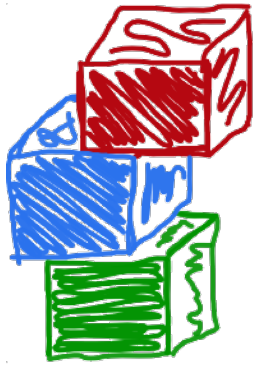


easybuild

Past, present & future

October 22nd 2018 - HPC-SIG workshop, Birmingham (UK)

http://users.ugent.be/~kehoste/EasyBuild_20181022_HPC-SIG_UK.pdf

kenneth.hoste@ugent.be
easybuild@lists.ugent.be

http://easybuilders.github.io/easybuild
http://easybuild.readthedocs.io

GHENT UNIVERSITY

http://www.ugent.be/hpc

https://www.vscentrum.be

Vlaanderen
is computing

HPC-UGent

GHENT UNIVERSITY

- part of central IT department of Ghent University (Belgium)
- centralised scientific computing services, training & support
- for researchers of UGent, industry & knowledge institutes
- core values:
  empowerment - centralisation - automation - collaboration
- member of Flemish Supercomputer Centre (VSC)
  https://www.vscentrum.be

Vlaanderen
is computing

whoami

kenneth.hoste@ugent.be

@boegel (GitHub, IRC, Slack)

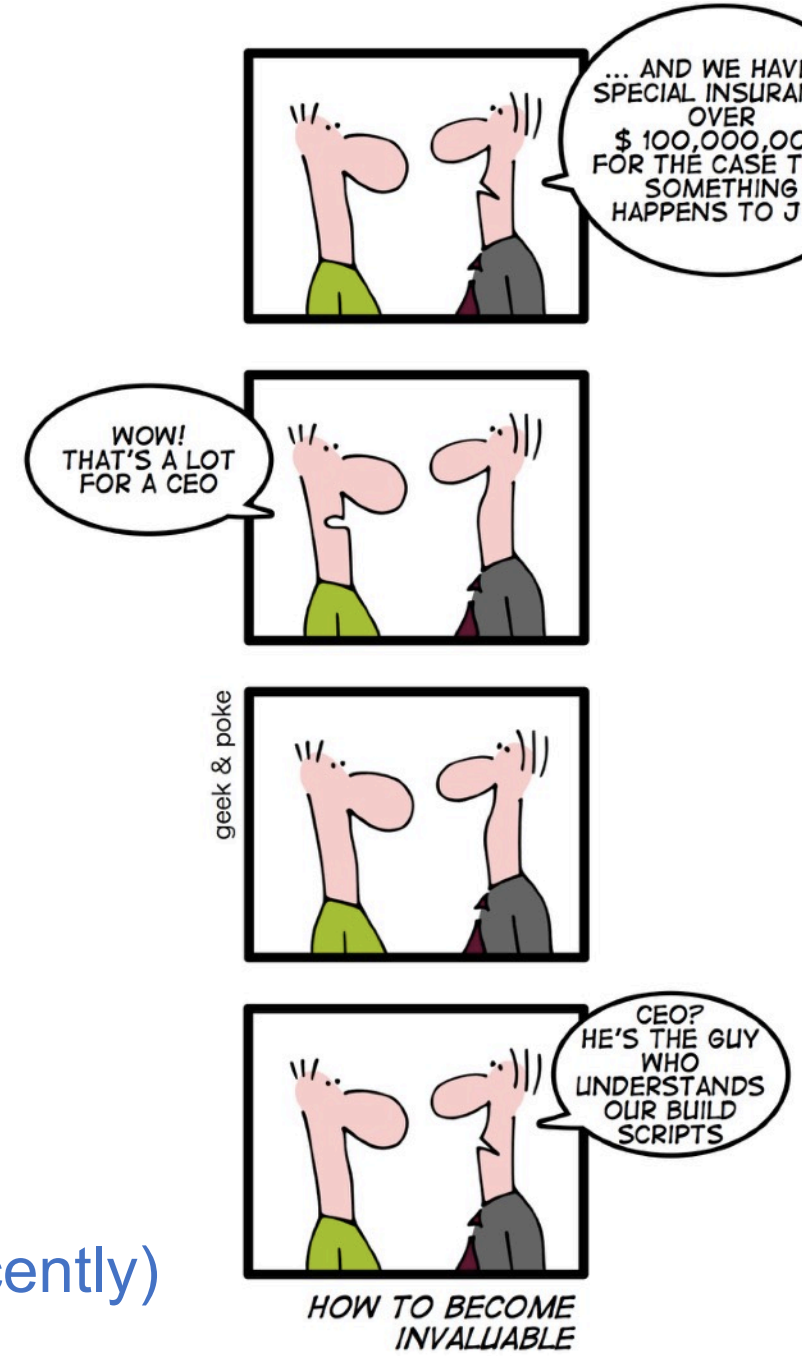
@kehoste (Twitter)

- Masters & PhD in Computer Science from UGent
- joined HPC-UGent team in October 2010
- main tasks: user support & training, *software installations*
- inherited maintenance of EasyBuild in 2011
- slowly also became lead developer & release manager
- likes beer, loud music, FOSS (Free & Open Source Software), dad jokes

Getting Scientific Software Installed

Installation of scientific software is a tremendous problem for HPC sites all around the world.

- ideally built from source (performance is key!)
- tedious, time-consuming, frustrating,
  sometimes simply not worth the (manual) effort...
- huge burden on HPC user support teams
  - over 25% of support tickets at HPC-UGent,
    but consumes *way* more time...
- very little collaboration among HPC sites (until recently)



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

Common issues with scientific software

Researchers focus on the *science* behind the software they implement,
and care little about software engineering, tools, build procedure, portability, ...

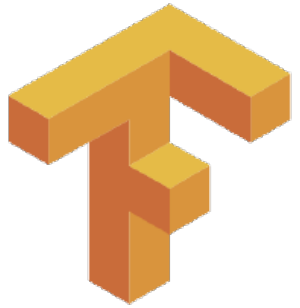
Scientists are (typically) no software developers or sysadmins (nor should they be).

"If we would know what we are doing, it would not be called 'research'."

This results in:

- use of non-standard build systems (or broken ones)
- "creative" software versioning (or no versions at all)
- dependency hell on steroids
- interactive installation scripts
- hardcoded parameters (compilers, libraries, paths, . . .)
- poor/outdated/missing/incorrect documentation

Prime example: TensorFlow



popular open-source software for Deep Learning (https://www.tensorflow.org)

- self-installs most dependencies, but not all...
- '`configure`' script is a custom interactive script
  - silent configuration possible, if you know which `$TF_*` environment variables to set
- uses ***Bazel*** (http://bazel.io) as build tool (there ~~is~~ was a contributed CMake alternative...)
  - *resets environment*, may result in unsetting important env vars (e.g., `$PYTHONPATH`)
  - quite different from other build tools; e.g. to build TensorFlow:
    ```
    bazel build --config=opt //tensorflow/tools/pip_package:build_pip_package
    ```
  - `--config=opt`, `-c opt` and `-copt=...` all mean different things... `0_o`
- installation via '`pip install`' of locally built Python wheel file (`.whl`)

No, that's not a typo...

How to make package managers cry

FOSDEM 2018

https://archive.fosdem.org/2018/schedule/event/how_to_make_package_managers_cry

https://www.youtube.com/watch?v=NSemlYagjIU

- me **venting ~7 years of frustration** with getting scientific software installed
- TensorFlow as main motivator
- **sarcastic tone for dramatic effect** (it worked!)
- lots of feedback (and ideas for an extended version of the talk), **clearly hit a nerve**...
- others discussing similar points:
  - *"Software disenchantment"*, blog post by Nikita Prokopov (Sept 2018)
    http://tonsky.me/blog/disenchantment
  - *"Don't package your libraries, write packageable libraries!"*, talk at CppCon 2018
    by Robert Schumacher (Microsoft) - *https://www.youtube.com/watch?v=sBP17HQAQjk*

What about existing software installation tools?

- package managers: *yum* (RPMs), *apt* (.deb), ...
- *Homebrew* (macOS), http://brew.sh ; *Linuxbrew*, http://linuxbrew.sh
- *Portage* (Gentoo), http://wiki.gentoo.org/wiki/Project:Portage
- *pkgsrc* (NetBSD & (a lot) more), http://pkgsrc.org

None are well suited to scientific software and HPC systems in particular.

Common problems:

- usually poor support for old/multiple versions and/or to have builds side-by-side
- not flexible enough to deal with idiosyncrasies of scientific software
- little support for scientific software, non-GCC compilers, MPI, ...

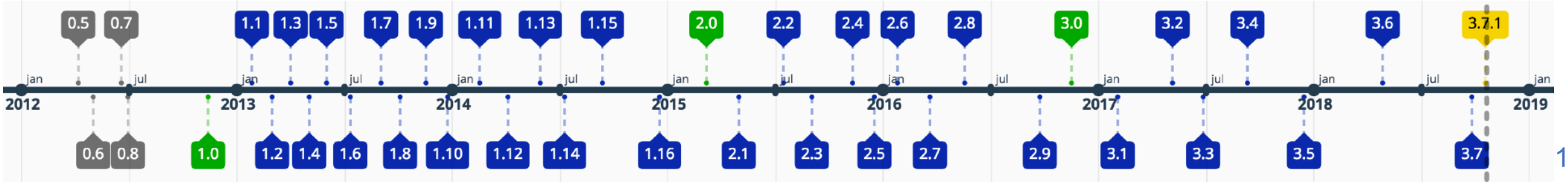
Modern exceptions (depending on use case): **Spack** & **Nix/Guix**

https://easybuilders.github.io/easybuild - https://easybuild.readthedocs.io

- **framework for building & installing scientific software**
- strong focus on Linux & HPC, requires environment modules tool (e.g. Lmod)
- implemented in Python 2, lead development by HPC-UGent
- available under GPLv2 license via PyPI, GitHub
- supports different compilers & MPI libraries, **> 1,500 different software packages**
- **active & helpful worldwide community**

Almost 6 years of (stable)

- in-house development at HPC-UGent since summer 2009
- first public release in April 2012 (EasyBuild v0.5)
- **first stable release on November 13th 2012, during SC'12 (EasyBuild v1.0)**
- intention was to get feedback, but gradually a community emerged around it...
- **frequent stable releases** since then (latest: EasyBuild v3.7.1, Oct 18th 2018)
- community-driven development: bug reports, feature requests, contributions



Supported software

http://easybuild.readthedocs.io/en/latest/version-specific/Supported_software.html

- EasyBuild v3.7.1 supports installing **over 1,500 software packages**
  - including CP2K, NAMD, NWChem, OpenFOAM, TensorFlow, WRF, ...
  - also a lot of bioinformatics software is supported out of the box
  - + ~1,000 extensions: Python packages, R libraries, Perl modules, X11 libraries, ...
- diverse toolchain support:
  - compilers: GCC, Intel, Clang, PGI, IBM XL, Cray, CUDA
  - MPI libraries: OpenMPI, Intel MPI, MPICH, MPICH2, MVAPICH2, Cray MPI, ...
  - BLAS/LAPACK libraries: Intel MKL, OpenBLAS, ScaLAPACK, BLIS, Cray LibSci, ...

Terminology

http://easybuild.readthedocs.io/en/latest/Concepts_and_Terminology.html

- **EasyBuild framework**
  - core of EasyBuild: Python modules & packages
  - provides supporting functionality for building/installing software, generating modules, ...
- **easyblock**
  - a Python module that serves as a build script, 'plugin' for the EasyBuild framework
  - implements a (generic or software-specific) build/install procedure
- **easyconfig file** (*.eb): build specification; software name/version, compiler toolchain, etc.
- **(compiler) toolchain***:* set of compilers + accompanying libraries (MPI, BLAS/LAPACK, ...)
- **extensions***:* additional packages for a particular applications (e.g., Python, R)

Feature highlights (1)

- fully **autonomously** building and installing (scientific) software
  - automatic dependency resolution (`--robot`)
  - automatic generation of module files (Tcl or Lua syntax)
- thorough **logging** of executed build/install procedure
- **archiving** of easyconfigs and patches
- highly **configurable**, via config files/environment/command line
- **dynamically extendable** with additional easyblocks, toolchains, etc.

Feature highlights (2)

- support for **custom module naming schemes** (incl. hierarchical)
- **transparency** via support for 'dry run' installation & trace output
- **comprehensively tested**: lots of unit tests, regression testing, ...
- actively developed, **frequent stable releases**
- **collaboration** between various HPC sites large & small
- integration with Torque/SLURM, FPM, Docker, Singularity, ...
- worldwide **community**

What EasyBuild is (not)

EasyBuild is:

- *not* YABT (Yet Another Build Tool)

  it does *not* replace build tools like `cmake` or `make`; it wraps around them

- *not* a replacement for your favourite package manager (`yum`, `apt-get`, ...)

  it leverages some tools & libraries provided by the OS (glibc, OpenSSL, libibverbs, ...)

- *not* a magic solution to all your (software compilation/installation) problems...

  you may still run into compiler errors (if nobody else has already taken care of it)

What EasyBuild is (not)

EasyBuild is:

- a **uniform interface** that wraps around software installation procedures
- a huge **time-saver**, by automating tedious/boring/repetitive tasks
- a way to provide a **consistent software stack** to your users
- an **expert system** for software installation on HPC systems
- a **platform for collaboration** with HPC sites worldwide
- a way to **empower *users* to self-manage their software stack** on HPC systems

Basic usage

http://easybuild.readthedocs.io/en/latest/Using_the_EasyBuild_command_line.html

http://easybuild.readthedocs.io/en/latest/Typical_workflow_example_with_WRF.html

- specify software name/version and toolchain to '`eb`' command
- commonly via easyconfig filename(s):

```
eb GCC-4.9.2.eb Clang-3.6.0-GCC-4.9.2.eb
```

- check whether required toolchain & dependencies are available using `--dry-run`/`-D`:

```
eb Python-2.7.14-intel-2017b.eb -D
```

- enable dependency resolution via `--robot`/`-r`:

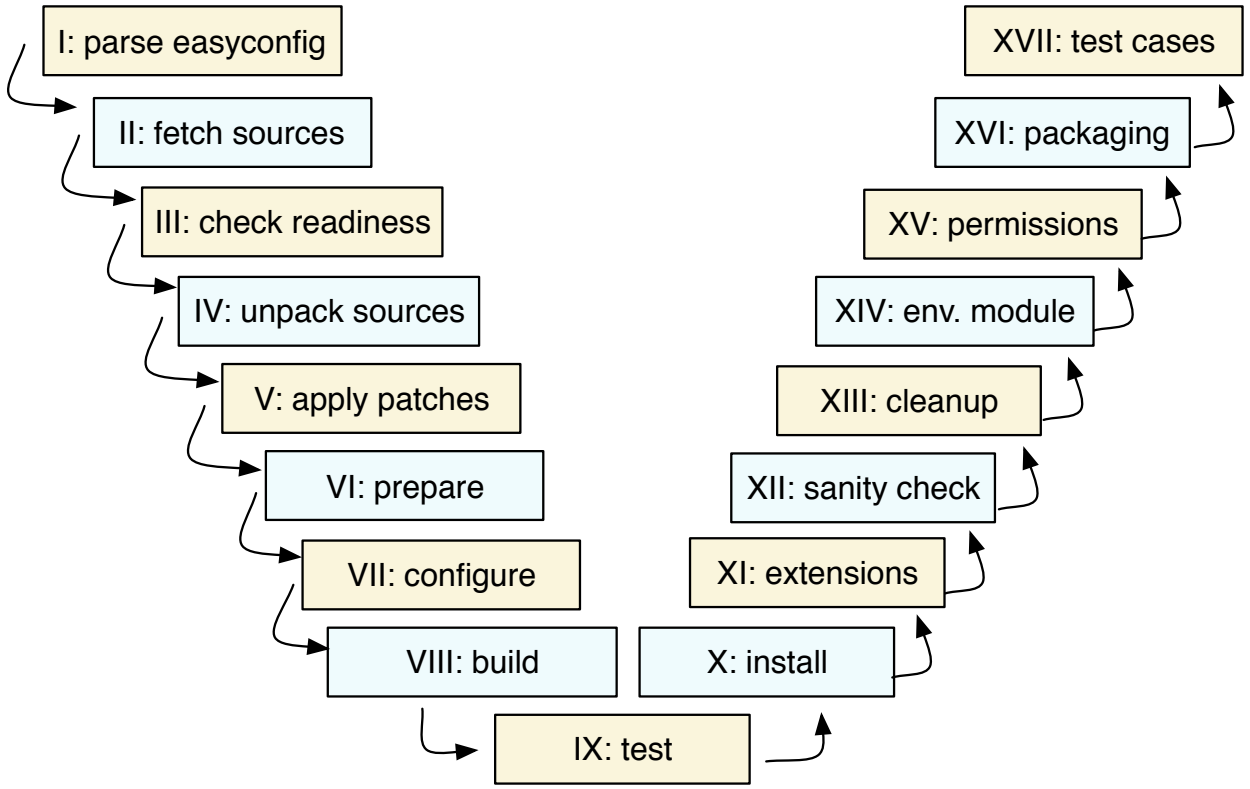
```
eb WRF-3.8.0-intel-2016b-dmpar.eb -r
```

Example output of using 'eb' command

```
$ eb TensorFlow-1.10.1-foss-2018b-Python-3.6.6.eb
== temporary log file in case of crash /tmp/eb-GyvPHx/easybuild-U1TkEI.log
== processing EasyBuild easyconfig TensorFlow-1.10.1-foss-2018b-Python-3.6.6.eb
== building and installing TensorFlow/1.10.1-foss-2018b-Python-3.6.6...
== fetching files...
== creating build dir, resetting environment...
== unpacking...
== patching...
== preparing...
== configuring...
== building...
== testing...
== installing...
== taking care of extensions...
== postprocessing...
== sanity checking...
== cleaning up...
== creating module...
== permissions...
== packaging...
== COMPLETED: Installation ended successfully
== Results of the build can be found in the log file /opt/easybuild/software/Tensor...
== Build succeeded for 1 out of 1
== Temporary log file(s) /tmp/eb-GyvPHx/easybuild-U1TkEI.log* have been removed.
== Temporary directory /tmp/eb-GyvPHx has been removed.
```

Step-wise installation procedure

EasyBuild performs a step-wise installation procedure for each software:



- download sources (best effort)
- set up build directory & environment
  - unpack sources (& apply patches)
  - load modules for toolchain & deps
  - define toolchain-related env vars (`$CC`, `$CFLAGS`, ...)
- configure, build, (test), install, (extensions)
- perform a simple sanity check on installation
- generate module file

each step can be customised via easyconfig parameters or an easyblock

Log files

http://easybuild.readthedocs.io/en/latest/Logfiles.html

- **EasyBuild *thoroughly* logs the executed installation procedure**
  - active EasyBuild configuration
  - easyconfig file that was used
  - modules that were loaded + resulting changes to environment
  - defined environment variables
  - output of executed commands
  - informative log messages produced by easyblock
- log is file is copied to software installation directory for future reference
- can be used to debug build problems or see how installation was performed exactly

Log files: example

```
== 2016-04-24 13:34:31,906 main.EB_HPL INFO This is EasyBuild 3.1.2 (framework:
3.1.2, easyblocks: 3.1.2) on host example.
...
== 2016-04-24 13:34:35,503 main.EB_HPL INFO configuring...
== 2016-04-24 13:34:48,817 main.EB_HPL INFO Starting configure step
...
== 2016-04-24 13:34:48,823 main.EB_HPL INFO Running method configure_step part of
step configure
...
== 2016-04-24 13:34:48,823 main.run DEBUG run_cmd: running cmd /bin/bash
make_generic (in /tmp/user/easybuild_build/HPL/2.0/goolf-1.4.10/hpl-2.0/setup)
== 2016-04-24 13:34:48,823 main.run DEBUG run_cmd: Command output will be logged
to /tmp/easybuild-W85p4r/easybuild-run_cmd-XoJwMY.log
== 2016-04-24 13:34:48,849 main.run INFO cmd "/bin/bash make_generic" exited with
exitcode 0 and output:
...
```

Easyconfig files as build specifications

http://easybuild.readthedocs.io/en/latest/Writing_easyconfig_files.html

- **simple text files including a set of easyconfig parameters** (in Python syntax)
- some are mandatory: software name/version, toolchain, metadata (homepage, descr.)
- other commonly used parameters:
  - easyblock to use
  - list of sources & patches
  - (build) dependencies
  - options for configure/build/install commands
  - files and directories that should be present (sanity check)

Example easyconfig file

no easyblock specified, which implies using a software-specific easyblock (`EB_WRF`)

software name and version

build variant (specific to WRF)
('dmpar': distributed, MPI)

software metadata

toolchain name & version

sources & patches

list of (build) dependencies
note: all versions are *fixed*!

```
name = 'WRF'
version = '3.8.0'

buildtype = 'dmpar'  # custom parameter for WRF
versionsuffix = '-' + buildtype  # part of module name

homepage = 'http://www.wrf-model.org'
description = "Weather Research and Forecasting (WRF) Model"

toolchain = {'name': 'intel', 'version': '2016b'}

source_urls = ['http://www.mmm.ucar.edu/wrf/src/']
sources = ['%(name)sV%(version_major_minor)s.TAR.gz']
patches = ['WRF-%(version)s_known_problems.patch']

builddependencies = [('tcsh', '6.20.00')]
dependencies = [
    ('JasPer', '2.0.10'),
    ('netCDF', '4.4.1'),
    ('netCDF-Fortran', '4.4.4'),
]
```

Configuring and extending EasyBuild

- configuring can be done via config files, environment, and command line interface
  - *all* configuration options are supported on all 3 levels
  - CLI overrides environment which overrides config files
  - *http://easybuild.readthedocs.io/en/latest/Configuration.html*
- easy to extend EasyBuild:
  - (also) use own easyconfigs repositories via `--robot-paths`
  - add additional easyblocks, toolchains, module naming schemes via --include-*
  - *http://easybuild.readthedocs.io/en/latest/Including_additional_Python_modules.html*

Inspecting the current configuration

http://easybuild.readthedocs.io/en/latest/Configuration.html

- use '`eb --show-config`' to get an overview of the current configuration
- only shows a couple of important settings + anything different from default
- full list of settings for current configuration via '`eb --show-full-config`'

```
$ EASYBUILD_PREFIX=/tmp eb --buildpath /dev/shm --show-config
#
# Current EasyBuild configuration
# (C: command line argument, D: default value, E: environment variable, F: configuration file)
#
buildpath      (C) = /dev/shm
installpath    (E) = /tmp
packagepath    (E) = /tmp/packages
prefix         (E) = /tmp
repositorypath (E) = /tmp/ebfiles_repo
robot-paths    (D) = /home/example/easybuild-easyconfigs/easybuild/easyconfigs
sourcepath     (E) = /tmp/sources
```

Integration with Lmod

- support for using Lmod as modules tool was added in EasyBuild v1.6.0 (July 2013)
- kind of out of necessity...
  - installing lots of modules was (too) easy using EasyBuild
  - traditional Tcl-based modules tool was too slow (no module cache)
- later also:
  - support for using a hierarchical module naming scheme (EasyBuild 1.14.0, July 2014)
  - support for module files in Lua syntax (EasyBuild v2.1.0, April 2015)
- Lmod (& Lua syntax) became the default in EasyBuild v3.0.0 (November 2016)

Integration with Lmod

- synergy between Lmod & EasyBuild has grown over time
- communities engaging with each other
- increased adoption for both tools thanks to integration
- (significant) enhancements to both tools inspired by or thanks to other community
  - improved performance, Lmod's `update_lmod_system_cache_files` script, ...
- joint papers on "Modern Scientific Software Management"
- lead developers became good friends :)

Flat module naming scheme

- all modules are always available for loading
- long(er) module names
- '`module avail`' may be overwhelming for users
- too easy to load incompatible modules together

legend

(not available)

(available)

(loaded)

GCC/5.3.0

GCC/6.1.0

OpenMPI/1.10.2-GCC-5.3.0

OpenMPI/2.1.0-GCC-5.3.0

OpenMPI/1.10.3-GCC-6.1.0

OpenMPI/2.1.0-GCC-6.1.0

FFTW/3.3.4-gompi-2016.04

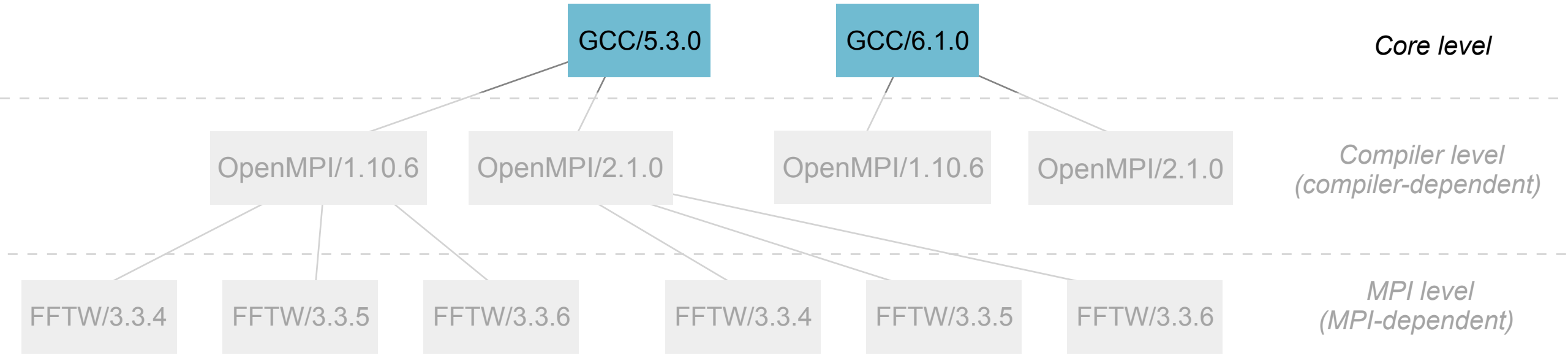
FFTW/3.3.6-gompi-2016.04

FFTW/3.3.4-gompi-2016.07

FFTW/3.3.6-gompi-2016.07

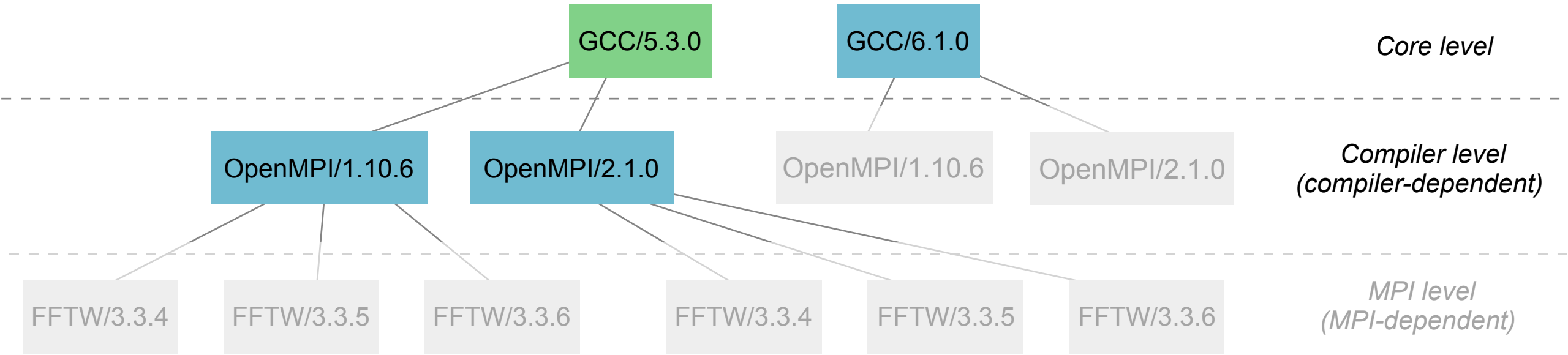
Hierarchical module naming scheme (1)

- modules are organised in a tree-like fashion
- initially, only 'core' modules are available for loading
- typically 3 hierarchy levels: core, compiler-dependent, MPI-dependent
- other modules are only visible via '`module spider`'



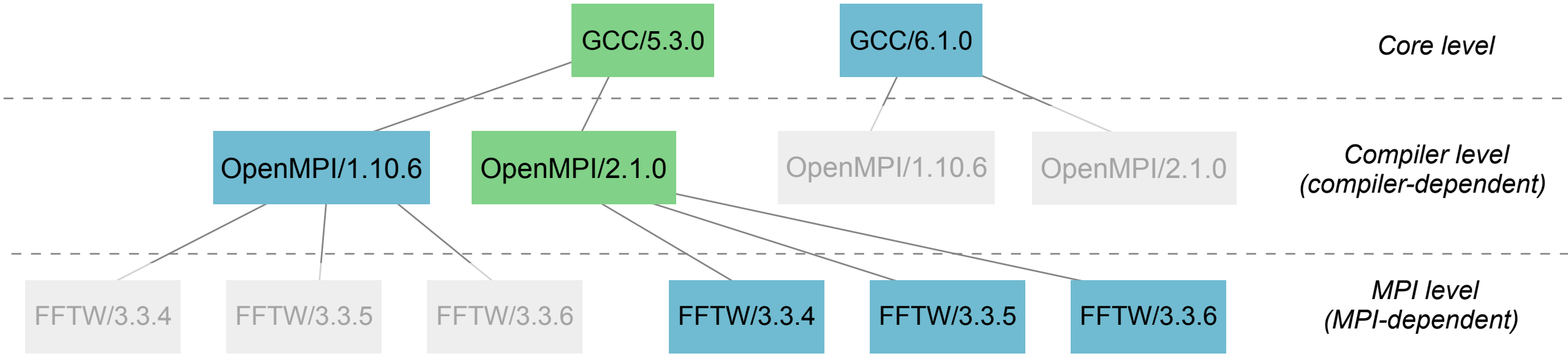
Hierarchical module naming scheme (2)

- Core modules may extend `$MODULEPATH` with an additional location
- loading a Core module may make more modules available
- in this example, loading a GCC module makes OpenMPI modules available



Hierarchical module naming scheme (3)

- even more modules may be made available by loading other modules
- for example, loading an OpenMPI modules reveals MPI-dependent modules
- EasyBuild can organise modules in hierarchy for you!



Using a custom module naming scheme

- a couple of different module naming schemes are included in EasyBuild
  - see `--avail-module-naming-schemes`
  - specify active module naming scheme via `--module-naming-scheme`
  - default: `EasyBuildMNS` (`<name>/<version>-<toolchain>-<versionsuffix>`)
- **you can implement your own module naming scheme relatively easily**
  - specify how to compose module name using provided metadata
  - via Python module that defines custom derivative class of `ModuleNamingScheme`
  - make EasyBuild aware of it via `--include-module-naming-schemes`
- decouple naming of install dirs vs modules via `--fixed-installdir-naming-scheme`

Transparency of performed install procedure

http://easybuild.readthedocs.io/en/latest/Extended_dry_run.html

- '`eb --extended-dry-run`' (or '`eb -x`') reveals planned installation procedure
- **runs in a matter of seconds**
- shows commands that will be executed, build environment, generated module file, ...
- any errors that occur in used easyblock are ignored but clearly reported
- not 100% accurate since easyblock may require certain files to be present, etc.
- very useful when debugging easyblocks, instant feedback as a first pass
- implementation motivated by requests from the community
- helps to avoid impression that EasyBuild is a magic black box for installing software

Example output of `--extended-dry-run` (1)


```
$ eb WRF-3.8.0-intel-2016b-dmpar.eb -x

== temporary log file in case of crash /tmp/eb-Dh1wOp/easybuild-0bu9u9.log

== processing EasyBuild easyconfig /home/example/eb/easybuild-easyconfigs/easybuild/
easyconfigs/w/WRF/WRF-3.8.0-intel-2016b-dmpar.eb

...

*** DRY RUN using 'EB_WRF' easyblock (easybuild.easyblocks.wrf @ /home/example/eb/easybuild-
easyblocks/easybuild/easyblocks/w/wrf.py) ***

== building and installing WRF/3.8.0-intel-2016b-dmpar...
fetching files... [DRY RUN]

[fetch_step method]
Available download URLs for sources/patches:
  * http://www2.mmm.ucar.edu/wrf/src//$source
  * http://www.mmm.ucar.edu/wrf/src//$source

List of sources:
  * WRFV3.8.0.TAR.gz will be downloaded to /home/example/eb/sources/w/WRF/WRFV3.8.0.TAR.gz
```

Example output of `--extended-dry-run` (2)


```
$ eb WRF-3.8.0-intel-2016b-dmpar.eb -x
...

building... [DRY RUN]

[build_step method]
  running command "tcsh ./compile -j 4 wrf"
  (in /home/example/eb/software/WRF/3.8.0-intel-2016b-dmpar/WRF-3.8.0)
  running command "tcsh ./compile -j 4 em_real"
  (in /home/example/eb/software/WRF/3.8.0-intel-2016b-dmpar/WRF-3.8.0)
  running command "tcsh ./compile -j 4 em_b_wave"
  (in /home/example/eb/software/WRF/3.8.0-intel-2016b-dmpar/WRF-3.8.0)
...
[sanity_check_step method]
Sanity check paths - file ['files']
  * WRFV3/main/libwrflib.a
  * WRFV3/main/real.exe
  * WRFV3/main/wrf.exe
Sanity check paths - (non-empty) directory ['dirs']
  * WRFV3/main
  * WRFV3/run
Sanity check commands
  (none)
```

Example output of `--extended-dry-run` (3)

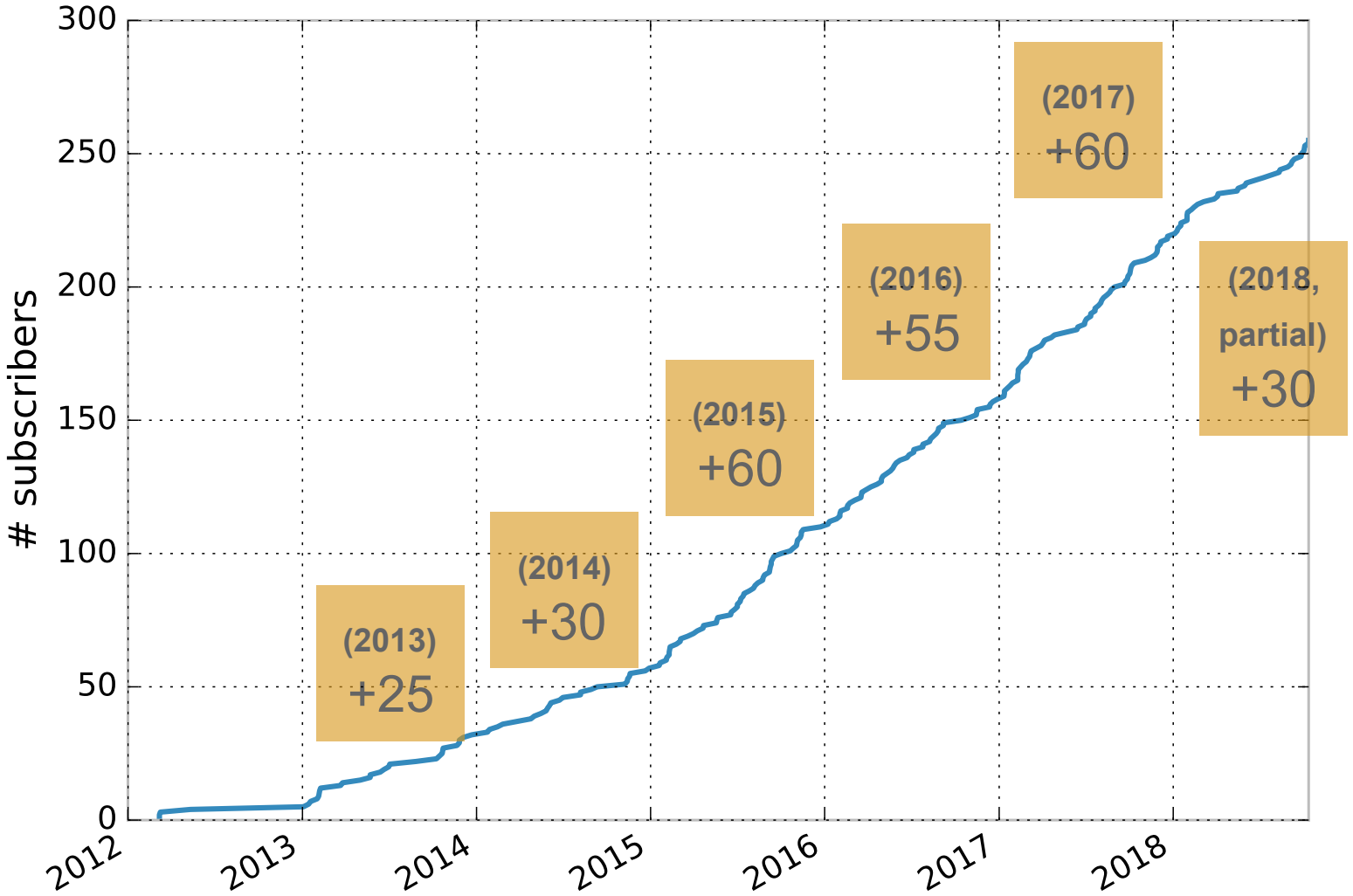

```
$ eb WRF-3.8.0-intel-2016b-dmpar.eb -x
...

[make_module_step method]
Generating module file /home/example/eb/modules/all/WRF/3.8.0-intel-2016b-dmpar,
with contents:

    #%Module
    proc ModulesHelp { } {
        puts stderr { The Weather Research and Forecasting (WRF) Model }
    }
    module-whatis {Description: WRF - Homepage: http://www.wrf-model.org}

    set root /home/example/eb/software/WRF/3.8.0-intel-2016b-dmpar

    conflict WRF

    if { ![ is-loaded intel/2016b ] } {
        module load intel/2016b
    }
    if { ![ is-loaded JasPer/1.900.1-intel-2016b ] } {
        module load JasPer/1.900.1-intel-2016b
    }
```

The EasyBuild community

- EasyBuild community has been growing rapidly the last couple of years
- **hundreds of HPC sites and companies worldwide, incl. JSC, CSCS, SURFsara, Pfizer, ...**
- very welcoming & supportive to newcomers
- significant overlap between EasyBuild & Lmod communities
- active mailing list: *https://lists.ugent.be/wws/info/easybuild*
- active IRC (`#easybuild` on FreeNode) & Slack channel (`easybuild.slack.com`)
- users are also contributing: bug reports, feature requests, code contributions, ...

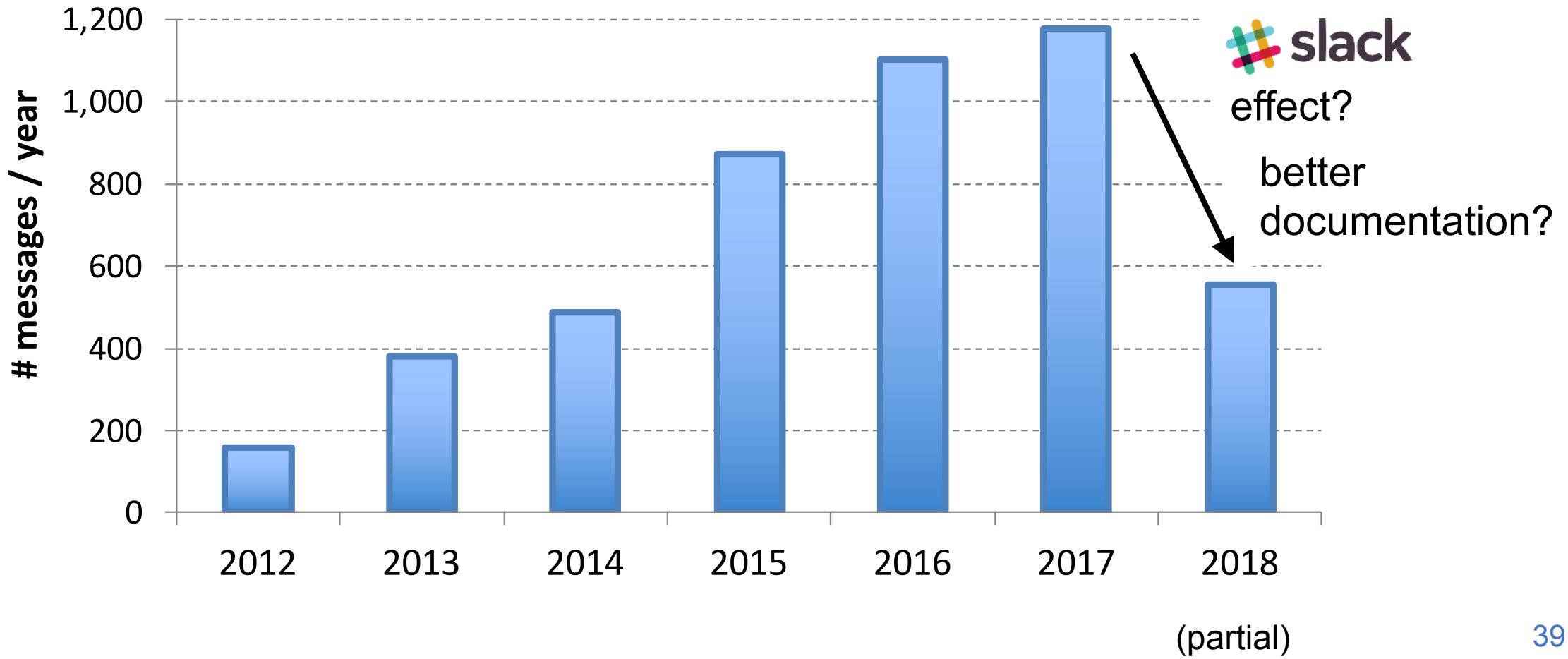
EasyBuild mailing list (subscribers)

Over 250 people subscribed to EasyBuild mailing list, still growing (but slower than before).



EasyBuild mailing list (traffic)

Traffic on EasyBuild mailing list peaked in 2017, more people getting help via other channels?

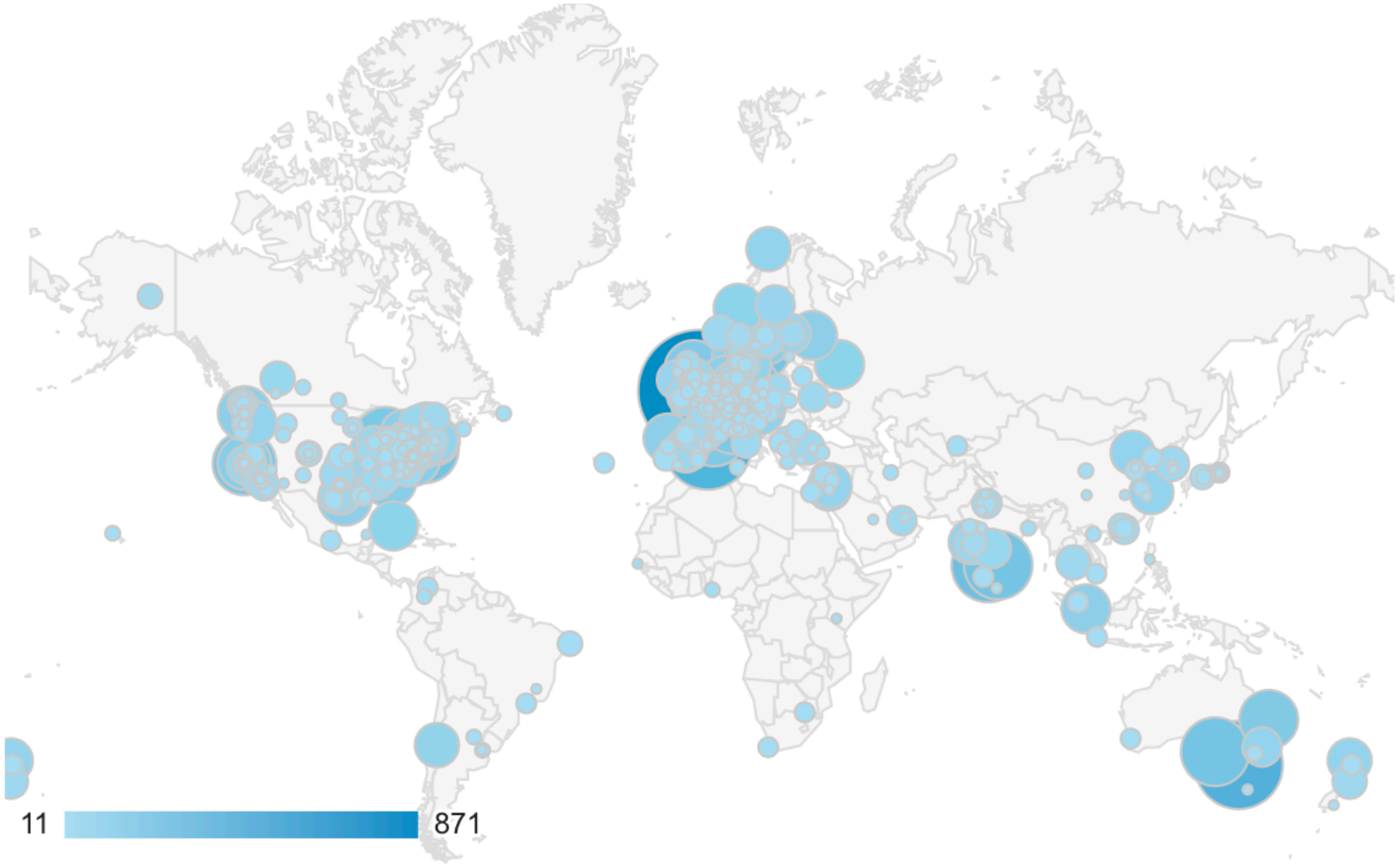


EasyBuild documentation visits

EasyBuild documentation at https://easybuild.readthedocs.io hitting **~500 weekly visitors**

(source: Google Analytics)

The sun never sets on EasyBuild...

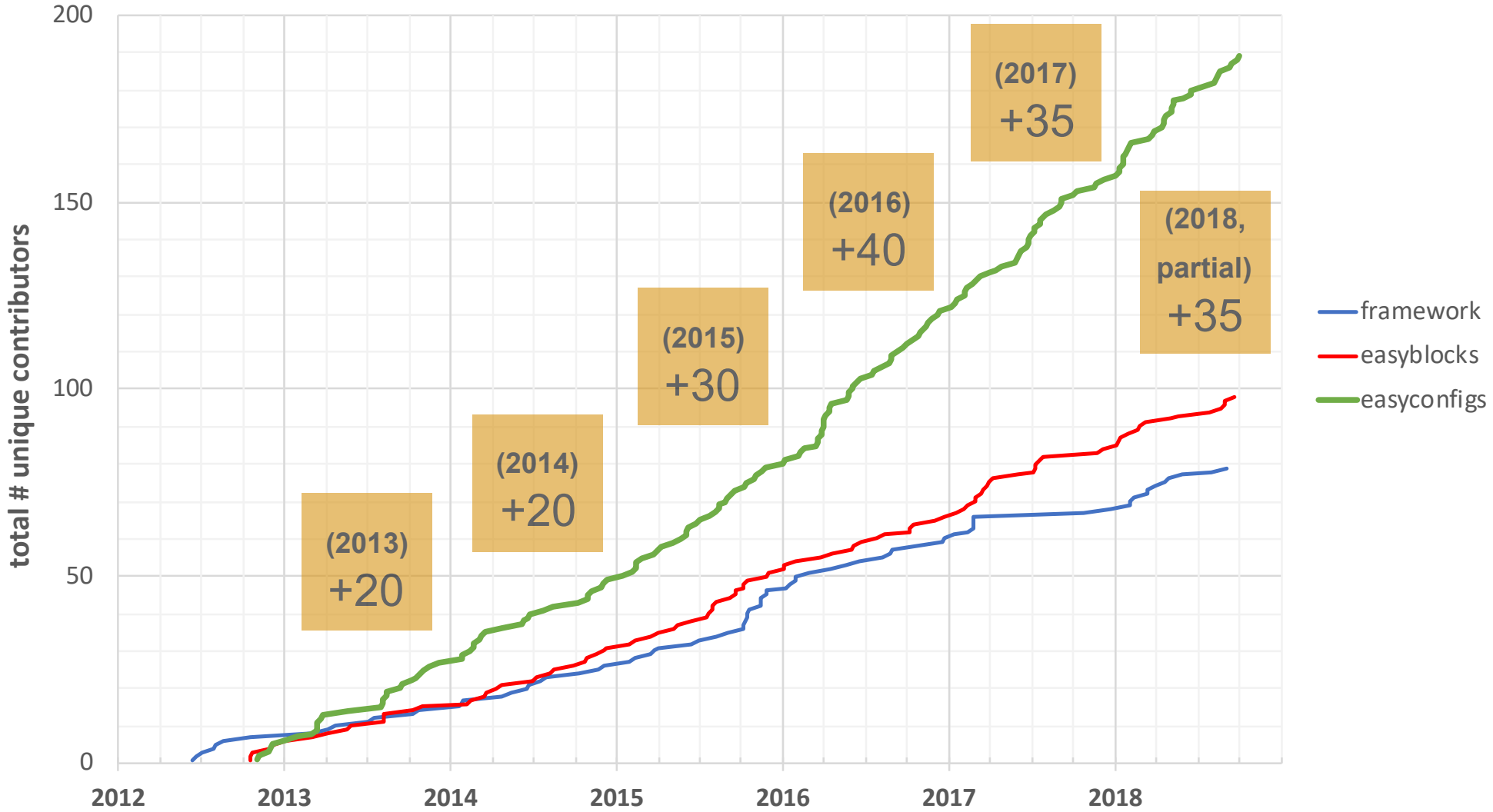


cities from where https://easybuild.readthedocs.io was visited at least 10 times during the last year

(source: Google Analytics)

Unique contributors (total, per repository)

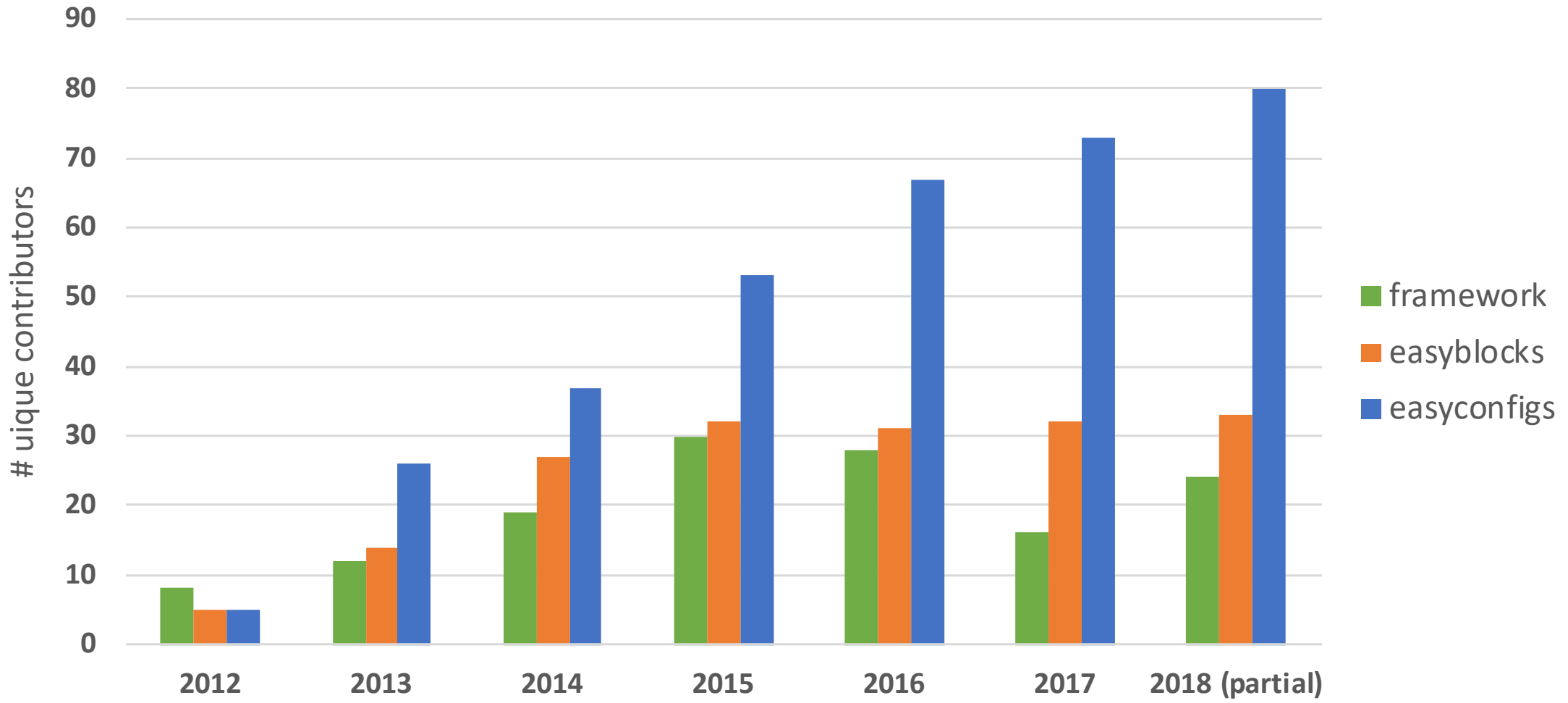
Getting close to 200 unique contributors for easyconfigs, 100 for easyblocks.



Unique contributors (per repository, per year)

Group of easyconfigs contributors is growing every year.

Stable group of easyblocks contributors, fluctuating a bit for framework.



Community (common) toolchains

http://easybuild.readthedocs.io/en/latest/Common-toolchains.html

- **`intel` and `foss`[1] toolchains** are most commonly used in EasyBuild community
- helps to focus efforts of HPC sites using one or both of these toolchains
- updated twice a year, clear versioning scheme: `<year>{a,b}` (`2017b`, `2018a`, ...)
- latest version:
  - `foss/2018b`

    binutils 2.30, GCC 7.3.0, OpenMPI 3.1.1, OpenBLAS 0.3.1 (incl. (Sca)LAPACK), FFTW 3.3.8
  - `intel/2018b`

    binutils 2.30 + GCC 7.3.0 as base

    Intel compilers 2018.3.222, Intel MPI 2018.3.222, Intel MKL 2018.3.222

(1) FOSS: Free and Open Source Software

VS

http://spack.rtfd.io/

- GPLv2 license
- requires Python 2(.6)
- targeted to HPC user support teams
- stable releases since Nov'12
- supports >1.5k software pkgs + ~1k ext.
- fixed dependency/toolchain versions
- *requires* a modules tool (Lmod/Tmod)
- framework, easyblocks (.py), easyconfigs (.eb)
- can do both `$LD_LIBRARY_PATH` & `RPATH`
- supports Linux (incl. Cray)

- ~~LGPL~~ MIT/Apache-2.0 dual-licensed
- compatible with Python 2.7 & Python 3.x
- (mainly) for scientific software developers
- alpha software (no stable/1.0 release yet)
- supports >2.5k software pkgs (incl. exts)
- very flexible dependency management
- compatible with Lmod/Tmod, dotkit
- core library + packages (.py), specs via CLI
- only does `RPATH`
- supports Linux (incl. Cray) & macOS

Both are backed by an active & supportive community!

Other features that were not covered...

http://easybuild.readthedocs.io

- letting users manage their software stack on top of centrally provided modules
- installing hidden modules, hiding certain dependencies & toolchains
- support for using RPATH linking
- partial installations: only (re)generate module file, install additional extensions
- tracing progress via '`eb --trace`'
- submitting installations as jobs to an HPC cluster via `--job` (distributed installation!)
- creating packages (RPMs, ...) for software installations done with EasyBuild
- (experimental) integration with Docker & Singularity for creating container images
- using EasyBuild on Cray systems, integration with Cray Programming Environment

Future work

- **more automation for incoming PRs to lower burden on maintainers**
- better error reporting
- stable integration with Docker & Singularity
- more manpower, dedicated funding (via EuroHPC project?)
- auto-updating easyconfigs to latest available version (incl. deps & extensions)
- slightly more flexibility w.r.t. dependency versions (e.g., allow "CMake 3.*")
- getting rid of dependency on `setuptools` (too much of a PITA...)
- support for using EasyBuild on top of Python 3 (EasyBuild 4.0)

Papers on EasyBuild (& Lmod)

Modern Scientific Software Management Using EasyBuild and Lmod

Markus Geimer (JSC), Kenneth Hoste (HPC-UGent), Robert McLay (TACC)

http://easybuilders.github.io/easybuild/files/hust14_paper.pdf

Making Scientific Software Installation Reproducible On Cray Systems Using EasyBuild

Petar Forai (IMP), Guilherme Peretti-Pezzi (CSCS), Kenneth Hoste (HPC-UGent)

https://cug.org/proceedings/cug2016_proceedings/includes/files/pap145.pdf

Scientific Software Management in Real Life:
Deployment of EasyBuild on a Large Scale System

Damian Alvarez, Alan O'Cais, Markus Geimer (JSC), Kenneth Hoste (HPC-UGent)

http://easybuilders.github.io/easybuild/files/eb-jsc-hust16.pdf

4th EasyBuild User Meeting

Wed 30 Jan - Fri Feb 1st 2019

Louvain-la-Neuve, Belgium

(Near Brussels)

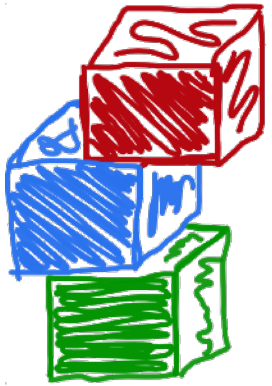
https://tiny.cc/eum19

close to FOSDEM 2019

Sat-Sun Feb 2-3 2019

https://fosdem.org/2019

devrooms on HPC, Big Data & Science,
Quantum Computing, ML on Code



easybuild

Questions?

kenneth.hoste@ugent.be
easybuild@lists.ugent.be

http://easybuilders.github.io/easybuild
http://easybuild.readthedocs.io

Contributing to EasyBuild

http://easybuild.readthedocs.io/en/latest/Contributing.html

EasyBuild has improved significantly thanks to the community.

You too can contribute back, by:

- sending feedback
- reporting bugs
- joining the discussion (mailing lists, IRC/Slack, EasyBuild conf calls)
- sharing suggestions/ideas for enhancements & additional features
- contributing easyconfigs, enhancing easyblocks, adding support for new software...
- extending & enhancing documentation

Adding support for additional software

- for each software installation, an **easyconfig file** is required
  - defines easyconfig parameters that specify to EasyBuild what to install, and how
  - existing easyconfig files can serve as examples
  - for version or toolchain updates, a tweaked easyconfig can be *generated* via `eb --try-*`
  - see *https://easybuild.readthedocs.io/en/latest/Writing_easyconfig_files.html*
- for 'standard' installation procedures, a **generic easyblock** can be used
  - installation can be controlled where needed via easyconfig parameters
- for custom installation procedures, a **software-specific easyblock** is required
  - see *https://easybuild.readthedocs.io/en/latest/Implementing-easyblocks.html*

Common generic easyblocks

http://easybuild.readthedocs.io/en/latest/version-specific/generic_easyblocks.html

- **`ConfigureMake`**
  standard '`./configure`' - '`make`' - '`make install`' installation procedure
- **`CMakeMake`**
  same as ConfigureMake, but using *CMake* for configuring
- **`PythonPackage`**
  installing Python packages ('`python setup.py install`', '`pip install`', ...)
- **`MakeCp`**
  no (standard) configuration step, build with '`make`', install by copying binaries/libraries
- **`Tarball`**: just unpack sources and copy everything to installation directory
- **`Binary`**: run binary installer (specified via '`install_cmd`' easyconfig parameter)

Easyconfig files vs easyblocks

http://easybuild.readthedocs.io/en/latest/Implementing-easyblocks.html

- thin line between using custom easyblock and 'fat' easyconfig with generic easyblock
- custom easyblocks are "do once and forget", **central solution to build peculiarities**
- reasons to consider implementing a software-specific easyblock include:
  - 'critical' values for easyconfig parameters required to make installation succeed
  - toolchain-specific aspects of the build and installation procedure (e.g., configure options)
  - interactive commands that need to be run
  - custom (configure) options for dependencies
  - having to create or adjust specific (configuration) files
  - 'hackish' usage of a generic easyblock

git can be a major hurdle for contributors

- Potential contributors may not be familiar (yet) with git...
- Do they ***have*** to learn git first before they can contribute?

So you want to contribute to easybuild ...

To contribute one or more easyconfig files,
you first need to prepare:

1) create a GitHub account via Sign up for GitHub

2) Fork https://github.com/easybuilders/easybuild-easyconfigs

3)
```
git clone git@github.com:YOU/easybuild-easyconfigs.git
```
(change this bit!)

Branch, stage, commit, push

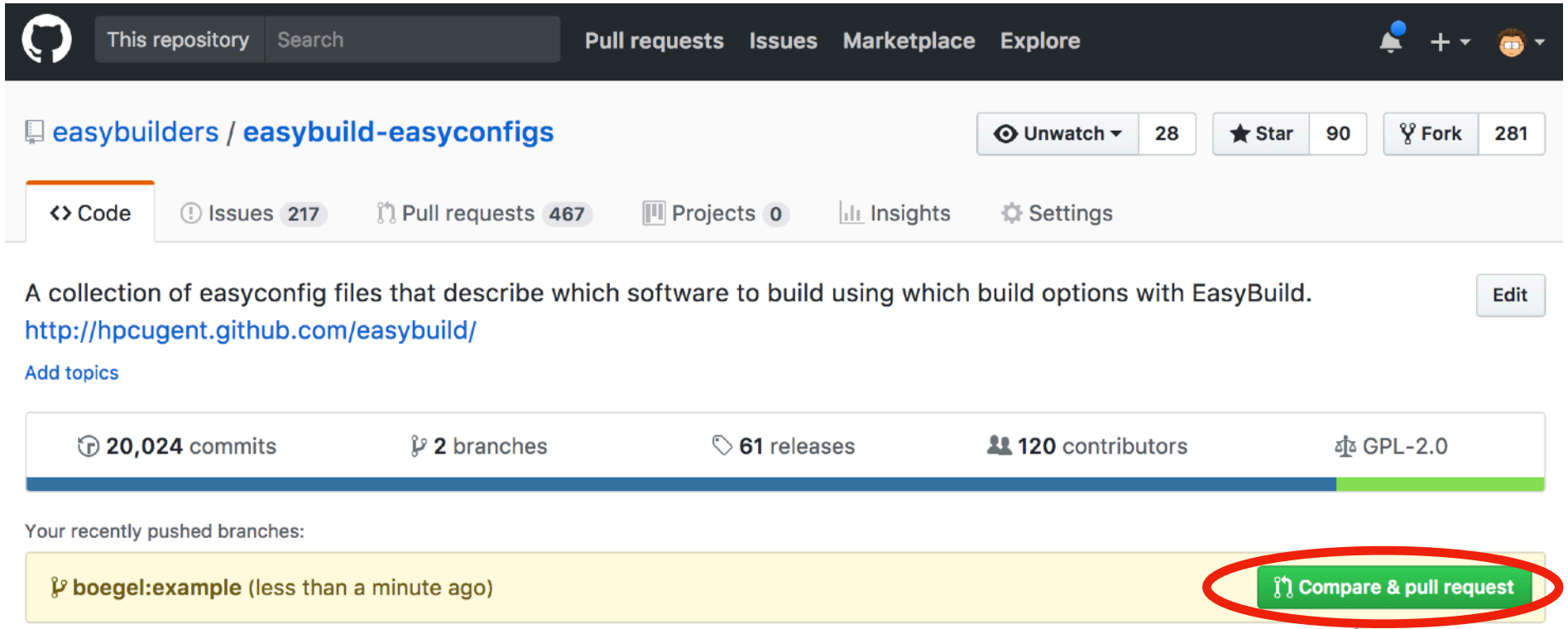
Once you have cloned the easyconfigs repository, you can really get to work...

```
# create and check out a new branch
$ git checkout -b funnybranchname

# stage the new/changed easyconfig file(s)
$ git add easybuild/easyconfigs/s/scikit-learn/*.eb

# commit the changes with a sensible commit message
$ git commit -m "update scikit-learn to 0.19.1"

# push your branch to your fork on GitHub
$ git push origin funnybranchname
```

(this may be different in your setup, see `git remote -v`)

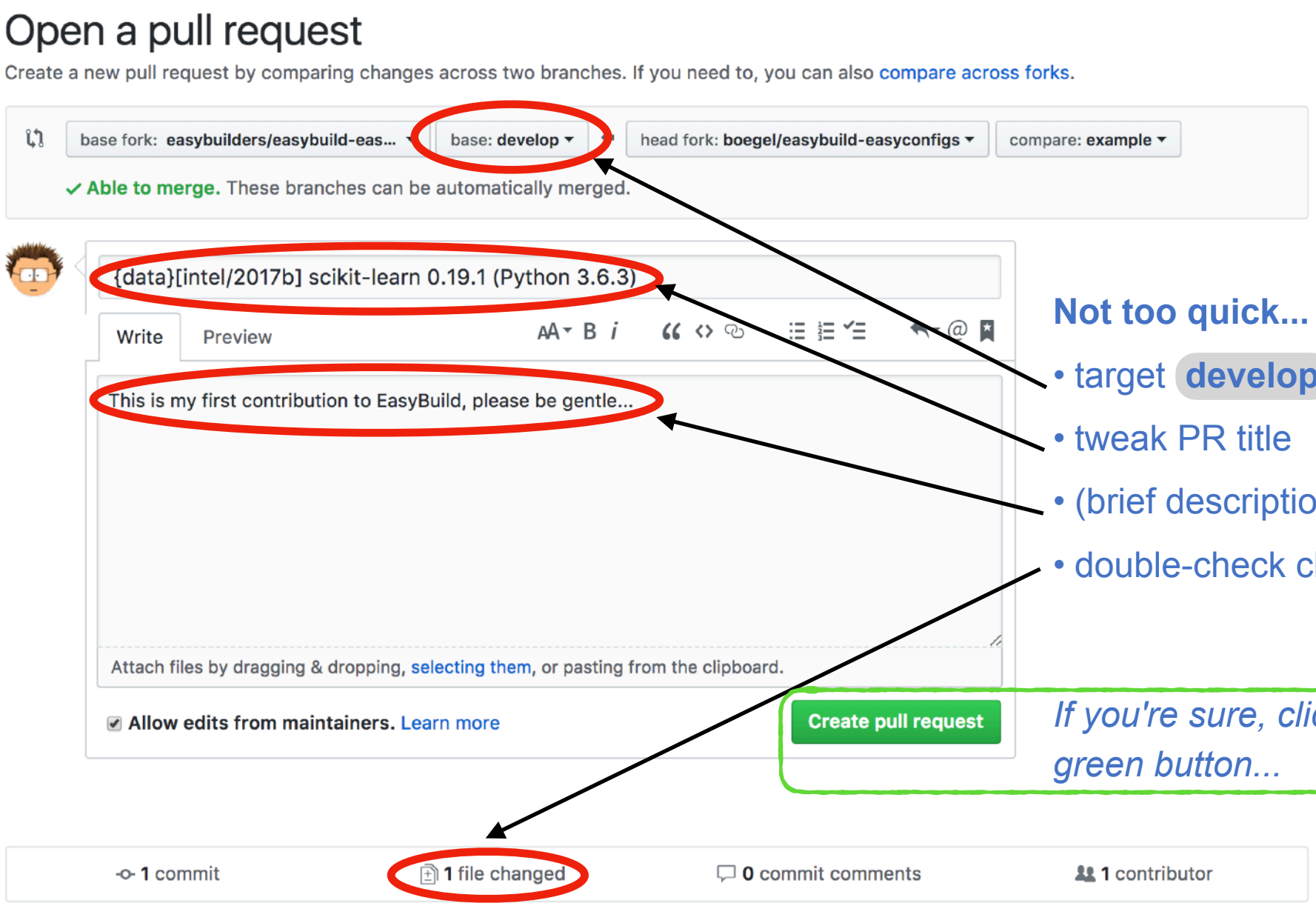
Clickety, clickety, click.

Next, log on to GitHub and open a pull request...



You can use this useful button , if you're quick enough!

Almost there...



Not too quick...

- target **develop**
- tweak PR title
- (brief description)
- double-check changed files

If you're sure, click the shiny green button...

Are we done yet?

- once your pull request is open, your changes will be:
  - quickly tested by Travis CI (automagically)
  - reviewed by a maintainer
  - thoroughly tested by at least one person
- additional changes may be requested...
- which means you may have to go back to git...
- maybe even multiple times... (spread over multiple hours/days)

Fine-tuning your contribution

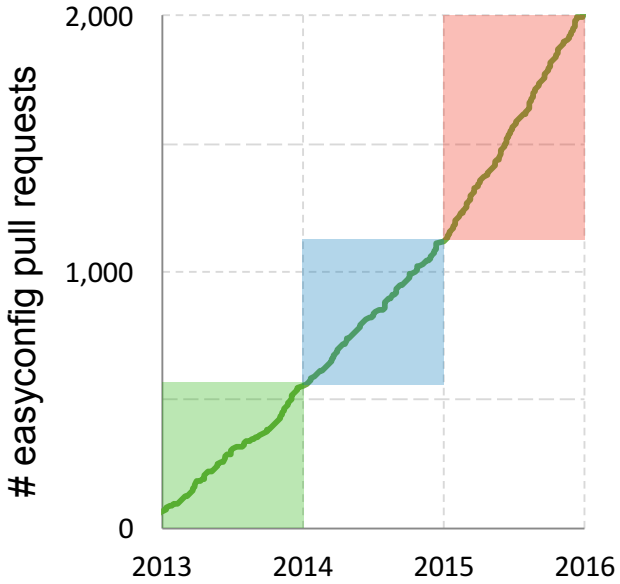
Do you still remember the funny branch name you used for that PR?

```
# check out your branch again
$ git checkout funnybranchname

# edit some files, as requested by the friendly reviewer
$ vim ...

# commit (all) your changes
$ git commit -am "fixed suggestions by reviewer"

# push your updated branch to GitHub to update the PR
$ git push origin funnybranchname
```

Too much overhead, manual steps...

- *"Isn't there an easier/more efficient way for doing this?!"*
  - need to come up with a sensible funny branch name first...
  - 4 git commands to prepare the contribution
  - at least 4 (well aimed) *mouse clicks* to open the pull request
  - if requested, N x 3 more git commands for fine-tuning the PR
  - + a bit of cleanup afterwards
    (delete branch both in local repository clone and on GitHub)
- All of this assuming you:
  - take into account the project policies
  - don't mess up with git at any time...

Contributions to easybuild



- more incoming contributions in 2015
  - ~500 easyconfig PRs in 2013 & 2014
  - almost double that in 2015...
- several people were struggling with the contribution procedure
  - mainly because they were not sufficiently familiar with git
  - most candidate contributors were not software developers
  - resulted in a lot of frustration & wasted time
  - some people even gave up on contributing :(
- quite annoying for a project that aims to *automate* things...

GitHub integration in 

https://easybuild.readthedocs.io/en/latest/Integration_with_GitHub.html

- goals:
  - **automate contribution workflow**
  - avoid direct interaction with git and 

- initial implementation added support for:
  - opening new PRs & updating existing PRs `eb --new-pr` `eb --update-pr`
  - downloading & using easyconfigs from a PR `eb --from-pr`
  - uploading test reports to a PR `eb --upload-test-report`
- limited to easyconfig files, for now...
  - but implemented such that it is easy to extend to easyblocks, and even the EasyBuild framework itself

Under the covers

- **GitPython** (https://pypi.python.org/pypi/GitPython)
  - Python library to interact with git
- **GitHub REST API** (https://developer.github.com/v3)
  - to automate interaction with GitHub (less clickety, clickety)
- **`keyring`** Python package (https://pypi.python.org/pypi/keyring)
  - Python library that provides access to system keyring
  - required to securely store GitHub tokens

Opening new pull requests

eb --new-pr

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

one single 'eb' command
no 'git' commands, no clickety on GitHub

Opening new pull requests

`eb --new-pr`

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

repository is ***cloned*** to temporary directory,
develop branch is ***checked out*** as a base

Opening new pull requests

`eb --new-pr`

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

GitHub repository to push to

auto-derived branch name

a new branch is created & pushed
(you can specify the branch name yourself)

Opening new pull requests

`eb --new-pr`

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

the easyconfig file is ***renamed***
and ***moved to the right location***

Opening new pull requests

`eb --new-pr`

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

the pull request is prepared:
target develop branch, ***title & description***

Opening new pull requests

`eb --new-pr`

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

a high-level ***overview of the changes*** is shown (similar to output of '`git stat`')

Opening new pull requests

`eb --new-pr`

```
$ eb --new-pr sklearn.eb
== temporary log file in case of crash /tmp/eb-Iv07fs/easybuild-inS8Ny.log
== fetching branch 'develop' from
   https://github.com/easybuilders/easybuild-easyconfigs.git...
== copying easyconfigs to
   /tmp/eb-Iv07fs/git-working-dirhYsAA_/easybuild-easyconfigs...

Opening pull request
* target: easybuilders/easybuild-easyconfigs:develop
* from: boegel/easybuild-easyconfigs:20180125211924_new_pr_scikit-learn0191
* title: "{data}[intel/2017b] scikit-learn v0.19.1"
* description:
"""
(created using `eb --new-pr`)

"""
* overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 31 ++++++++++++
 1 file changed, 31 insertions(+)

Opened pull request: https://github.com/easybuilders/easybuild-easyconfigs/pull/12345
== Temporary log file(s) /tmp/eb-Iv07fs/easybuild-inS8Ny.log* have been removed.
== Temporary directory /tmp/eb-Iv07fs has been removed.
```

the ***pull request is opened automatically***, using the GitHub API & token

Opening a pull request is now... easy!


```
$ mv sklearn.eb scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb
$ mv scikit*.eb easybuild/easyconfigs/s/scikit-learn
$ git checkout develop && git pull upstream develop
$ git checkout -b scikit_learn_0191_intel_2017b
$ git add easybuild/easyconfigs/s/scikit-learn
$ git commit -m "{data}[intel/2017b] scikit-learn v0.19.1"
$ git push origin scikit_learn_0191_intel_2017b
```

+ log into GitHub to actually open the pull request (clickety, clickety...)

one single `eb` command

no `git` commands

no GitHub interaction

metadata is automatically derived from easyconfig

saves a lot of time!

```
eb --new-pr sklearn.eb
```

Updating existing pull requests

`eb --update-pr`

to update a PR: provide PR#, updated files and commit message

```
$ eb --update-pr 12345 sklearn.eb --pr-commit-msg "minor style fixes"
== temporary log file in case of crash /tmp/eb-EKe1WF/easybuild-w1Ycat.log
== Determined branch name corresponding to easybuilders/easybuild-easyconfigs
PR #12345: 20180125211924_new_pr_scikit-learn0191
== fetching branch '20180125211924_new_pr_scikit-learn0191' from https://
github.com/boegel/easybuild-easyconfigs.git...
== copying easyconfigs to /tmp/eb-EKe1WF/git-working-dirbIzGHn/easybuild-
easyconfigs...
Overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Updated easybuilders/easybuild-easyconfigs PR #12345 by pushing to branch
boegel/20180125211924_new_pr_scikit-learn0191
== Temporary log file(s) /tmp/eb-EKe1WF/easybuild-w1Ycat.log* have been removed.
== Temporary directory /tmp/eb-EKe1WF has been removed.
```

Updating existing pull requests

`eb --update-pr`

branch that corresponds with PR is determined *automatically*

```
$ eb --update-pr 12345 sklearn.eb --pr-commit-msg "minor style fixes"
== temporary log file in case of crash /tmp/eb-EKe1WF/easybuild-w1Ycat.log
== Determined branch name corresponding to easybuilders/easybuild-easyconfigs
PR #12345: 20180125211924_new_pr_scikit-learn0191
== fetching branch '20180125211924_new_pr_scikit-learn0191' from https://
github.com/boegel/easybuild-easyconfigs.git...
== copying easyconfigs to /tmp/eb-EKe1WF/git-working-dirbIzGHn/easybuild-
easyconfigs...
Overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Updated easybuilders/easybuild-easyconfigs PR #12345 by pushing to branch
boegel/20180125211924_new_pr_scikit-learn0191
== Temporary log file(s) /tmp/eb-EKe1WF/easybuild-w1Ycat.log* have been removed.
== Temporary directory /tmp/eb-EKe1WF has been removed.
```

Updating existing pull requests

`eb --update-pr`

PR branch is downloaded to temporary directory,
updated files are copied

```
$ eb --update-pr 12345 sklearn.eb --pr-commit-msg "minor style fixes"
== temporary log file in case of crash /tmp/eb-EKe1WF/easybuild-w1Ycat.log
== Determined branch name corresponding to easybuilders/easybuild-easyconfigs
PR #12345: 20180125211924_new_pr_scikit-learn0191
== fetching branch '20180125211924_new_pr_scikit-learn0191' from https://
github.com/boegel/easybuild-easyconfigs.git...
== copying easyconfigs to /tmp/eb-EKe1WF/git-working-dirbIzGHn/easybuild-
easyconfigs...
Overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Updated easybuilders/easybuild-easyconfigs PR #12345 by pushing to branch
boegel/20180125211924_new_pr_scikit-learn0191
== Temporary log file(s) /tmp/eb-EKe1WF/easybuild-w1Ycat.log* have been removed.
== Temporary directory /tmp/eb-EKe1WF has been removed.
```

Updating existing pull requests

`eb --update-pr`

updated files are renamed and moved to right location

```
$ eb --update-pr 12345 sklearn.eb --pr-commit-msg "minor style fixes"
== temporary log file in case of crash /tmp/eb-EKe1WF/easybuild-w1Ycat.log
== Determined branch name corresponding to easybuilders/easybuild-easyconfigs
PR #12345: 20180125211924_new_pr_scikit-learn0191
== fetching branch '20180125211924_new_pr_scikit-learn0191' from https://
github.com/boegel/easybuild-easyconfigs.git...
== copying easyconfigs to /tmp/eb-EKe1WF/git-working-dirbIzGHn/easybuild-
easyconfigs...
Overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb  2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Updated easybuilders/easybuild-easyconfigs PR #12345 by pushing to branch
boegel/20180125211924_new_pr_scikit-learn0191
== Temporary log file(s) /tmp/eb-EKe1WF/easybuild-w1Ycat.log* have been removed.
== Temporary directory /tmp/eb-EKe1WF has been removed.
```

Updating existing pull requests

`eb --update-pr`

changes are committed, high-level overview is shown (cfr. '`git stat`')

```
$ eb --update-pr 12345 sklearn.eb --pr-commit-msg "minor style fixes"
== temporary log file in case of crash /tmp/eb-EKe1WF/easybuild-w1Ycat.log
== Determined branch name corresponding to easybuilders/easybuild-easyconfigs
PR #12345: 20180125211924_new_pr_scikit-learn0191
== fetching branch '20180125211924_new_pr_scikit-learn0191' from https://
github.com/boegel/easybuild-easyconfigs.git...
== copying easyconfigs to /tmp/eb-EKe1WF/git-working-dirbIzGHn/easybuild-
easyconfigs...
Overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Updated easybuilders/easybuild-easyconfigs PR #12345 by pushing to branch
boegel/20180125211924_new_pr_scikit-learn0191
== Temporary log file(s) /tmp/eb-EKe1WF/easybuild-w1Ycat.log* have been removed.
== Temporary directory /tmp/eb-EKe1WF has been removed.
```

Updating existing pull requests

`eb --update-pr`

modified branch is pushed to GitHub to update the PR

```
$ eb --update-pr 12345 sklearn.eb --pr-commit-msg "minor style fixes"
== temporary log file in case of crash /tmp/eb-EKe1WF/easybuild-w1Ycat.log
== Determined branch name corresponding to easybuilders/easybuild-easyconfigs
PR #12345: 20180125211924_new_pr_scikit-learn0191
== fetching branch '20180125211924_new_pr_scikit-learn0191' from https://
github.com/boegel/easybuild-easyconfigs.git...
== copying easyconfigs to /tmp/eb-EKe1WF/git-working-dirbIzGHn/easybuild-
easyconfigs...
Overview of changes:
 .../s/scikit-learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Updated easybuilders/easybuild-easyconfigs PR #12345 by pushing to branch
boegel/20180125211924_new_pr_scikit-learn0191
== Temporary log file(s) /tmp/eb-EKe1WF/easybuild-w1Ycat.log* have been removed.
== Temporary directory /tmp/eb-EKe1WF has been removed.
```

Updating a PR in 1, ~~2, 3~~

First, figure out the (funny) branch name corresponding to the PR...

```
$ git checkout scikit_learn_0191_intel_2017b
$ cd easybuild/easyconfigs/s/scikit-learn
$ vim scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb
$ git add scikit*.eb
$ git commit -m "fix typo"
$ git push origin scikit_learn_0191_intel_2017b
```

one single `eb` command

no `git` commands

no GitHub interaction

no local branch to clean up once PR is merged

saves a lot of time!

```
eb --update-pr 12345 sklearn.eb --pr-commit-msg "fix typo"
```

Using easyconfigs from a PR

`eb --from-pr`

- no need to manually download easyconfigs to feed them to `eb`
- PR is (virtually) merged with current develop branch in /tmp
- very useful when testing contributions!

```
$ eb --from-pr 12345
== temporary log file in case of crash /tmp/eb-kQRw7W/easybuild-HU4hce.log
== processing EasyBuild easyconfig /tmp/eb-kQRw7W/files_pr12345/s/scikit-
learn/scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb
== building and installing scikit-learn/0.19.1-intel-2017b-Python-3.6.3...
== fetching files...
== creating build dir, resetting environment...
== unpacking...
== patching...
== preparing...
== configuring...
== building...
== testing...
== installing...
```

Uploading test reports to a PR

`eb --upload-test-report`

- combined with **`--from-pr`** to test contribution and report results
- test report with info on host & environment is uploaded as a gist
- a comment is added in the PR to trigger a notification by GitHub

```
$ eb --from-pr 12345 --upload-test-report
...
== building and installing scikit-learn/0.19.1-intel-2017b-Python-3.6.3...
...
== COMPLETED: Installation ended successfully
...
Adding comment to easybuild-easyconfigs issue #12345: 'Test report by @boegel
**SUCCESS**
Build succeeded for 1 out of 1 (1 easyconfigs in this PR)
<hostname> - Linux CentOS 7.5, Intel Xeon CPU E5-2680 v3 @ 2.50GHz, Python 2.7.5
See https://gist.github.com/3c...58 for a full test report.'
== Test report uploaded to https://gist.github.com/3c...58 and mentioned in a
comment in easyconfigs PR#12345
== Build succeeded for 1 out of 1
```

Uploading test reports to a PR

```
eb --upload-test-report
```

boegel commented on 21 Nov 2017 — Owner

Test report by **@boegel**
SUCCESS
Build succeeded for 1 out of 1 (1 easyconfigs in this PR)
node2403.golett.os - Linux centos linux 7.4.1708, Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz, Python 2.7.5
See https://gist.github.com/2ce867c657e05a8717978f3a73b9e019 for a full test report.

verdurin commented 3 days ago — Member

Test report by **@verdurin**
FAILED
Build succeeded for 0 out of 1 (1 easyconfigs in this PR)
rescomp2 - Linux centos 6.8, Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz, Python 2.7.11
See https://gist.github.com/0dd5b4a3338a1350fe3b92b247bf84bb for a full test report.

Uploading test reports to a PR

```
eb --upload-test-report
```

EasyBuild test report for easyconfigs PR #5375

easybuild_test_report_easyconfigs_pr5375_20175921-UTC-10-59-58.md

Test report for https://github.com/easybuilders/easybuild-easyconfigs/pull/5375

Test result

Build succeeded for 1 out of 1 (1 easyconfigs in this PR)

Overview of tested easyconfigs (in order)

- SUCCESS *scikit-learn-0.19.1-intel-2017b-Python-3.6.3.eb*

Time info

- start: Tue, 21 Nov 2017 10:57:11 +0000 (UTC)
- end: Tue, 21 Nov 2017 10:59:58 +0000 (UTC)

Uploading test reports to a PR

```
eb --upload-test-report
```

EasyBuild test report for easyconfigs PR #5719

easybuild_test_report_easyconfigs_pr5719_20180124-UTC-18-01-27.md

Test report for https://github.com/easybuilders/easybuild-easyconfigs/pull/5719

Test result

Build succeeded for 0 out of 1 (1 easyconfigs in this PR)

Overview of tested easyconfigs (in order)

- **FAIL (build issue)** *RSEM-1.3.0-intel-2017b.eb* (partial log available at https://gist.github.com/7d9e48b400232b5fa0924e2bf28d240a)

a separate gist provides partial build log

Time info

- start: Wed, 24 Jan 2018 18:01:18 +0000 (UTC)

Checking configuration for GitHub integration

`eb --check-github`

```
$  eb --check-github

Checking status of GitHub integration...

Making sure we're online...OK

* GitHub user... boegel => OK
* GitHub token... e87..3b1 (len: 40) => OK (validated)
* git command... OK ("git version 1.8.3.1")
* GitPython module... OK (GitPython version 2.1.1)
* push access to boegel/easybuild-easyconfigs repo @ GitHub... OK
* creating gists... OK
* location to Git working dirs... OK (/home/example)

All checks PASSed!

Status of GitHub integration:
* --from-pr: OK
* --new-pr: OK
* --review-pr: OK
* --update-pr: OK
* --upload-test-report: OK
```

various **checks** are performed to verify whether required stuff is in place

Checking configuration for GitHub integration

`eb --check-github`

```
$  eb --check-github

Checking status of GitHub integration...

Making sure we're online...OK

* GitHub user... boegel => OK
* GitHub token... e87..3b1 (len: 40) => OK (validated)
* git command... OK ("git version 1.8.3.1")
* GitPython module... OK (GitPython version 2.1.1)
* push access to boegel/easybuild-easyconfigs repo @ GitHub... OK
* creating gists... OK
* location to Git working dirs... OK (/home/example)

All checks PASSed!

Status of GitHub integration:
* --from-pr: OK
* --new-pr: OK
* --review-pr: OK
* --update-pr: OK
* --upload-test-report: OK
```

overview of **what is supported** is produced based on result of checks

Reviewing pull requests


```
eb --review-pr
```

- reviewer should ideally also compare new easyconfig files with existing ones for the same software
    - just a version bump or different toolchain?
    - also other changes?
    - to check whether fixes that are already in place in develop branch are not overlooked
- this goes beyond what GitHub can do
- quite tedious to do manually...
- what are the most relevant easyconfigs to compare with?
    - same software (major) version, same toolchain (version), ...

Previewing & reviewing pull requests

`eb --preview-pr` **`eb --review-pr`**

- (multi)diff between contributed easyconfig and existing one(s)
- can be used both to *preview* a contribution and to *review* a PR

```
$ eb --review-pr 5736

Comparing BamTools-2.5.1-intel-2017b.eb with
BamTools-2.5.0-intel-2017b.eb
=====
2 - version = '2.5.0' (1/1)
2 + version = '2.5.1' (1/1)

-----

11 - checksums = ['dd4185bdba6e3adf2c24b7f93a57233d'] (1/1)
11 + checksums = ['98e90632058f85bd5eed6088b3ff912e'] (1/1)
```

removed

added

comparison is done with (most) relevant easyconfigs in current ***develop*** branch

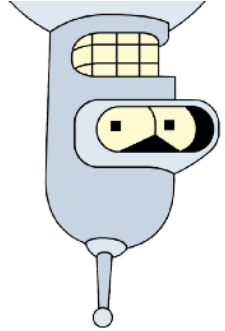
Merging pull requests

`eb --merge-pr`

verifies whether PR is **eligible** to be merged, and merges it if so

```
$ eb --merge-pr 5722

easybuilders/easybuild-easyconfigs PR #5722 was submitted by vanzod,
you are using GitHub account 'boegel'

Checking eligibility of easybuilders/easybuild-easyconfigs PR #5722
for merging...
* targets develop branch: OK
* test suite passes: OK
* last test report is successful: OK
* approved review: OK (by boegel)
* milestone is set: OK (3.6.0)

Review OK, merging pull request!

Adding comment to easybuild-easyconfigs issue #5722:
'Going in, thanks @vanzod!'
```

merging pull requests can be done ***from the command line*** *(after approved visual review)*



boegelbot

- bot that creates comments in PRs if tests fail in Travis
- mainly to trigger GitHub to send notifications
- see https://github.com/boegel/boegelbot

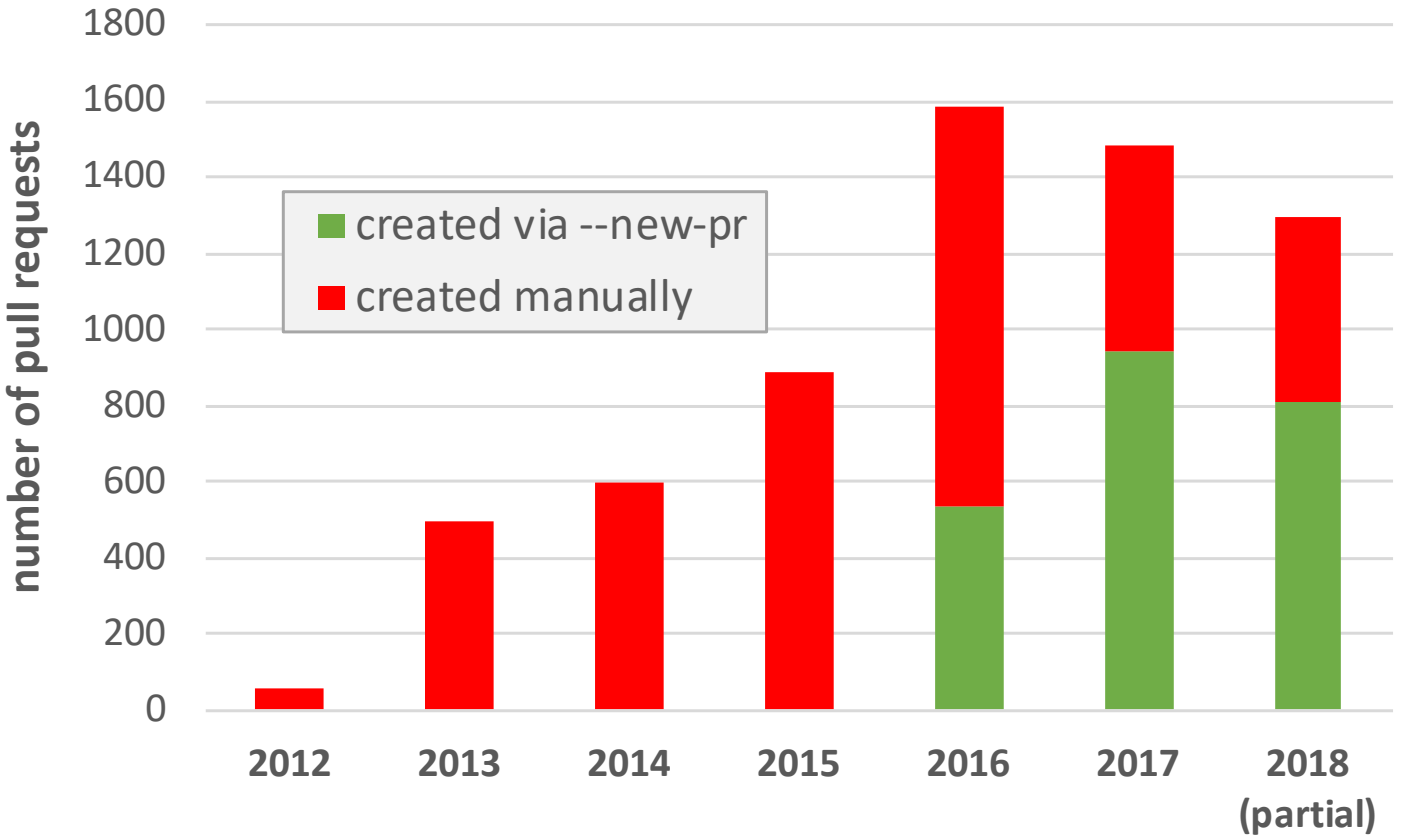
boegelbot commented on 8 Sep 2017

Travis test report: 8/8 runs failed - see https://travis-ci.org/easybuilders/easybuild-easyconfigs/builds/273198653

Only showing partial log for 1st failed test suite run 5974.1;
full log at https://travis-ci.org/easybuilders/easybuild-easyconfigs/jobs/273198655

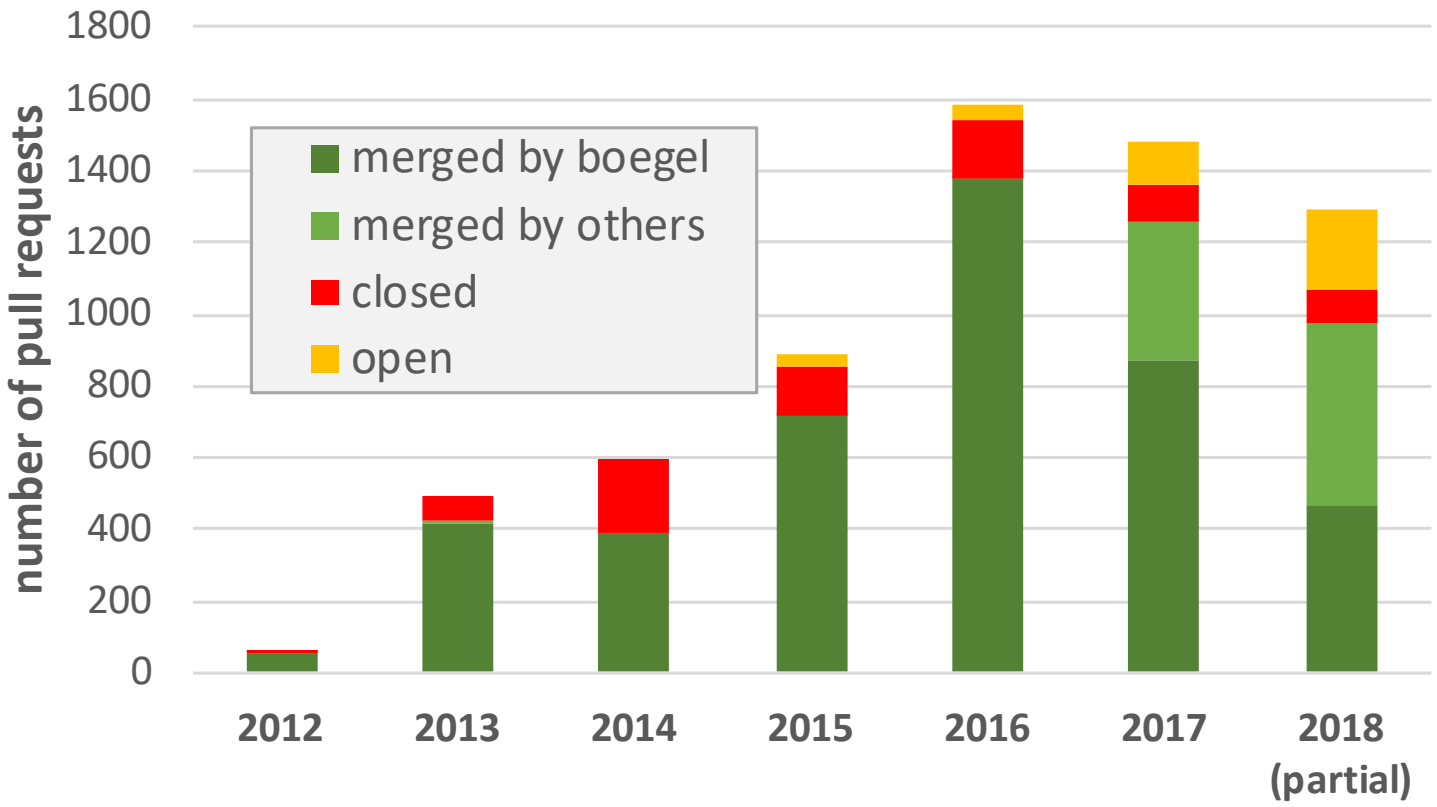
```
...
ERROR: Test for parsing of easyconfig Stata-15.eb
----------------------------------------------------------------------
Traceback (most recent call last):
  File "<string>", line 1, in innertest
  File "/home/travis/build/easybuilders/easybuild-easyconfigs/test/easyconfigs/easyconfigs.py", line
    ecs = process_easyconfig(spec)
```

Impact on contributions

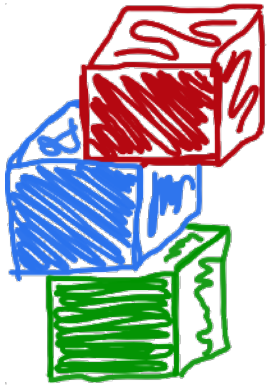


- > 500 PRs opened using eb `--new-pr` in 2016 (34% of total)
- almost 1000 PRs in 2017 (63% of total), similar ratio for 2018
- ~80% more PRs in 2016/2017 compared to 2015

Impact on contributions



- GitHub integration has helped (a lot) to keep pace with incoming PRs
- ~85% of PRs gets merged (there's usually a good reason for the ones that are not)
- would have been difficult without automation due to amount of PRs



easybuild

Questions?

kenneth.hoste@ugent.be
easybuild@lists.ugent.be

http://easybuilders.github.io/easybuild
http://easybuild.readthedocs.io