EasyBuild: Building Software With Ease

FOSDEM '14

HPC and computational science devroom
Feb 1th 2014

Jens.timmerman@ugent.be
easybuild@lists.ugent.be
HPC-UGent @ Ghent University, Belgium

- central contact for High Performance Computing at university
- established in 2008, part of central IT department (DICT)
- member of Flemish Supercomputer Centre (VSC)
  - collaboration between Flemish university associations
our computing infrastructure:

- seven Tier 2 systems (capacity computing)
- one Tier 1 system
  #119 in Top500 (June’12), currently at #306

HPC-UGent team currently consists of 8 FTEs

- system administration of HPC infrastructure
  - top-down for Tier2 systems: hardware, configuration, user support
  - Tier1: owned by UGent, setup together with HP, user support
- user support and training
  - EasyBuild grew out of need from this
- convincing groups to switch to central infrastructure
Building scientific software is... fun!

Scientists focus on the *functionality* of their software, not on portability, build system, ...

Common *issues* with build procedures of scientific software:

- **incomplete**, e.g. no install step
- requiring human *interaction*
- heavily customized and *non-standard*
- uses *hard-coded* settings
- poor and/or outdated *documentation*

*Very time-consuming* for user support teams!
building from source is preferred in an HPC environment

performance is critical, instruction selection is key (e.g. AVX)

eexisting build tools are

- hard to maintain (e.g., bash scripts)
- stand-alone, no reuse of previous efforts
- OS-dependent (HomeBrew, *Ports, ...)
- custom to (groups of) software packages
e.g., Dorsal (DOLFIN), gmkpack (ALADIN)
not a lot of packaged scientific software available (RPMs, ...)

requires **huge effort**, which is duplicated across distros

Hard to install multiple versions of a program

version

Compiler (intel / gcc / clang)

Mpi stack (openmpi, intel mpi, mpich)

Math kernel (Atlas, Openblas, Gotoblas, IMKL)
Our build tool wish list

- **flexible** framework
- allows for **reproducible** builds
- supports **co-existence** of versions/builds
- enables **sharing** of build procedure implementations
- **fully** **automates** builds
- **dependency** resolution
Building software with ease

EasyBuild

a software build and installation framework

- written in **Python**
- developed in-house for 2.5 years before public release
- **open-source (GPLv2)** since April 2012
- EasyBuild v1.0: **stable API** (November 2012)
- **monthly releases** (latest: v1.10, Dec 24th 2013)
- continuously enhanced and extended

[http://hpcugent.github.io/easybuild](http://hpcugent.github.io/easybuild)
Building software with ease

easybuild

Various contributors

- University of Auckland
- Gregor Mendel Institute of Molecular Plant Biology (GMI), Austria
- University of Luxembourg
- The Cyprus Institute
- Jülich Supercomputing Centre
- Nvidia
- High Performance Computing Center at NTUU "KPI", Kiev
‘Quick’ demo for the impatient

```
$ eb HPL-2.0-goolf-1.4.10.eb -r
```

- downloads all required sources (best effort)
- builds `goolf` toolchain (be patient), and builds HPL with it
  - goolf: GCC, OpenMPI, OpenBlas, ScaLAPACK, FFTW
- Generates a module file
- default: source/build/install dir in `~/.local/easybuild`
Step-wise install procedure

build and install procedure as implemented by EasyBuild

most of these steps can be customized if required
easybuild

Features

- **logging** and archiving
  - entire build process is logged thoroughly, logs stored in install dir
  - easyconfig file used for build is archived (file/svn/git repo)

- **automatic dependency resolution**
  - build stack of software with a single command, using `--robot`

- running **interactive installers autonomously**
  - by passing a Q&A Python dictionary to the `run_cmd_qa` function

- building software in **parallel**
  - e.g., on a (PBS) cluster, by using `--job`

- comprehensive **testing**: unit tests, regression testing
Supported Packages

- 443 packages build out of the box

- Over 3000 example (tested!) easyconfigs

- Including
  - ALADIN, CP2K, DOLFIN, OpenFOAM, NEURON, WPS, WRF
  - QuantumESPRESSO, MWChem
EasyBuild: high-level design
Terminology

**framework**
- Python packages and modules forming *the core of EasyBuild*
- provides (loads of) supporting functionality
- very modular and dynamic design w.r.t. easyblocks, toolchains, ...

**easyblock**
- a Python module providing *implementation of a build procedure*
- can be generic or software-specific

**easyconfig file (.eb)**
- *build specification*:
  - software name/version, toolchain, build options, ...
- simple text files, Python syntax
High-level design: easyblocks

- build procedure implementations

- modular design, dynamically extensible
  - add your easyblock in the Python search path
  - EasyBuild will pick it up

- object-oriented scheme
  - subclass from existing easyblocks or abstract class *EasyBlock*
High-level design: easyblocks

- **build procedure implementations**

- **easyblocks.generic**: generic easyblocks
  - custom support for groups of applications
  - e.g., ConfigureMake, CMakeMake, ...

- **easyblocks**: application-specific easyblocks
tools package

- supporting functionality, e.g.:
  - run_cmd for shell commands
  - run_cmd_qa for interactive commands
  - extract_file for unpacking
  - apply_patch for patching

- tools.toolchain package for compiler toolchains
- tools.module_naming_scheme for module naming schemes
**toolchains** package

- support for **compiler toolchains**
- relies on *tools.toolchain*
- toolchains are defined in here
- organized in subpackages:
  - *toolchains.compiler*
  - *toolchains.mpi*
  - *toolchains.linalg* (BLAS, LAPACK, ...)
  - *toolchains.fft*

- very modular design for allowing extensibility
- plug in a Python module for compiler/library to extend it
**High-level design: framework**

**module_naming_scheme** package

- Support for **custom module naming schemes**
- Flat vs tree
  - e.g.: always prefix compiler/toolchain
- Define your module naming scheme
  - EasyBuild picks up any scheme following the specifications
  - See “Using a custom module naming scheme” wiki page
- Our naming scheme: EasyBuildModuleNamingScheme
- Available since EasyBuild v1.8.0, with limited capabilities
  - Only name, version, versionsuffix and toolchain available
test package

- unit testing of EasyBuild
  
  python -m test.framework.suite

mainly for EasyBuild developers

- New features must have tests
- New bugfixes must have a failing and working test
Comprehensive testing

- **unit tests** are run automagically by Jenkins
- **regression test** results are pulled in on request
- publicly accessible: [https://jenkins1.ugent.be/view/EasyBuild](https://jenkins1.ugent.be/view/EasyBuild)

![Jenkins dashboard showing EasyBuild builds and test results](image)

- unit tests (framework)
- unit tests (easyblocks)
- unit tests (easyconfigs)
- full regression test
Known problems

- Beter tests
  - Validate installations
  - Benchmarks
  - Require domain specific knowledge
- -rpath vs $LD_LIBRARY_PATH
- Sources being removed from the web
- Others?
EasyBuild dependencies

- **Linux / OS X**
  - used daily on Scientific Linux 5.x/6.x (Red Hat-based)
  - also tested on Fedora, Debian, Ubuntu, CentOS, SLES, ...
  - some known issues on OS X, focus is on Linux
  - no Windows support (and none planned for now)

- **Python v2.4** or more recent version (2.x, no Python 3 support yet)

- **environment modules** (or Lmod)

- system C/C++ compiler to bootstrap a GCC toolchain
EasyBuild suffers from the mess that is Python packaging...

$ easy_install --user easybuild
error: option --user not recognized (only for recent versions of easy_install / setuptools)

"You should be using pip!"

$ pip install --user easybuild
pip: No such file or directory (pip not installed)

"Just use --prefix with easy_install!"

$ easy_install --prefix=$HOME easybuild

$ export PATH=$HOME/bin:$PATH

$ eb --version

ERROR: Failed to locate EasyBuild's main script
($PYTHONPATH is not set correctly)
Bootstrapping EasyBuild

The easiest way to install EasyBuild is by bootstrapping it.

https://github.com/hpcugent/easybuild/wiki/Bootstrapping-EasyBuild

$ wget http://hpcugent.github.com/easybuild/bootstrap_eb.py
$ python bootstrap_eb.py $HOME

This will install EasyBuild using EasyBuild, and produce a module:

$ export MODULEPATH=$HOME/modules/all:$MODULEPATH
$ module load EasyBuild
$ eb --version
This is EasyBuild 1.8.2 (framework: 1.8.2, easyblocks: 1.8.2)

We’re also looking into a packaged release (RPM, .deb, ...).
By default, EasyBuild will install software to

\[ $\text{HOME} /.local/easybuild/software \]

and produce modules files in

\[ $\text{HOME} /.local/easybuild/modules/all \]

You can instruct EasyBuild otherwise by **configuring** it, using:

- a **configuration file**, e.g., $HOME/.easybuild/config.cfg
- **environment variables**, e.g., $EASYBUILD_INSTALLPATH
- **command line**, e.g. \(--\text{installpath} \)

https://github.com/hpcugent/easybuild/wiki/Configuration

(note: documentation needs work)
easybuild

building software with ease

Do you want to know more?

**website:** [http://hpcugent.github.com/easybuild](http://hpcugent.github.com/easybuild)

**GitHub:** [https://github.com/hpcugent/easybuild[-framework|-easyblocks|-easyconfigs]](https://github.com/hpcugent/easybuild[-framework|-easyblocks|-easyconfigs])

**PyPi:** [http://pypi.python.org/pypi/easybuild[-framework|-easyblocks|-easyconfigs]](http://pypi.python.org/pypi/easybuild[-framework|-easyblocks|-easyconfigs])

**mailing list:** easybuild@lists.ugent.be

**Twitter:** @easy_build

**IRC:** #easybuild on freenode.net
EasyBuild: Building Software With Ease
FOSDEM '14
HPC and computational science devroom
Feb 1\textsuperscript{th} 2014

Jens.timmerman@ugent.be
easybuild@lists.ugent.be