An introduction to EasyBuild (and Lmod)

Kenneth Hoste
HPC-UGent, Ghent University, Belgium
kenneth.hoste@ugent.be

Bayer CropScience, Ghent, Belgium
20150123

HPC-UGent in a nutshell

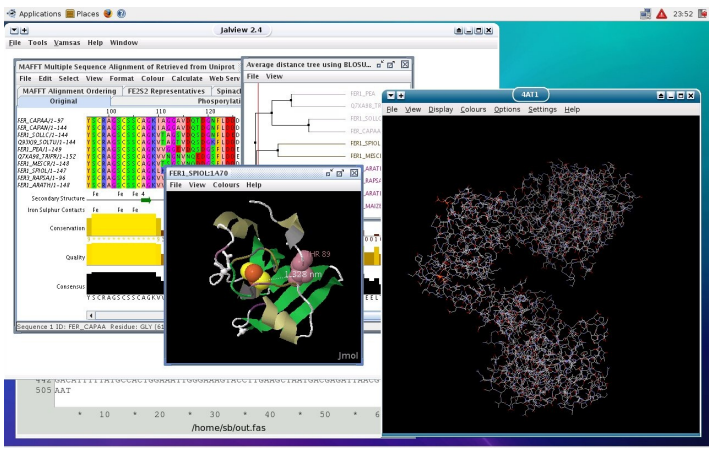
- HPC team at central IT dept. of Ghent University (Belgium)
- 9 team members: 1 manager, ~3 user support, ~5 sysadmin
- 6(+2) Tier2 clusters + one Tier1 (8.5k cores), >1k servers in total
- ~1.5k user accounts, across all scientific domains
- tasks: hardware, system administration, user support/training, ...

- member of Flemish Supercomputer Centre (VSC)
  virtual centre, collaboration between Flemish university associations

Scientists and software

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

Scientists are not software developers or system administrators (nor should they be).



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

"Please install this on the HPC?"

In the context of high performance computing, *building from source* should be preferred, when possible (when sources are available).

This allows for controlling used compilers and libraries, optimizing the software for the specific system architecture (e.g., AVX, network), etc.

Installing (lots of) scientific software is typically:

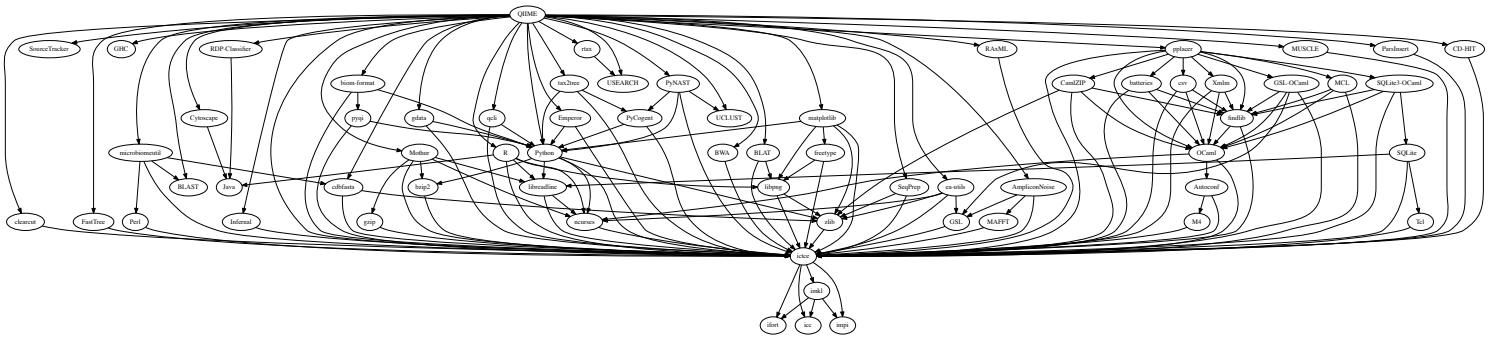
- error-prone, trial-and-error
- tedious, hard to get right
- repetitive & boring (well...)
- time-consuming (hours, days, even weeks)
- frustrating ("*Pandora's box*")
- ...

Common issues with scientific software

- use of non-standard build tools (or broken ones)
- incomplete build procedure, e.g., no configure/install step
- interactive installation scripts
- hardcoded parameters (compilers, libraries, paths, . . .)
- poor/outdated/missing/wrong documentation
- dependency (version) hell

Prime example: QIIME

QIIME: Quantitative Insights Into Microbial Ecology (`http://qiime.org/`)



- scientific research domain: bioinformatics ...
- 59 dependencies in total (*without* compiler toolchain), some optional
  - depends on Haskell (GHC), Java, Python, R, Perl, OCaml, ...
  - several deps use a non-standard build procedure (in various degrees)
- *very* picky about dependency versions (e.g., *must* be Python v2.7.*3*)
- took us several weeks to get it installed (like we wanted)...
- **...now we can (re)build/install it all with a single command!**

(disclaimer: QIIME is not supported yet in the latest EasyBuild release)

What about existing tools?

Existing tools are not well suited to scientific software and HPC systems.

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

Common problems:

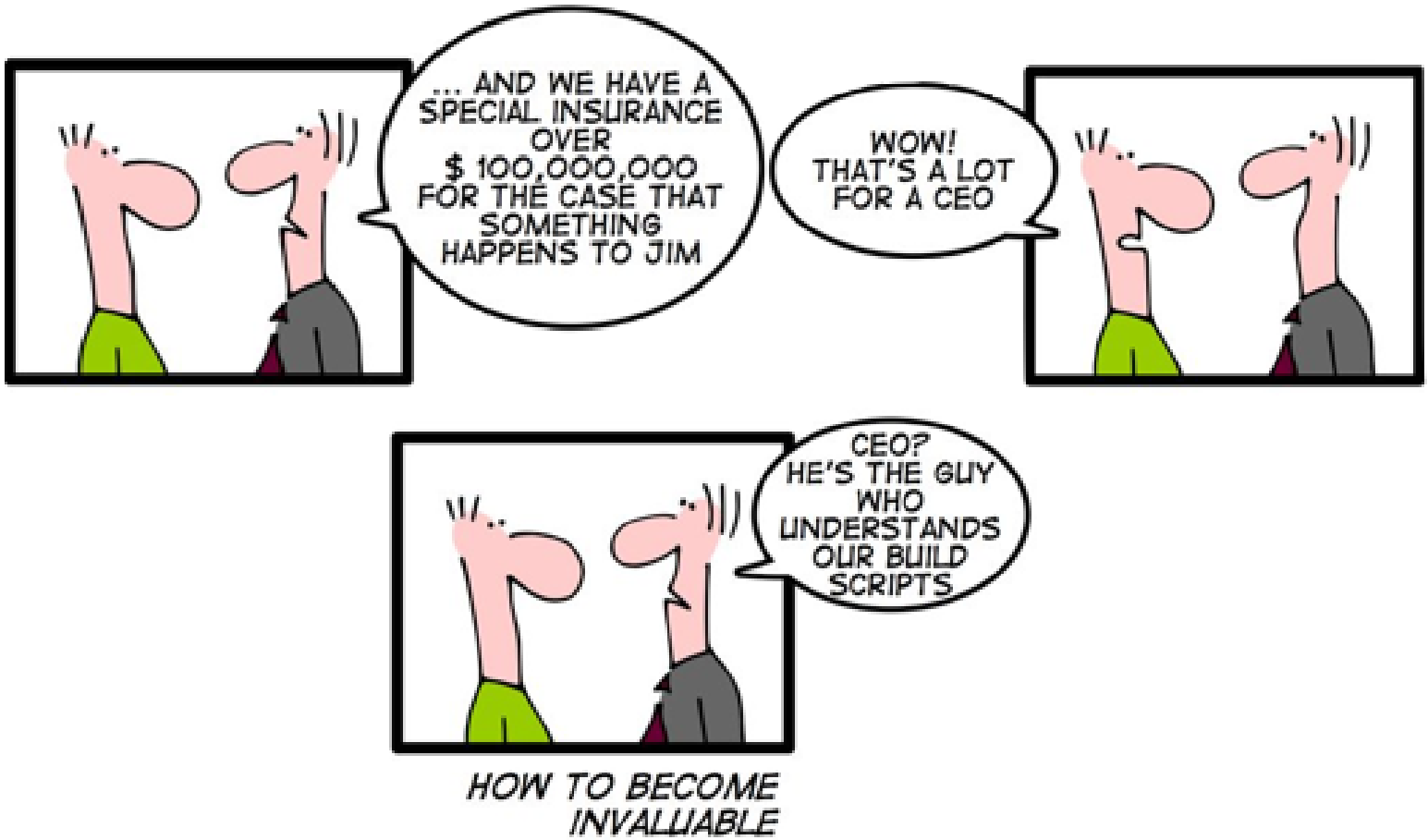
- poor support for multiple versions/builds existing side-by-side
- not flexible enough to deal with idiosyncrasies of scientific software
- hard to maintain (bash, heavy copy-pasting, ...)
- little support for scientific software, other compilers (not GCC), ...

Houston, we have a problem

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

- huge burden on HPC user support teams
- researchers lose time just to get stuff installed
- every site deals with it in its own way (scripting, . . .)
- very little collaboration among HPC sites :(

How to become invaluable



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

EasyBuild: building software with ease

`http://hpcugent.github.io/easybuild/`

- open source (GPLv2) framework for building and installing software
- collection of Python packages and modules
- original implementation by HPC-UGent, since 2009
- thriving community: actively contributing, driving development
- new release every 4–6 weeks (latest: EasyBuild v1.16.1, Dec'14)

  next release: EasyBuild v2.0.0!
- supports *over 550* different software packages

  including CP2K, NAMD, NWChem, OpenFOAM, PETSc, QuantumESPRESSO, WRF, WPS, ...
- well documented: `http://easybuild.readthedocs.org`

Similar projects

- **Spack** (LLNL), `http://scalability-llnl.github.io/spack/`
- **iVEC Build System (iBS)**, `http://ivec.org/` *(not public (yet))*
- **Smithy** (NICS, ORNL), `http://anthonydigirolamo.github.io/smithy/`

Major differences with EasyBuild:

- smaller community
- fewer supported software packages
- less flexibility

Most (all?) are interested in switching to/merging with EasyBuild.

EasyBuild: requirements

- main target platform (for now) is Linux x86_64 (HPC systems)
  - Red Hat Linux based (Scientific Linux, CentOS, RHEL, ...)
  - also used on Debian, Ubuntu, OpenSUSE, SLES, ...
  - (kind of) works on OS X
  - *no* Windows support (and none planned)

- Python v2.6.x or more recent v2.x (no Python 3 support yet)

- a modules tool:
  - latest release of Tcl(/C) environment modules;
  - or a recent version of *Lmod, a modern alternative*

- (a system C/C++ compiler, to get started)

EasyBuild: feature highlights

- fully autonomously building and installing (scientific) software
  - automatic dependency resolution
  - automatic generation of module files
- thorough logging of executed build/install procedure
- archiving of build specifications
- highly configurable, via config files/environment/command line
- dynamically extendable with additional easyblocks, toolchains, etc.
- support for custom module naming schemes (including hierarchical)
- comprehensive testing: lots of unit tests, regression testing, ...

'Quick' demo for the impatient

```
eb HPL-2.0-goolf-1.4.10-no-OFED.eb --robot
```

- **downloads** all required sources (best effort)
- **builds/installs** *goolf* toolchain (be patient) + HPL with it
  goolf: GCC, OpenMPI, LAPACK, OpenBLAS, FFTW, ScaLAPACK
- **generates module file** for each software package
- default: source/build/install dir in `$HOME/.local/easybuild`
  can be easily changed by configuring EasyBuild differently

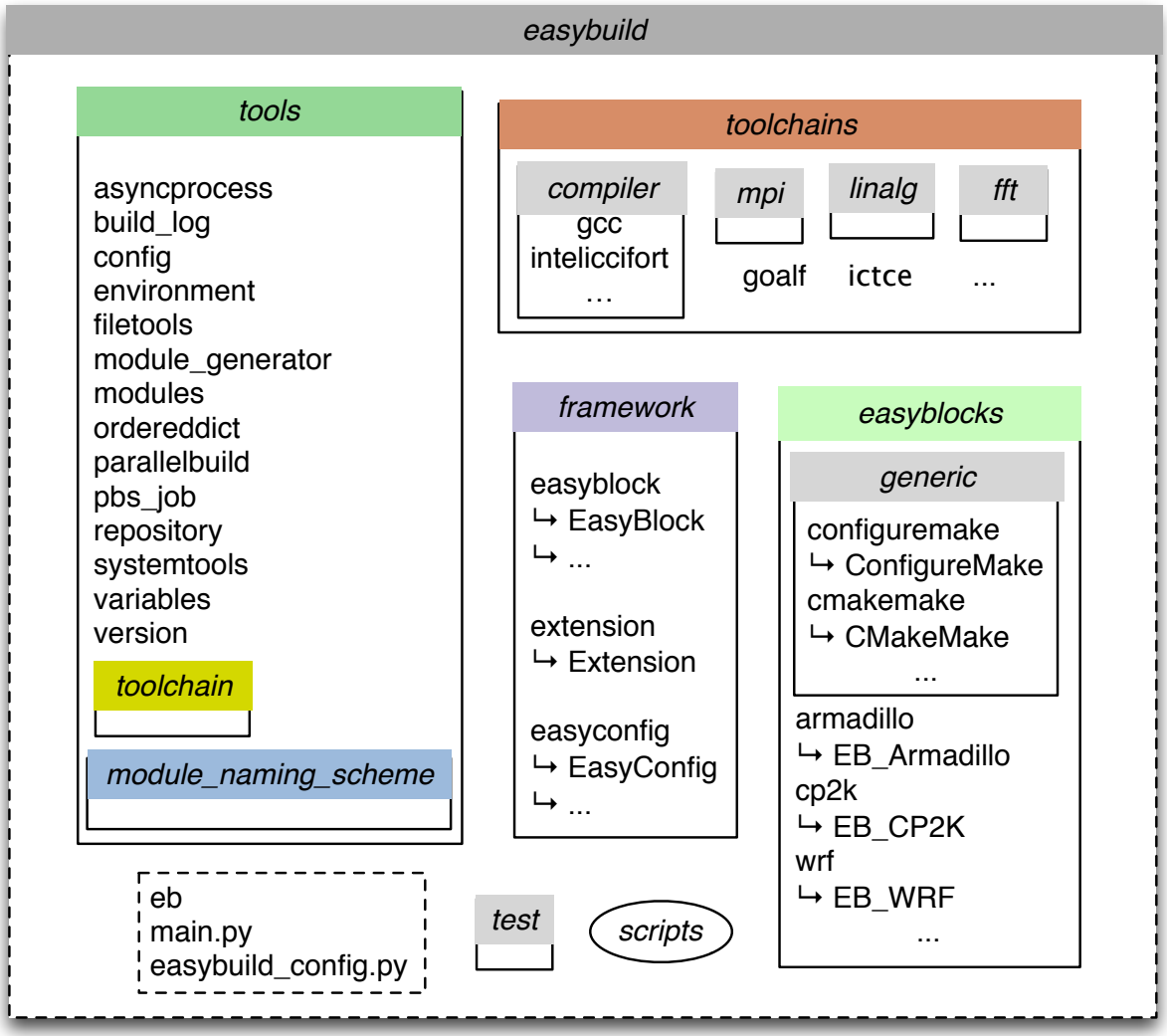
EasyBuild: high-level design overview

- EasyBuild *framework*
  - core of EasyBuild
  - provides supporting functionality for building and installing software
- *easyblock*
  - a Python module
  - implements a (generic) software build/install procedure
- *easyconfig* file
  - build specification: software name/version, compiler toolchain, etc.
- compiler *toolchain*
  - compilers with accompanying libraries (MPI, BLAS/LAPACK, etc.)

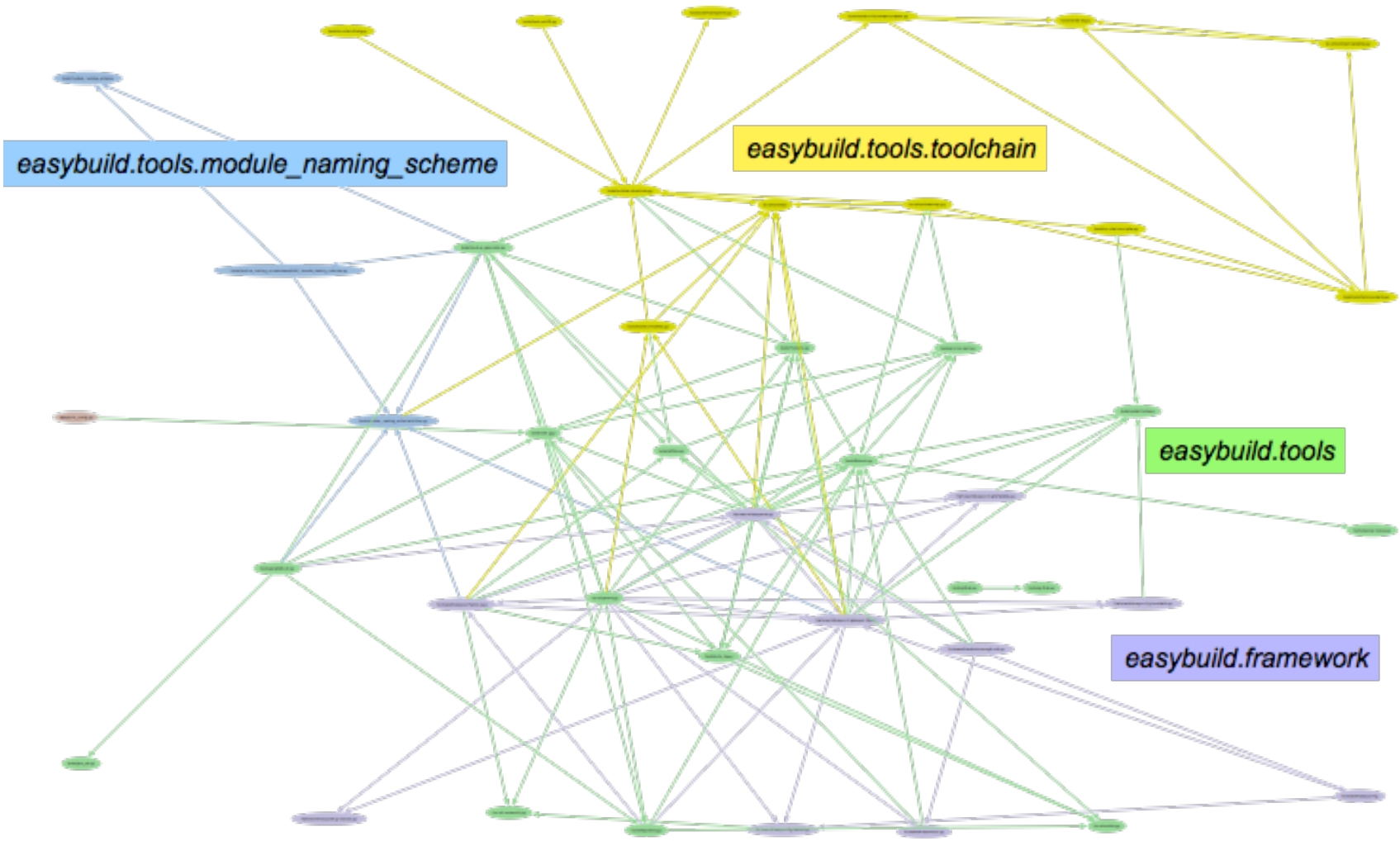
Putting it all together

The EasyBuild *framework* leverages *easyblocks* to automatically build and install (scientific) software using a particular *compiler toolchain*, as specified by one or multiple *easyconfig files*.

EasyBuild: high-level design overview

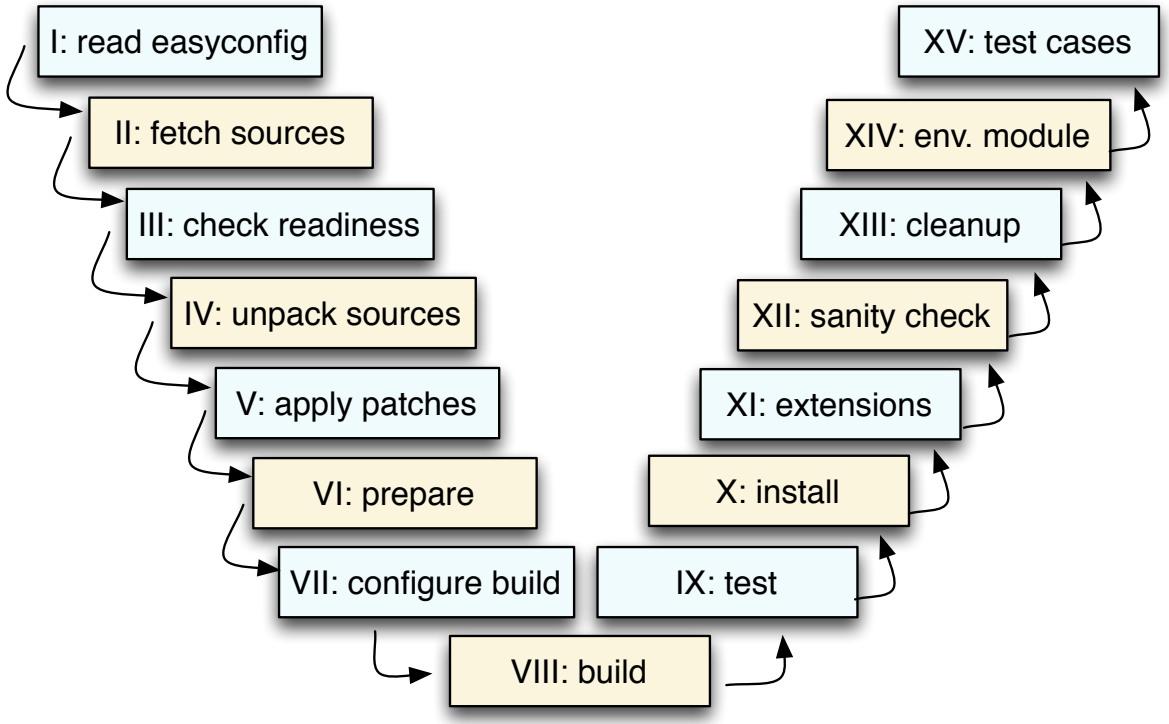


EasyBuild: high-level design overview



Step-wise install procedure

build and install procedure as implemented by EasyBuild



most of these steps can be customised if required,
via *easyconfig parameters* or a *custom easyblock*

EasyBuild: statistics

EasyBuild v1.16.1:

- lines of Python code:

  framework: $\sim 31,500$ (14 packages, 132 modules)
  easyblocks: $\sim 15,000$

- 161 easyblocks in total

  141 software-specific easyblocks
  20 generic easyblocks

- 597 different software packages supported (incl. toolchains)

  bio: 113, tools: 73, lib: 62, devel: 55, vis: 51, data: 40,
  toolchain: 39, math: 39, numlib: 22, mpi: 21, chem: 24,
  lang: 24, system: 17

- 3,066 easyconfig files: different versions/variants, toolchains, ...

List of supported software packages

a2ps ABAQUS ABINIT ABySS ACML **ALADIN** Allinea ALLPATHS-LG AMOS AnalyzeFMRI ANSYS ant ANTs APBS ARB argtable aria2 Armadillo arpack-ng ASE ATLAS Autoconf Automake bam2fastq BamTools Bash BayesTraits bbcp bbFTP bbftpPRO bc beagle-lib Beast BEDOPS BEDTools BFAST binutils BioPerl Biopython BiSearch Bison BitSeq BLACS BLAST BLAT BOINC Bonnie++ Boost Bowtie Bowtie2 BWA byacc bzip2 cairo CAP3 CBLAS ccache CCfits CD-HIT CDO CEM CFITSIO cflow CGAL cgdb Chapel CHARMM Chimera Circos Clang CLHEP CLooG Clustal-Omega ClustalW2 CMake Coreutils Corkscrew **CP2K** CPLEX CRF++ ctffind Cube CUDA Cufflinks cURL cutadapt CVS CVXOPT Cython DB DBD-mysql DBD-SQLite DB_File DIALIGN-TX Diffutils DL_POLY_Classic Docutils **DOLFIN** Doxygen **EasyBuild** ECore ed Eigen ELinks ELPA ELPH Emacs EMBOSS EPD ErlangOTP ESMF ESPResSo evmix expat eXpress FASTA fastahack FastTree FASTX-Toolkit FCM FDS FDTD_Solutions Ferret FFC ffmpeg FFTW FIAT file findutils fixesproto flex FLTK FLUENT fmri FoldX fontconfig FRC_align freeglut FreeSurfer freetype FSL g2clib g2lib GAMESS-US GATE GATK gawk GCC GD GDAL GDB Geant4 GEM-library GEMSTAT GenomeAnalysisTK GEOS gettext GHC Ghostscript GIMPS git GLib GLIMMER GLPK glproto GMAP-GSNAP GMP GMT gnuplot gnutls Go GObject-Introspection google-sparsehash GPAW gperf gperftools grace Graphviz GraphViz2 Greenlet grep grib_api GROMACS GSL gsl GSSAPI GTI GTS guile gzip h4toh5 h5py h5utils HarfBuzz Harminv HDF HDF5 HH-suite HMMER horton HPCG HPL HTSeq HTSlib hwloc Hypre icc ifort imake imkl impi Infernal inputproto Inspector Instant intltool iompi IOR Iperf ipp IPython IsoInfer ispc itac JAGS Jansson JasPer Java Jellyfish Jinja2 JUnit kbproto Kerberos_V5 LAPACK less lftp libcap-ng libcircle libctl libdrm libevent libffi libgd libgtextutils libharu libibmad libibumad libibverbs libICE libidn Libint libint2 libjpeg-turbo libmatheval libpciaccess libpng libpthread-stubs libreadline libSM libsmm LIBSVM LibTIFF libtool libudev libungif libunistring libunwind libX11 libXau libXaw libxc libxcb libXdmcp libXext libXfixes libXft libXi libXinerama libxml2 libXmu libXp libXpm libXrender libxslt libXt libyaml likwid Lmod Lua LWM2 lxml lynx LZO M4 MAFFT make makedepend Maple MariaDB Mathematica MATLAB matplotlib Maven mc MCL mcpp MDP mdtest Meep MEME Mercurial Mesa Mesquite MetaVelvet MethPipe METIS MMSEQ Modeller Molden Molekel molmod Mothur motif MPFR mpi4py mpiBLAST MPICH MPICH2 MrBayes MTL4 MUMmer MUMPS MUSCLE MUST MUSTANG MVAPICH2 MySQL NAMD nano NASM NCBI-Toolkit ncdf4 **NCL** ncurses ncview NEdit netaddr netCDF netCDF-C++ netCDF-C++4 netCDF-Fortran netcdf4-python netifaces NetLibIDN netloc nettle NEURON nodejs ns numactl numexpr numpy NWChem O2scl Oases OCaml Oger OPARI2 OpenBabel OpenBLAS OpenCV **OpenFOAM OpenFOAM-Extend** OpenIFS OpenMD OpenMPI OpenPGM OpenSees OpenSSL ORCA orthomcl otcl OTF OTF2 packmol PAML pandas PANDAseq Pango PAPI parallel Paraview ParFlow ParMETIS ParMGridGen Pasha patch paycheck PCC PCRE PDT Perl **PETSc** petsc4py phonopy PhyML picard pixman pkg-config PLINK PnMPI popt PP PRANK Primer3 printproto problog protobuf pscom PSI psmpi psmpi2 PyQuante pysqlite pyTables **Python** python-dateutil python-meep PyYAML PyZMQ Qhull QLogicMPI Qt qtop QuadProg++ **QuantumESPRESSO** R RAxML RCS RDP-Classifier RELION renderproto rjags RNAz ROOT Rosetta rSeq RSEQtools Ruby runjags Sablotron SAMtools ScaLAPACK Scalasca ScientificPython scikit-learn scipy SCons SCOOP Score-P SCOTCH SDCC SDPA sed segemehl setuptools Shapely SHRiMP SIBELia sickle Silo slalib-c SLEPc SOAPaligner SOAPdenovo SOAPdenovo2 SOAPec SPAdes Sphinx SPRNG SQLite SRA-Tookit Stacks stemming Stow Stride SuiteSparse SURF SWIG sympy systemd Szip TAMkin Tar tbb TCC Tcl tclcl tcsh Tesla-Deployment-Kit texinfo Theano TiCCutils TiMBL TinySVM Tk TopHat Tornado TotalView TREE-PUZZLE Trilinos Trinity UDUNITS UFC UFL util-linux Valgrind VCFtools Velvet ViennaRNA Vim Viper vsc-base vsc-mympirun vsc-mympirun-scoop vsc-processcontrol VSC-tools VTK VTune WHAM **WIEN2k** wiki2beamer **WPS WRF** xbitmaps xcb-proto XCrySDen xextproto xineramaproto XML XML-Dumper XML-LibXML XML-Parser XML-Simple XML-Twig xorg-macros xproto xtrans XZ yaff YamCha YAML-Syck Yasm YAXT ZeroMQ zlib zsh zsync

Installing EasyBuild

http://easybuild.readthedocs.org/en/latest/Installation.html

Install EasyBuild using the bootstrap script *(highly recommended)*:

```
$ curl -O https://raw.github.com/hpcugent/easybuild-framework/develop/easybuild/
scripts/bootstrap_eb.py

$ python bootstrap_eb.py /tmp/$USER # specify your install prefix

$ export MODULEPATH=/tmp/$USER/modules/all:$MODULEPATH

$ module load EasyBuild
```

Update EasyBuild with . . . EasyBuild!

```
$ module load EasyBuild/1.15.2

$ eb EasyBuild-1.15.1.eb --try-software-version=1.16.1

$ module swap EasyBuild EasyBuild/1.16.1
```

Configuring EasyBuild

`http://easybuild.readthedocs.org/en/latest/Configuration.html`

By default, EasyBuild will (ab)use `$HOME/.local/easybuild`.

You should configure EasyBuild to your preferences, via:

- *configuration file(s):* key-value lines, text files (e.g., `prefix=/tmp`)
- *environment variables* (e.g., `$EASYBUILD_PREFIX` set to `/tmp`)
- *command line parameters* (e.g., `--prefix=/tmp`)

Consistency across these options is guaranteed (see `eb --help | tail`).

Priority among different options: cmdline, env vars, config file.

For example:

- `--prefix` overrules `$EASYBUILD_PREFIX`
- `$EASYBUILD_PREFIX` overrules `prefix` in configuration file

First steps with `eb`

- installing bzip2 v1.0.6 with system compiler:

```
$ eb bzip2-1.0.6.eb
$ module load bzip2/1.0.6
```

- or (equivalent), install latest version (that EasyBuild knows about):

```
$ eb --software-name=bzip2 --toolchain-name=dummy
```

- install goolf compiler toolchain (be patient):

```
$ eb goolf-1.4.10-no-OFED.eb --robot # no-OFED: no IB support
```

- install gzip v1.6 on top of goolf toolchain, debug log to stdout:

```
$ eb gzip-1.6-goolf-1.5.14-no-OFED.eb -ldr
```

EasyBuild command line cheat sheet

- getting help, overview of options: `eb --help`
- list of available easyconfig parameters: `eb -a`
- robot build, debug log to stdout: `eb bzip2-1.0.6.eb -ldr`
- overview of required/available modules: `eb --dry-run` or `eb -D`
- list of known toolchains and their definition: `eb --list-toolchains`
- generating easyconfigs:
  `eb bzip2-1.0.6.eb --try-toolchain=ictce,6.2.5 -r`
- searching for easyconfigs: `eb -S` or `eb --search`
- use easyconfigs available in an pull request: `eb --from-pr <PR#>`
- test pull request and upload test report:
  `eb --from-pr <PR#> --upload-test-report --github-user=boegel`

Writing easyconfig files, contributing back

For software packages that follow some type of standard build procedure, just writing an easyconfig file is likely sufficient.

`http://easybuild.readthedocs.org/en/latest/Writing_easyconfig_files.html`

If you have working easyconfigs for new software packages, version updates, or using a different toolchain, consider *contributing back* to the central easybuild-easyconfigs repository.

`https://github.com/hpcugent/easybuild/wiki/Contributing-back`

Over 25% of the builds currently supported were defined by *external contributors* (non-HPC-UGent).

Lmod: a modern modules tool

https://tacc.utexas.edu/research-development/tacc-projects/lmod

- drop-in alternative for Tcl-based module tools (a few edge cases)
- improves user experience, without hindering experts
- written in Lua, consumes Lua (and Tcl) module files
- available since Oct'08, *actively developed*, frequent stable releases
- driven by community demands and feedback
- developed by Dr. Robert McLay (TACC, UT Austin)

Example: swapping modules using `ml` shorthand in a module hierarchy

```
$ ml
Currently loaded modules:
 1) GCC/4.8.2   2) MPICH/3.1.1   3) FFTW/3.3.2
$ ml -GCC Clang
The following have been reloaded:
 1) FFTW/3.3.2   2) MPICH/3.1.1
$ ml
Currently loaded modules:
 1) Clang/3.4   2) MPICH/3.1.1   3) FFTW/3.3.2
```

Lmod: feature highlights

- module hierarchy-aware design and functionality
  - searching across entire module tree with 'module `spider`'
  - automatic reloading of dependent modules on '`module swap`'
  - marking missing dependent modules as inactive after '`module swap`'
- caching of module files, for responsive subcommands (e.g., `avail`)
- site-customizable behavior via provided hooks
- `ml` command ('`ml`' is '`module list`', '`ml GCC`' is '`module load GCC`', ...)
- load/unload shortcuts via + and -
- various other useful/advanced features, including:
  - case-insensitive '`avail`' subcommand
  - can send subcommand output to stdout (rather than to stderr)
  - defining module families (e.g., '`compiler`', '`mpi`')
  - assigning properties to modules (e.g., '`Phi-aware`')
  - stack-based definition of environment variables (using `pushenv`)
  - user-definable collections of modules (`module save`)
  - and a lot more ...

Combined power of EasyBuild and Lmod

Build and install WRF in a module hierarchy, with two different toolchains

```
$ export EASYBUILD_MODULE_NAMING_SCHEME=HierarchicalMNS
$ eb WRF-3.5-goolf-1.4.10.eb --robot # using GCC
$ eb WRF-3.5-ictce-5.3.0.eb --robot # using Intel compilers (icc/ifort)
```

List existing WRF modules, load GCC build

```
$ module spider WRF
...
$ ml GCC OpenMPI WRF
```

Swap to Intel build of WRF

```
$ ml -GCC -OpenMPI +icc +ifort +impi
The following have been reloaded:
...
```

Synergy between EasyBuild and Lmod

- EasyBuild can easily build and install hundreds of packages
  - ⇒ lots of modules, overwhelming for users
- Lmod's support for hierarchical modules trees can help
  - ⇒ support for using Lmod and hierarchical module naming schemes was added to EasyBuild
- EasyBuild uncovered performance issues in Lmod
  - ⇒ Lmod has significantly improved the speed of certain operations (e.g., `module --terse avail`)
- feature requests from the EasyBuild community
  - ⇒ Lmod has added new functionality, for example stack-based definition of environment variables using '`pushenv`'

The synergy between both tools has made them significantly better!

EasyBuild and Lmod communities

- both EasyBuild and Lmod have vibrant communities
- estimating their sizes is difficult, though
  - EasyBuild
    - close to 100 subscribers to the mailing list
    - 15-20 active members on the `#easybuild` IRC channel
    - users/contributors at HPC sites and companies around the world (e.g., JSC, Stanford, U. Auckland, Bayer AG, ...)
    - six 3-day 'hackathon' workshops, at various European HPC sites
    - **next hackathon at University of Basel (Switzerland), Feb 9-11 2015**
  - Lmod
    - $\sim$50 subscribers to mailing list
    - deployed by a couple of hundred HPC sites (e.g., Stanford, Harvard, TACC, U. Warwick, JSC, Total, NASA, ...)
    - number of users $\mathcal{O}$(10,000)
- many sites/users contribute by
  - requests, suggestions, and bug reports
  - sharing patches and implementing new features

HUST-14 paper

Modern Scientific Software Management Using EasyBuild and Lmod

Markus Geimer (JSC)

Kenneth Hoste (HPC-UGent)

Robert McLay (TACC)

`http://hpcugent.github.io/easybuild/files/hust14_paper.pdf`

- paper at HUST-14 workshop (during SC14)
- explains basics of module tools, EasyBuild and Lmod
- highlights issues with current approaches in software installation
- advocates use of a hierarchical module naming scheme
- presents EasyBuild and Lmod as adequate tools for software management

EasyBuild: future work

- command line support for contributing easyconfigs: `eb --new-pr`
- support for Lmod-specific features in EasyBuild
  - module files in Lua syntax
  - module properties
  - module families
  - non-strict version loads: `load(atleast("GCC","4.8"))`
- make the dependency resolution mechanism aware of 'subtoolchains'
- extend EasyBuild to support multiple module naming schemes concurrently (e.g., to gradually move from one layout to another)
- support for creating packages for software installed using EasyBuild
- support for rpath-style dynamic linking of libraries

Do you want to know more?

- EasyBuild website: `http://hpcugent.github.com/easybuild`
- EasyBuild documentation: `http://easybuild.readthedocs.org`
- Lmod website: `http://www.tacc.utexas.edu/tacc-projects/lmod`
- stable EasyBuild releases: `http://pypi.python.org/pypi/easybuild`

  EasyBuild framework: `http://pypi.python.org/pypi/easybuild-framework`

  easyblocks: `http://pypi.python.org/pypi/easybuild-easyblocks`

  easyconfigs `http://pypi.python.org/pypi/easybuild-easyconfigs`
- stable Lmod releases: `http://sourceforge.net/projects/lmod/files`
- source repositories on GitHub

  EasyBuild meta package + docs: `https://github.com/hpcugent/easybuild`

  EasyBuild framework: `https://github.com/hpcugent/easybuild-framework`

  easyblocks: `https://github.com/hpcugent/easybuild-easyblocks`

  easyconfigs: `https://github.com/hpcugent/easybuild-easyconfigs`

  Lmod: `https://github.com/TACC/Lmod`
- EasyBuild mailing list: easybuild@lists.ugent.be

  `https://lists.ugent.be/wws/subscribe/easybuild`
- Twitter: @easy_build, IRC: `#easybuild` on `chat.freenode.net`