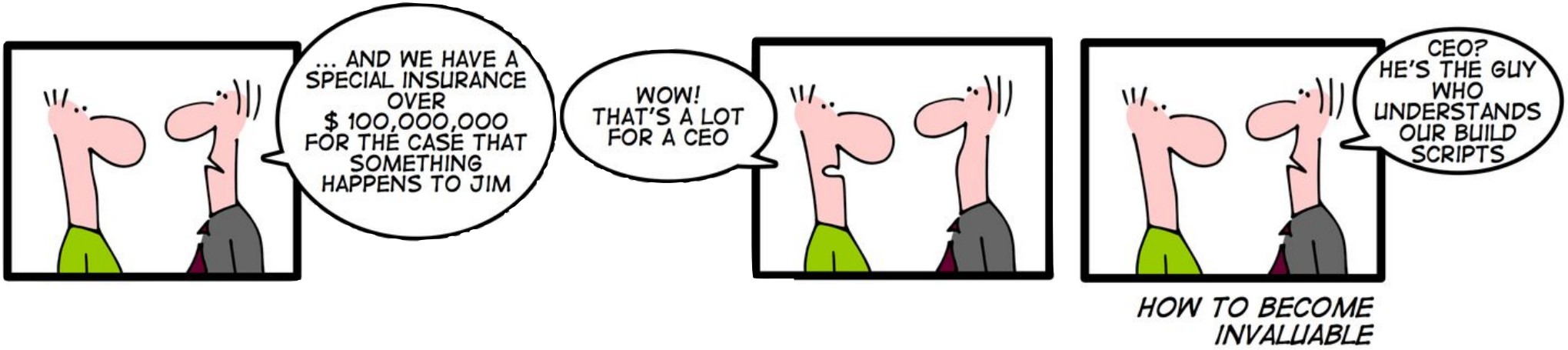
- collaboration with other sites
- automation of builds
- automagical generation of module files
- multiple builds of the same software package
- testing of the installed software
  - simple: verifying existence of binaries, libraries, …
  - thorough: well-defined tests, result verification, …
- performance evaluation and monitoring (over time)
- track build metadata: procedure, log, time, dependencies, …
- archive package sources to counter disappearance upstream

socrative.com - 570181

Do you keep an archive of software sources?



Getting Scientific Software Installed
Tools & Best Practices



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

Sc 14 Bird-of-a-Feather session

19 November 2014

andy.georges@ugent.be and jens.timmerman@ugent.be