

EASYBUILD & EESSI @ SURF

Caspar van Leeuwen
High Performance Machine Learning Consultant
Wed 26 Mar 2025

SURF

The past

- November 2016: EasyBuild 3.0.0 released with RPATH support
- Mid 2017: first (publicly available) installations @ SURF through EasyBuild
- Start 2019: EasyBuild becomes the default installation method. Manual installation only by *rare* exception!
- Initially build everything with foss & intel toolchains
- Since Snellius (2021): foss-only
 - More build issues with intel, too time consuming for little benefit

The present: Snellius

General-purpose HPC system

- Phase 1 (Q3 2021, Q4 2022)
 - 600 AMD Rome 7H12 (zen2) CPU nodes
 - 72 Intel Ice Lake + A100 GPU nodes
- Phase 2 (Q3 2023)
 - 800 AMD Genoa 9654 (zen4) CPU nodes
- Phase 3 (Q2 2024)
 - 88 AMD Genoa 9334 (zen4) + H100 GPU nodes
- Total: 230k CPU cores, 640 GPUs, 38 PFLOPS

Variant symlink setup

Architecture-specific symlink: `/sw/arch`

- Points to an architecture-specific prefix:

```
$ ls -al /sw/arch
lrwxrwxrwx 1 root root 32 Mar  3 12:12 /sw/arch -> /gpfs/admin/hpc/sw/arch/AMD-ZEN2

$ ls -al /gpfs/admin/_hpc/sw/arch
...
drwxrwxr-x 6 jenkins jenkins 8192 Dec 19 15:07 AMD-ZEN2
drwxrwxr-x 6 jenkins jenkins 4096 Oct  3 13:17 AMD-ZEN4
drwxrwxr-x 5 jenkins jenkins 4096 Oct  3 13:09 AMD-ZEN4-H100
drwxrwxr-x 6 jenkins jenkins 4096 Oct  3 13:15 INTEL-AVX512
```

Variant symlink setup

We don't export submission environment in batch jobs, but even if you do, it resolves to the correct executable:

```
int1 $ module load GROMACS/2024.3-foss-2024a
int1 $ which gmx_mpi
/sw/arch/RHEL9/EB_production/2024/software/GROMACS/2024.3-foss-2024a/bin/gmx_mpi
int1 $ realpath $(which gmx_mpi)
/gpfs/admin/_hpc/sw/arch/AMD-ZEN2/RHEL9/EB_production/2024/software/GROMACS/2024.3-foss-2024a/bin/gmx_mpi
int1 $ srun -p genoa realpath $(which gmx_mpi)
/gpfs/admin/_hpc/sw/arch/AMD-ZEN4/RHEL9/EB_production/2024/software/GROMACS/2024.3-foss-2024a/bin/gmx_mpi
```

How we manage 4 optimizations

Semi-automated builds by Jenkins, based on a buildlist, e.g.

Site-specific customizations are done through hooks, rather than customizing EasyConfigs / EasyBlocks

```
### Basic Compilers ###
GCCcore-13.3.0.eb
intel-compilers-2024.2.0.eb --accept-eula-for=Intel-oneAPI --from-pr=20903
Clang-18.1.8-GCCcore-13.3.0.eb --from-pr=21117

##### MPI Libraries ###
OpenMPI-5.0.3-GCC-13.3.0.eb --hooks=/sw/eb/easyconfigs-surf/hooks/mpi_hook.py --include-
easyblocks=/sw/eb/easyblocks-surf/openmpi.py --from-commit=a6c22ba28a69f0c42724a72243f92aa27fc6459c
impi-2021.13.0-intel-compilers-2024.2.0.eb --accept-eula-for=Intel-oneAPI --hooks=/sw/eb/easyconfigs-
surf/hooks/mpi_hook.py --from-pr=20919
MPICH-4.2.2-GCC-13.3.0.eb --from-pr=21442 --hooks=/sw/eb/easyconfigs-surf/hooks/mpi_hook.py
mpi4py-4.0.1-gompi-2024a.eb --from-pr=21662

##### Libraries #####
pybind11-2.12.0-GCC-13.3.0.eb --from-pr=20829
libcint-6.1.2-gfbf-2024a.eb --from-pr=21545
libpng-1.6.43-GCCcore-13.3.0.eb
libxml2-2.12.7-GCCcore-13.3.0.eb --hook=/sw/eb/easyconfigs-surf/hooks/libxml2_hook.py
```



Branch: —

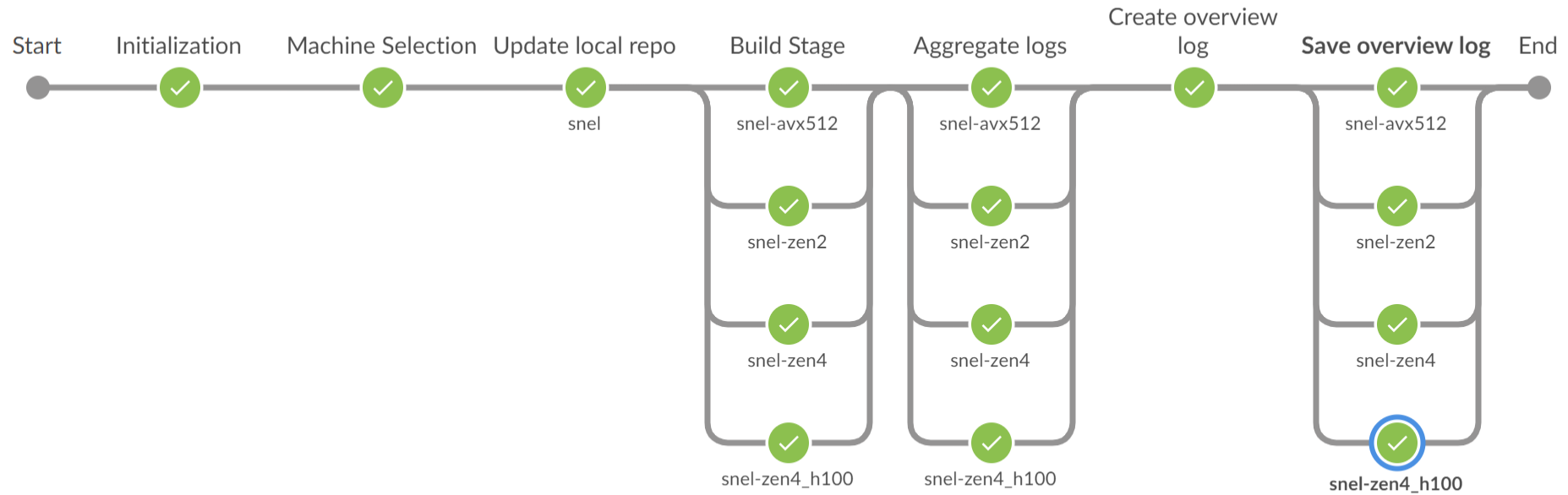
🕒 4m 44s

Changes by Benjamin Czaja

Commit: —

🕒 a day ago

Started by user Benjamin Czaja



Save overview log / snel-zen4_h100 - 19s



- ✓ > overview_logfile — Restore files previously stashed <1s
- ✓ > srun -p gpu_sw_h100 -N 1 --ntasks-per-node=1 -c 64 --gpus-per-node=4 -t 5:00 cp overview_summary.log \${JENKINS_LOGPATH_REAL}/overview_summary.log — Shell Script 16s

Managing OS upgrades

OS upgrades have caused issues in the past, requiring rebuilds

- One solution was to pro-actively just rebuild the entire stack, on the new OS
- OS-specific prefix allows building new stack while keeping old stack
- Update `MODULEPATH` to new prefix after upgrade to 'flip the switch'

```
int1 $ realpath $(which gmx_mpi)
```

```
/gpfs/admin/_hpc/sw/arch/AMD-ZEN2/RHEL9/EB_production/2024/software/GROMACS/2024.3-foss-2024a/bin/gmx_mpi
```

- But: strategy not used anymore on RHEL8 to RHEL9 upgrade
 - ReFrame tests didn't signal many issues
 - There were *some* issues in the end (e.g. OpenSSL related), but not many, and all handled on the fly after the upgrade

User installations

Wrapper *eblocalinstall*:

- `EASYBUILD_INSTALLPATH = ~/.local/easybuild/...`
- `EASYBUILD_OPTARCH = <lowest_common_denominator>`
- `eb ...`

Snellius software policy

We update our software stack once per year. Typically:

- based on 202Xa ← Currently considering to add 202Xb, but toolchains-only
- released around Q3 or Q4
- O(400) packages

Only honor software installation requests if:

- ... is versioned
- ... has an official release within last 2 years
- Max. 2 versions per software per year

Though we may allow exceptions (but don't tell our users 😊)

Snellius software policy

Yearly software stacks available through meta-modules (that set `MODULEPATH`)

- Clear what is still supported
- Confronts users with how old things are that they are using...

```
$ module av
----- /sw/noarch/environment -----
2022  2023  2024  EESSI/2023.06
```



Full support: additional installations + resolve issues with existing installations

Limited support: resolve issues with existing installations

Deprecated: if it still works for you, we won't stop you from using it...

13 Older stacks (2020, 2021) are still there if you know where to look (but don't tell our users 😊)



The future

- EasyBuild: share installation recipes
 - Share some of the software-installation burden with the community
 - But: potential issues, e.g. because of differences in OS stack
- European Environment for Scientific Software Installations (EESSI)¹: share *actual installations*
 - Share more of the software-installation burden with the community

¹<https://www.eessi.io/docs/>

The future

Use EESSI as the base software stack

- Main software environment on Snellius based on EESSI
 - Already the case on SURF Experimental Technologies Platform
- Provide site installations (e.g. proprietary software) on top through the `EESSI-extend` module¹
 - Initially on local filesystem
 - Long term, through CVMFS ('software.surf.nl') with whitelisting for SURF-internal systems
 - Build and deployed by `eessi-bot`²
- Webinar, (probably) June 2nd 2025. See: <https://gitlab.com/eessi/support/-/issues/135>

¹https://www.eessi.io/docs/using_eessi/building_on_eessi/#building-software-on-top-of-eessi-with-easybuild

²<https://www.eessi.io/docs/bot/>

The future

Advantages for EESSI-based software environment

- More synergy / shared effort with other HPC teams
- Less build effort when new system or phase gets deployed
- Uniform software stack across SURF systems (Snellius, SURF Research Cloud & Spider cluster)
- Better build automation (EESSI build bot instead of Jenkins)
- Optimized user-space builds (through `EESSI-extend`) instead of lowest common denominator

The future

Risks & mitigations in adopting EESSI-based software environment

- (How) do we provide support on software that we did not build/deploy ourselves?
 - Active involvement in EESSI to build knowledge & experience
 - EESSI *community* can *also* provide support, and has much more knowledge!
- Risk: EESSI disappears / is no longer maintained
 - ‘software.surf.nl’ build on top of EESSI => easy to (re)build full stack
 - Bigger community = smaller risks

The future

Risks & mitigations in adopting EESSI-based software environment

- Risk: EESSI infrastructure (Stratum 0 / 1) is unavailable
 - Private Stratum 1 hosts full copy of the SW stack (+ better performance!)
- Risk: EESSI does not support the hardware architecture I care about
 - Provide build infrastructure for EESSI

The future

So if it's EESSI, EESSI, EESSI, is there a future for EasyBuild?

- Yes! EasyBuild is a *key* technology in EESSI facilitating both the builds in the repository as well as the user-local builds!
- EESSI may create additional adoption of EasyBuild
 - Community adopts EESSI (to get easy access to software they care about)
 - Community wants newer versions => Motivation to dive into EasyBuild
 - Have seen this in discussions with geosciences & radio-astronomy communities in NL