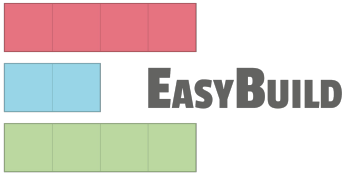


EESSI CI/CD Webinar

11 May 2026 - Alan O'Cais

https://www.eessi.io/docs/training-events/2026/webinar-series-2026Q2

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European High Performance Computing Joint Undertaking (JU) and countries participating in the project. Neither the European Union nor the granting authority can be held responsible for them.

Helpful knowledge

- You have watched the episodes from the webinar series:
  - *Introduction to EESSI*
  - *Building software on top of EESSI*
- You have built a software package in some way (e.g., compiled)
  - Nice to have: heard of (and maybe used) CMake/Autotools
- You are somewhat familiar with Git
  - Nice to have: familiar with GitHub/GitLab
- Familiar with *environment modules (Lmod)*

What if you no longer have to install

a ***broad range of scientific software***

from scratch on every laptop, HPC cluster,

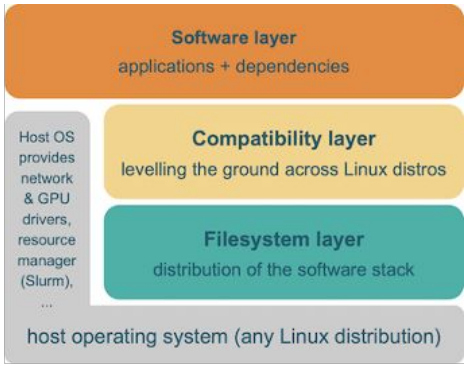
or cloud instance you use or maintain,

without compromising on performance?

Major goals of EESSI

- Providing a truly **uniform software stack**
  - Use the (exact) same software environment everywhere
  - **Without sacrificing performance** for “mobility of compute”
    (like is typically done with containers/conda)
- **Avoid duplicate work** (for researchers, HPC support teams, sysadmins, …)
  - Tools that automate software installation process
    (EasyBuild, Spack) are not sufficient anymore
  - Go beyond sharing build recipes => work towards a shared software stack
- Facilitate HPC training, development of (scientific) software, …

How does EESSI work?



- Software installations included in EESSI are:
  - Automatically **“streamed in” on demand** (via CernVM-FS)
  - Built to be **independent of the host operating system**

    “Containers without the containing”
  - **Optimized** for specific CPU generations + specific GPU types
- Initialization script auto-detects CPU + GPU of the system

The EESSI User Experience

```
$ source /cvmfs/software.eessi.io/versions/2023.06/init/lmod/bash
Module for EESSI/2023.06 loaded successfully
$ module load GROMACS/2024.1-foss-2023b
$ gmx mdrun ...
```

EESSI provides **on-demand streaming**

of (scientific) software (like music, TV-series, …)

EESSI for CI/CD

- Introducing Continuous Integration (CI)
- Navigating EESSI to build my project
- Building my project with the EESSI GitHub Action
- Navigating EasyBuild to build with **`EESSI-extend`**
- Building with **`EESSI-extend`** and the EESSI GitHub Action
- Building my project with the EESSI GitLab Component
- CI with EESSI+Spack
- Continuous Deployment and what EESSI can offer there

What is Continuous Integration (CI)?

- Development practice of frequently merging code into a shared repository
- Automated build and test run on each code change
- Helps detect bugs early and improve code quality
- Provides immediate feedback to developers
- Ensures the application is always in a deployable state

Key Components of CI

- ✅ Version Control System (e.g., Git)
- 🔄 Automated Build System
- 🧪 Automated Testing Suite
- 📣 Notification/Feedback Mechanism

Benefits of Continuous Integration

- 🔍 Early detection of integration issues
- 🚀 Faster and safer release cycles
- 🔧 Encourages small, incremental changes
- 🧠 Improved code quality and team collaboration
- 🔁 Easier refactoring and code maintenance

Specific Challenges of CI in HPC context

- Need HPC-suitable toolchains and dependencies
- Need a way to deal with MPI
- Typically want to test defined architectures
- Want to have accelerator support
- Performance and scalability (also) matter

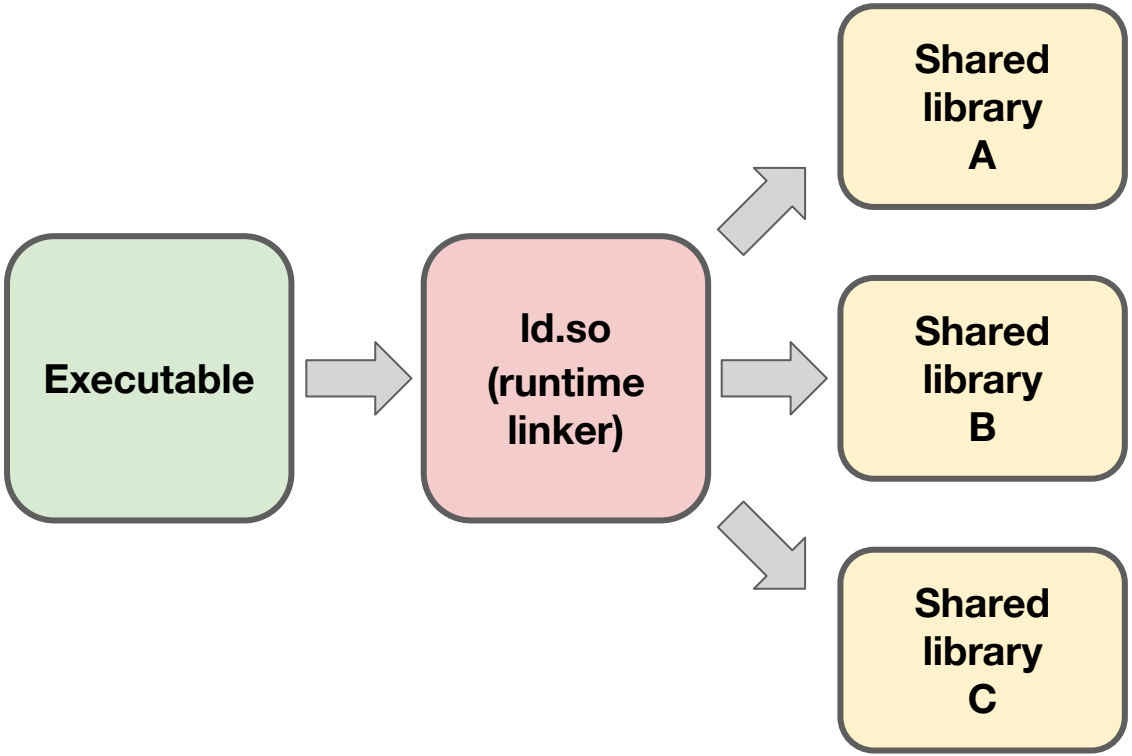
Leveraging EESSI in CI environments

- EESSI can be used in CI environments like:
  - GitHub: github.com/marketplace/actions/eessi
  - GitLab: gitlab.com/explore/catalog/eessi/gitlab-eessi
- EESSI can provide:
  - Different compilers to test your software with
  - Required dependencies for your software
  - Additional tools like ReFrame, performance analysis tools, …
- Other than CernVM-FS to get access to EESSI, no installations required!
  - Everything is pulled in on-demand by CernVM-FS
- Significantly facilitates also running CI tests in other contexts

Building on top of EESSI

- Load the necessary modules that you need for compilation
  - Compiler, MPI, libraries, dependencies,...
  - Best to try to keep dependencies in the same *toolchain*
- Load a **`buildenv`** module for your toolchain
  - Configures typical environment variables for a build (**`CC`**, **`CFLAGS`**, **`LIBS`**, **`...`**)
  - Uses *compiler wrappers* to simplify building with EESSI

Building on top of EESSI



Where does the loader find the libraries?

- Clues in environment variables
- Information in the binary (RPATH)
- It's own defaults

Big long demo:
An example software package

- github.com/EESSI/cicd-demo
- Dependencies: HDF5 and MPI
- Builds with CMake
- Tests via **`ctest`**

Cheat sheet: https://hackmd.io/myPkGyj-Rz6pQNMOZWi1IA?view

EESSI plus Spack can also work in CI

- EESSI integration with Spack is still (very much) a work in progress
- github.com/lorisercole/spood is where things are currently happening
  - Demonstrates that the integration can work
- Proof of principle of EESSI + Spack multi-arch CI on GitHub
  - github.com/ocaisa/spood/pull/1

Cheat sheet: https://hackmd.io/myPkGyj-Rz6pQNMOZWi1IA?view

Continuous Delivery (CD)

- Consider case of “release candidates” and production releases
- Developers commit code → triggers CI
- If tests pass, code can be included in a Release Candidate (RC)
- RC is deployed to a staging environment
- Manual/automated approval takes place
- RC is promoted to production manually or with a controlled deployment

What can EESSI do for CD?

- Production quality software deployed to **`software.eessi.io`**
- Development service **`dev.eessi.io`**, mostly for release candidates
  - More control for developers
  - Access/deployment to specific architectures
- EESSI is working on getting EuroHPC site support for **EESSI test suite**, could be leveraged for **`dev.eessi.io`** to remove need for direct access to resources for development

Repeat demo for yourself

- Good exercise to get the concepts straight
  - Fork github.com/EESSI/cicd-demo
  - Build locally with EESSI first
  - Transfer build to GitHub (or GitLab)
  - Port to EasyBuild (this is an investment for EESSI CD)
  - Transfer port to GitHub (or GitLab)
- Cheat sheet: https://hackmd.io/myPkGyj-Rz6pQNMOZWi1IA?view

Website: https://eessi.io

Join our Slack channel (see join link on website)

Documentation: https://eessi.io/docs

Blog: https://eessi.io/docs/blog

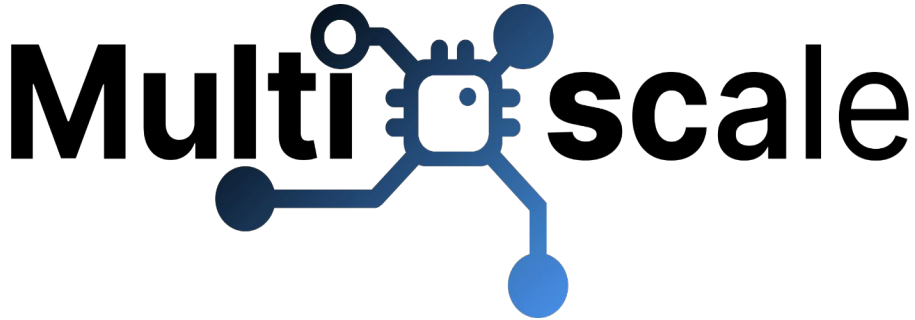
GitHub: https://github.com/eessi

Paper (open access): https://doi.org/10.1002/spe.3075

EESSI YouTube channel

Bi-monthly online meetings
(first Thursday of every other month, 14:00 CEST)

EESSI Happy Hour
(every Monday, 14:00 CEST)



Co-funded by the European Union

EuroHPC Joint Undertaking

Web page: multixscale.eu

Facebook: MultiXscale

Twitter: @MultiXscale

LinkedIn: MultiXscale

BlueSky: MultiXscale

KEMIJSKI INŠTITUT

UNIVERSITAT DE BARCELONA

Universität Stuttgart

UNIVERSITAS BERGENSIS

SORBONNE UNIVERSITÉ

Université de Toulouse

Consiglio Nazionale delle Ricerche

MAX-PLANCK-GESELLSCHAFT

HPC Now!

university of groningen

SURF

JÜLICH Forschungszentrum

GHENT UNIVERSITY

BSC Barcelona Supercomputing Center Centro Nacional de Supercomputación

LEONARDO

iit ISTITUTO ITALIANO DI TECNOLOGIA

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and countries participating in the project under grant agreement No 101093169.