Package management in the hands of users: dream and reality

Ludovic Courtès

FOSDEM 2026

Inria


[bob@supercomp ~]$ module av mpi
-------- /cm/shared/modules/generic/modulefiles ---
mpi/intel/2019.4.243
mpi/openmpi/2.0.4
mpi/openmpi/3.1.4
mpi/openmpi/4.0.1
mpi/openmpi/4.0.1-intel
mpi/openmpi/4.0.2
mpi/openmpi/4.0.2-testing
mpi/openmpi/4.0.3
mpi/openmpi/4.0.3-mlx
mpi/openmpi/4.0.7
mpi/openmpi/4.1.1
mpi/openmpi/4.1.5
mpi/openmpi/5.0.1
[bob@supercomp ~]$ module load openmpi/2.0.4
ATARI
XC12

```
$TOOL install gromacs
```

```
$TOOL install gromacs
```

+ **flexibility**
+ **portability, with performance**
+ **reproducibility**

Reproducible and User-Controlled Software Environments in HPC with Guix

Ludovic Courtès[1] and Ricardo Wurmus[2]

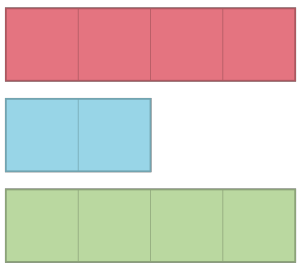
[1] Inria, Bordeaux, France
[2] Max Delbrück Center for Molecular Medicine, Berlin, Germany

Abstract. Support teams of high-performance computing (HPC) systems often find themselves between a rock and a hard place: on one hand, they understandably administrate these large systems in a conservative way, but on the other hand, they try to satisfy their users by deploying up-to-date tool chains as well as libraries and scientific software. HPC system users often have no guarantee that they will be able to reproduce results at a later point in time, even on the same system—software may have been upgraded, removed, or recompiled under their feet, and they have little hope of being able to reproduce the same software environment elsewhere. We present GNU Guix and the functional package management paradigm and show how it can improve reproducibility and sharing among researchers with representative use cases.

https://hal.science/hal-01161771/

Spack

CONDA



EASYBUILD.io

building software with ease

APPTAINER

- **PlaFRIM** (FR): Bordeaux (3,000+ cores)
- **GriCAD** (FR): Grenoble (1,000+ cores)
- **GLICID** (FR): Nantes (4,000+ cores)
- **Grid'5000/SLICES** (FR): 9 sites (12,000+ cores)
- **Max Delbrück Center** (DE): 250-node cluster + workstations
- ...
- Tier-1? Tier-0?

```
guix shell intel-mpi-benchmarks -- \
  srun ... IMB-MPI1 PingPong
```

```
guix shell intel-mpi-benchmarks \
  --with-input=openmpi=mpich \
  -- srun ... IMB-MPI1 PingPong
```

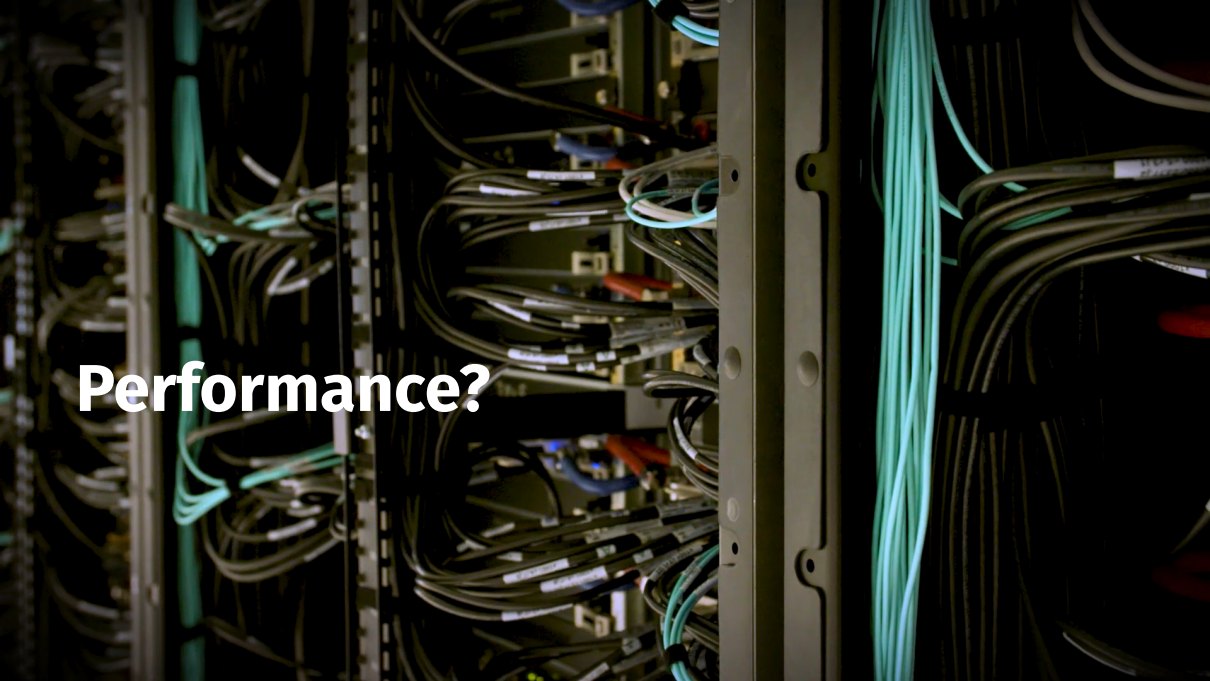
https://guix.gnu.org/manual/devel/en/html_node/Package-Transformation-Options.html

```
guix shell --manifest=manifest.scm -C
```

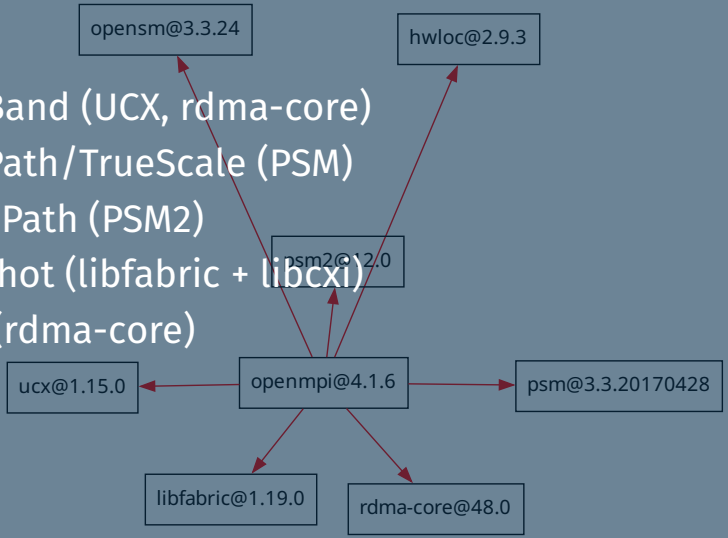
```
(specifications->manifest
  '("gcc-toolchain" "coreutils" "grep" "sed"
    "openmpi" "openblas" "petsc"))
```

Reproducible environments: 2 files, 2 commands

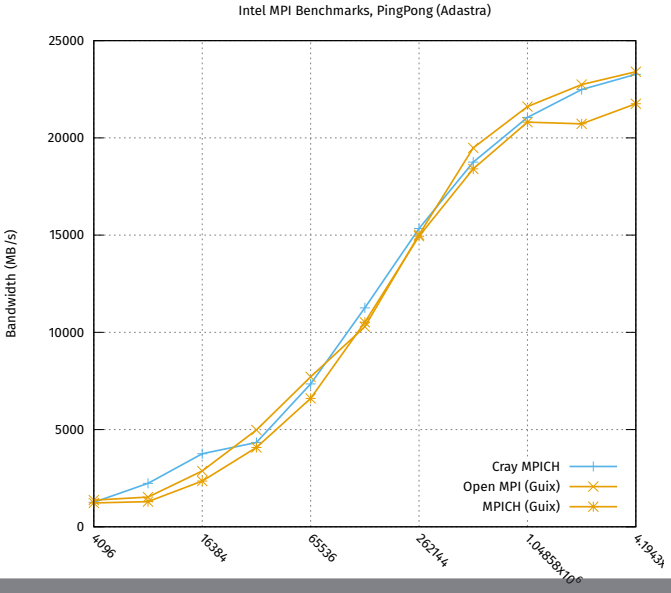
1. `guix describe -f channels > channels.scm`
2. ```
   guix time-machine -C channels.scm -- \
     shell -m manifest.scm
   ```
   ```


Performance?

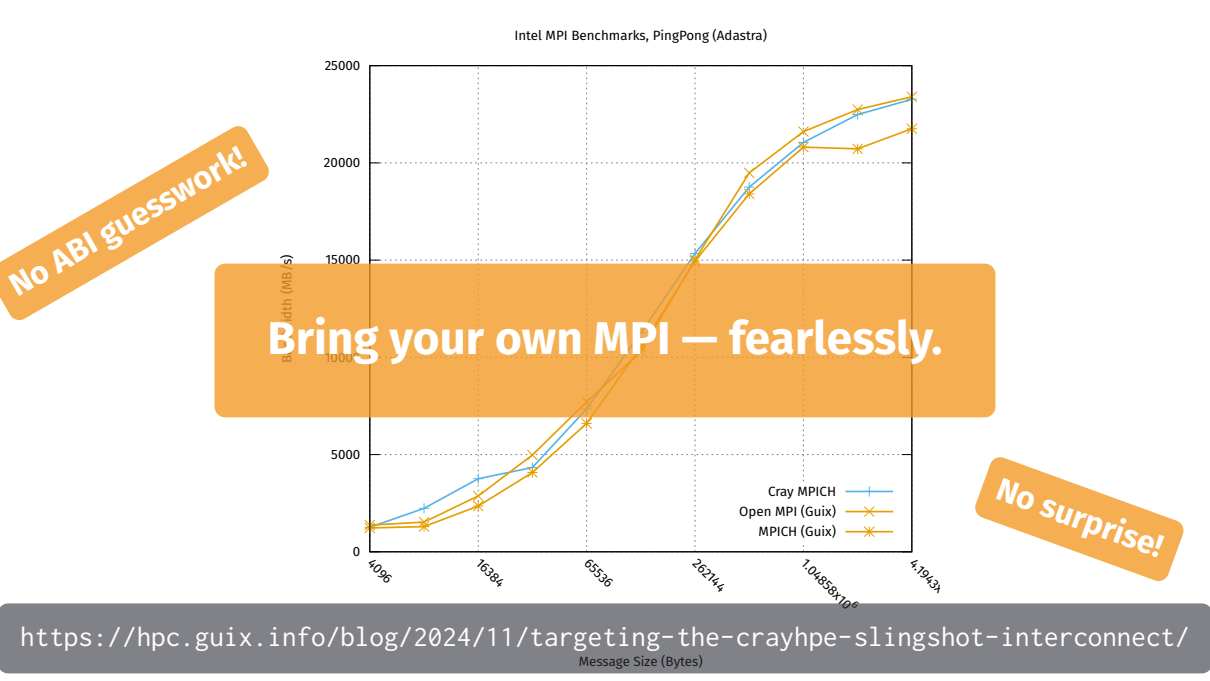
- InfiniBand (UCX, rdma-core)
- InfiniPath/TrueScale (PSM)
- Omni-Path (PSM2)
- Slingshot (libfabric + libcxi)
- RoCE (rdma-core)
- ...



https://hpc.guix.info/blog/2019/12/optimized-and-portable-open-mpi-packaging/



https://hpc.guix.info/blog/2024/11/targeting-the-crayhpe-slingshot-interconnect/


Intel MPI Benchmarks, PingPong (Adastra)
25000
20000
15000
5000
0
4096
16384
65536
262144
1.04858x10^6
4.19430
Message Size (Bytes)
Cray MPICH
Open MPI (Guix)
MPICH (Guix)
No ABI guesswork!
Bring your own MPI — fearlessly.
No surprise!
https://hpc.guix.info/blog/2024/11/targeting-the-crayhpe-slingshot-interconnect/

```
$ guix shell eigen-benchmarks -- \
    benchBlasGemm 240 240 240
240 x 240 x 240
cblas: 0.20367 (16.289 GFlops/s)
eigen : 0.285149 (11.635 GFlops/s)
```

```
$ guix shell eigen-benchmarks -- \
    benchBlasGemm 240 240 240
240 x 240 x 240
cblas: 0.20367 (16.289 GFlops/s)
eigen : 0.285149 (11.635 GFlops/s)

$ guix shell --tune=skylake eigen-benchmarks -- \
    benchBlasGemm 240 240 240
guix shell: tuning for CPU micro-architecture skylake
240 x 240 x 240
cblas: 0.203131 (16.333 GFlops/s)
eigen : 0.0929638 (35.688 GFlops/s)
```

Package multi-versioning

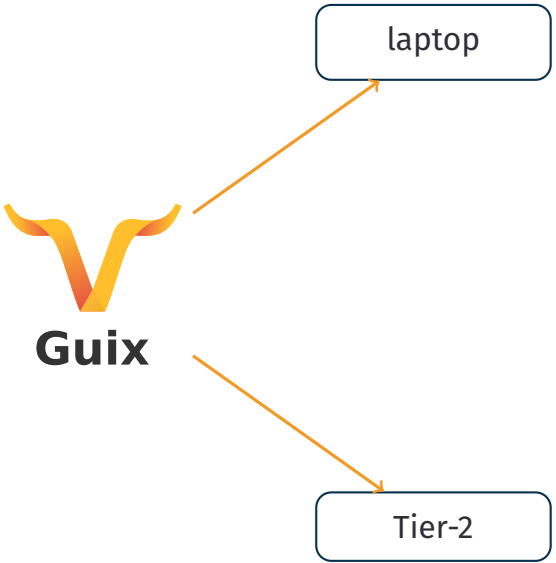
https://hpc.guix.info/blog/2022/01/tuning-packages-for-a-cpu-micro-architecture/

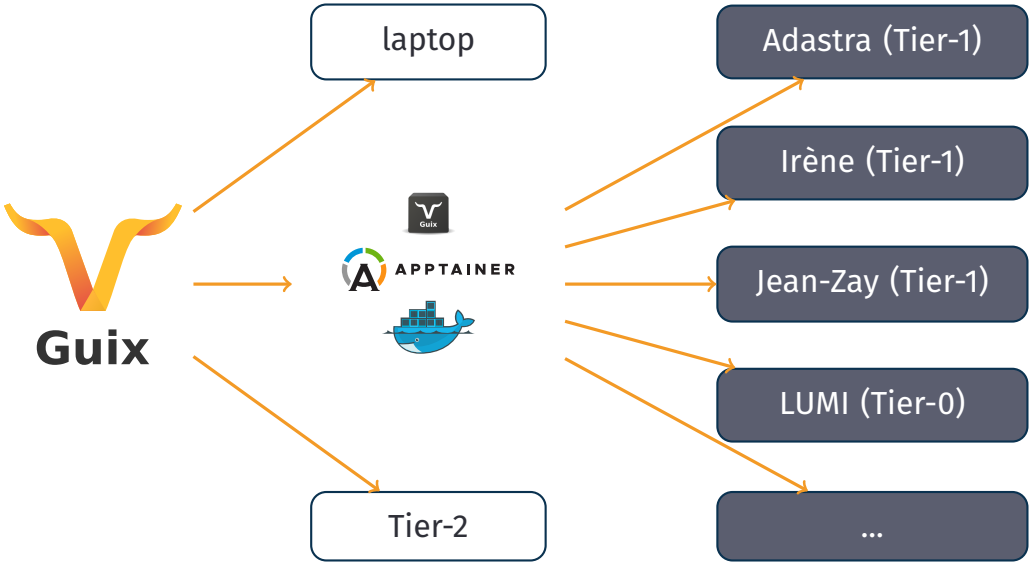

```
guix shell --amd-gpu=gfx90a,gfx942 \
  rocm-bandwidth-test
```

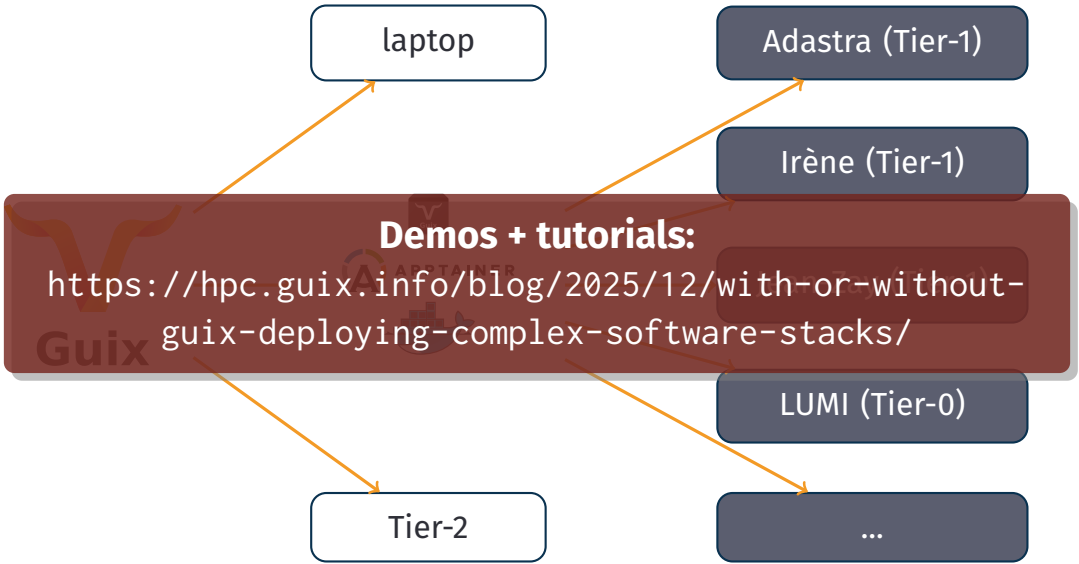
```
laptop$ guix pack -R -S /bin=bin \
      intel-mpi-benchmarks
```

```
laptop$ guix pack -R -S /bin=bin \
      intel-mpi-benchmarks
```

```
supercomp$ tar xf pack.tar.gz
supercomp$ unset LD_LIBRARY_PATH
supercomp$ ./bin/srun -A ... ./bin/IMB-MPI1
```


laptop
Guix
Tier-2

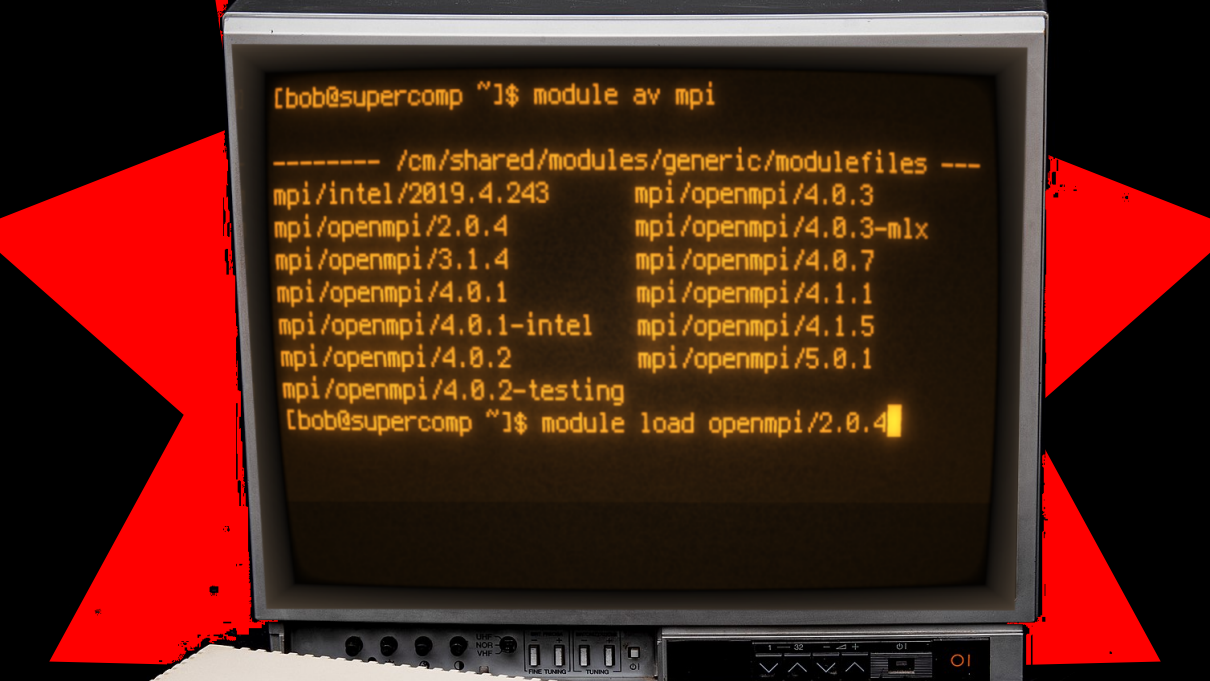

laptop
Guix
Guix
APPTAINER
Tier-2
Adastra (Tier-1)
Irène (Tier-1)
Jean-Zay (Tier-1)
LUMI (Tier-0)
...


laptop
Adastra (Tier-1)
Irène (Tier-1)
Demos + tutorials:
https://hpc.guix.info/blog/2025/12/with-or-without-guix-deploying-complex-software-stacks/
LUMI (Tier-0)
Tier-2
...

- N. Vallet *et al.*, *Toward practical transparent verifiable and long-term reproducible research using Guix*, Nature Scientific Data, Oct. 2022
- M. Felšöci, *Fast Solvers for High-Frequency Aeroacoustics*, PhD thesis, Feb. 2023
- L. Bilke *et al.*, *Reproducible HPC software deployments, simulations, and workflows — a case study for far-field deep geological repository assessment*, Aug. 2025

https://guix.gnu.org/publications

HPC software environments today: reality.


[bob@supercomp ~]$ module av mpi
-------- /cm/shared/modules/generic/modulefiles ---
mpi/intel/2019.4.243
mpi/openmpi/2.0.4
mpi/openmpi/3.1.4
mpi/openmpi/4.0.1
mpi/openmpi/4.0.1-intel
mpi/openmpi/4.0.2
mpi/openmpi/4.0.2-testing
mpi/openmpi/4.0.3
mpi/openmpi/4.0.3-mlx
mpi/openmpi/4.0.7
mpi/openmpi/4.1.1
mpi/openmpi/4.1.5
mpi/openmpi/5.0.1
[bob@supercomp ~]$ module load openmpi/2.0.4

Unprivileged user installs

Spack does not require administrator privileges to install packages. You can install software in any directory you choose, making it easy to manage packages in your home directory or shared project locations without needing sudo access.

```
https://spack.io/about/
```

***Inode quota**: The inode quota is super restrictive for someone doing multiple local installs (Spack or miniConda) [...] which forces us to archive certain installations to test others.*

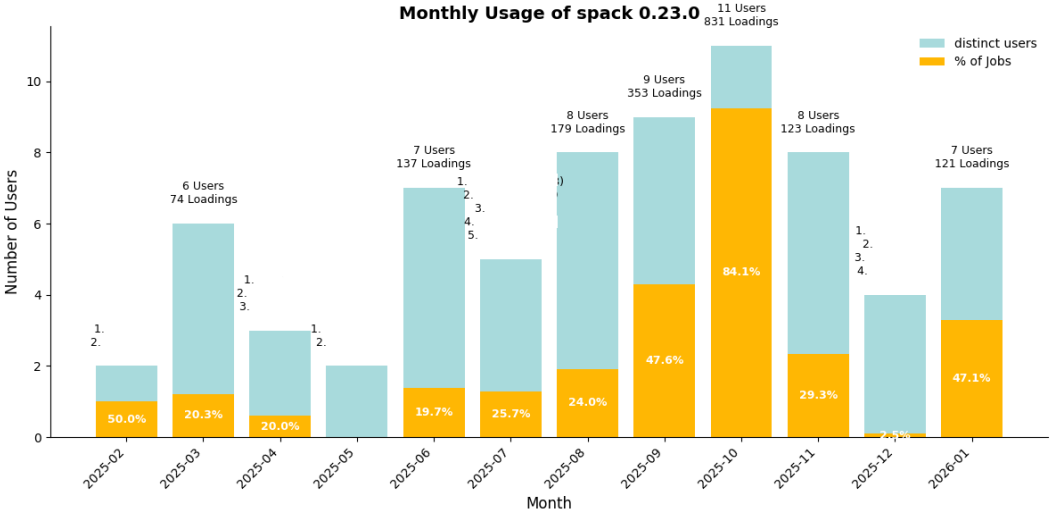
– IDRIS User Committee report, Sept. 2024

LUMI provides a module to load a pre-configured Spack instance: `module load spack`. When you load this module, you will use a Spack instance configured to compile software with the Cray programming environment. The software will be installed in a location determined by you in the environmental variable `$SPACK_USER_PREFIX`. This Spack instance is "chained" to the upstream instance in `/appl/lumi/spack`, which means that you can build new packages on top of the already installed ones by the LUMI User Support Team (similar to how the LUMI EasyBuild setup works).

```
https://docs.lumi-supercomputer.eu/software/installing/spack/
```

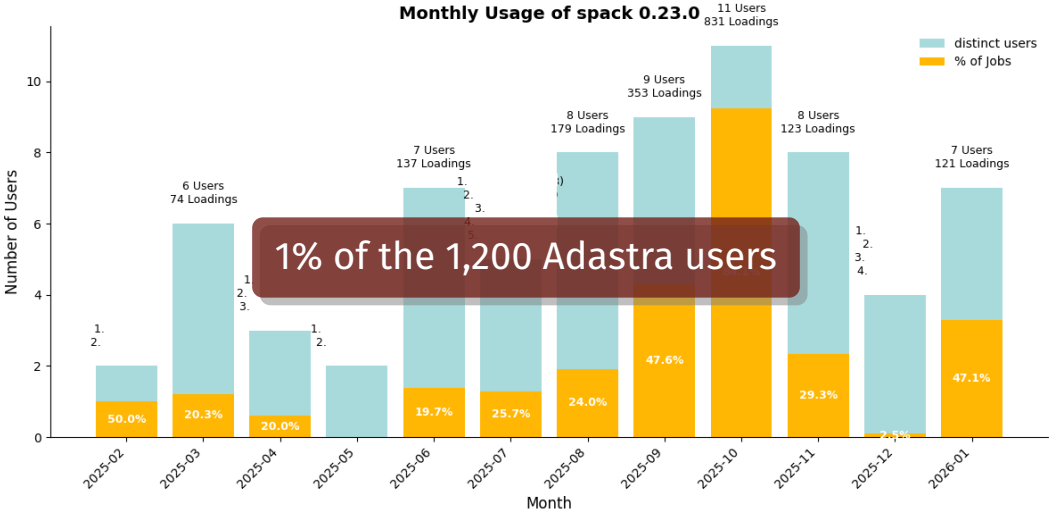
	EasyBuild-user?	Spack-user?
LUMI (FI)	✓	✓
JUWELS (DE)	✓	✗
MareNostrum (ES)	✓	✓
MeLuXina (LU)	✗	✗
Alps (CH)	✗	✗
Adastra (FR)	✗	✓
Irène (FR)	✗	✗
Jean-Zay (FR)	✗	✗

Spack-user usage on Adastra (FR)



Courtesy of the HPC dept. at CINES, thanks!

Spack-user usage on Adastra (FR)



Courtesy of the HPC dept. at CINES, thanks!

So... what's next?

5 Inria centers

CEA

CNRS, univ., ...

LLNL, Stanford, ...

https://numpex.org/exadi-development-and-integration/

CINES

5 Inria centers

TGCC

CEA

CNRS, univ., ...

IDRIS

LLNL, Stanford, ...

Tier-2 clusters

https://numpex.org/exadi-development-and-integration/

By the way, we're hiring! :-)

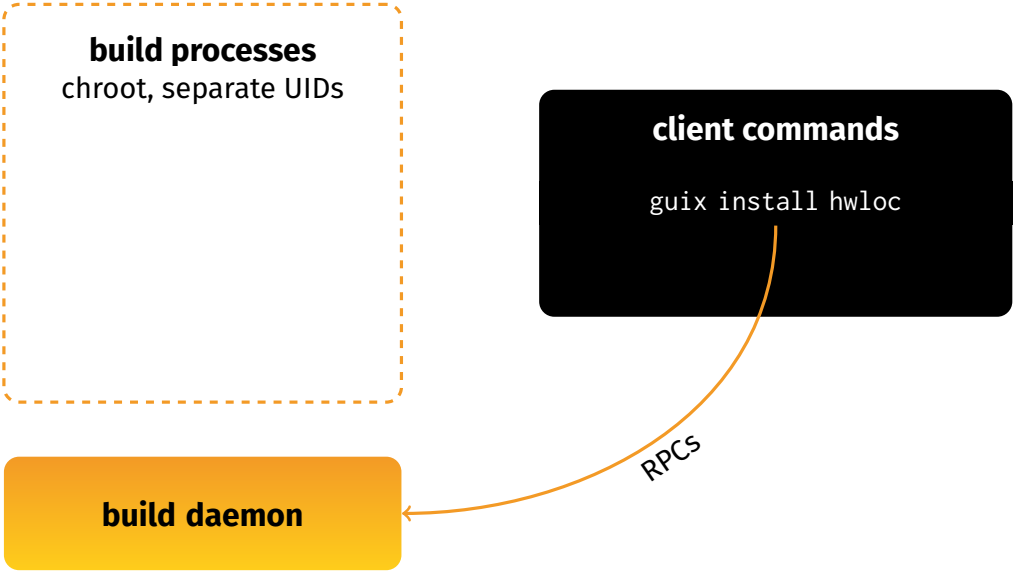
`https://hpc.guix.info/jobs`

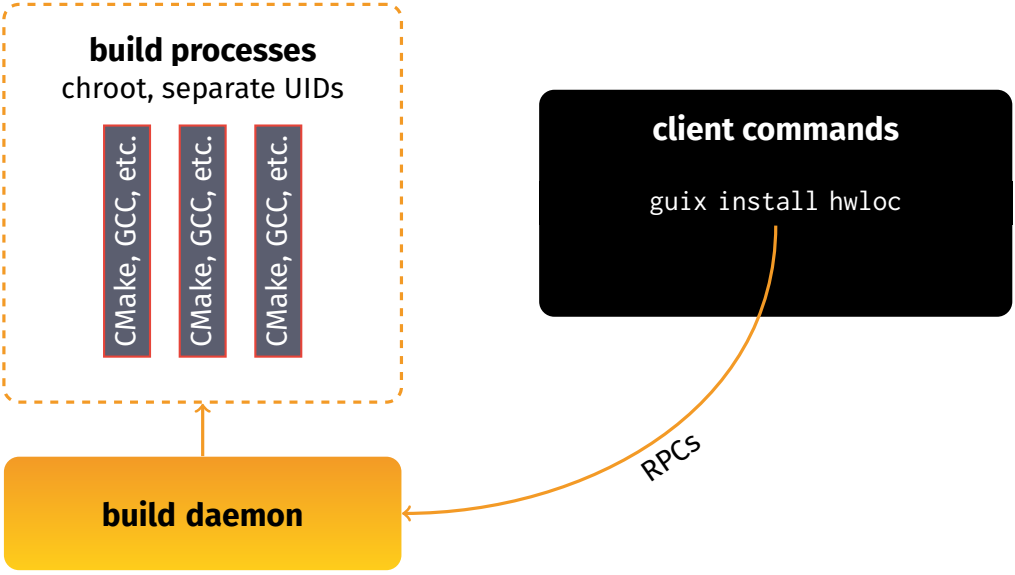
build processes
chroot, separate UIDs

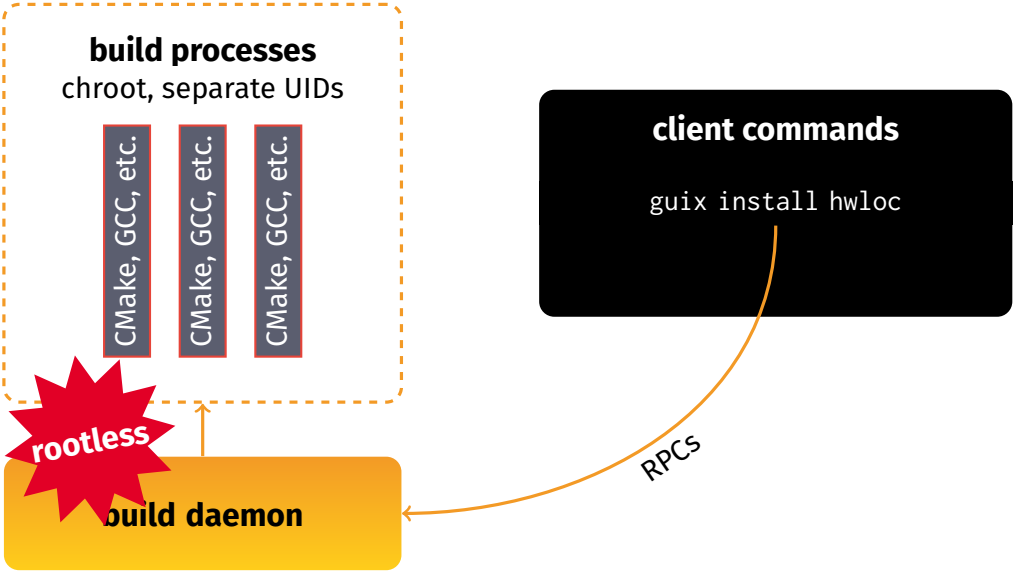
client commands

```
guix install hwloc
```

build daemon


build processes
chroot, separate UIDs
client commands
guix install hwloc
RPCs
build daemon


build processes
chroot, separate UIDs
CMake, GCC, etc.
CMake, GCC, etc.
CMake, GCC, etc.
client commands
guix install hwloc
RPCs
build daemon



https://hpc.guix.info/blog/2025/03/build-daemon-drops-its-privileges/

Guix comes to Adastra

2026Q1

CINES
CENTRE INFORMATIQUE NATIONAL
DE L'ENSEIGNEMENT SUPERIEUR

Wrapping up

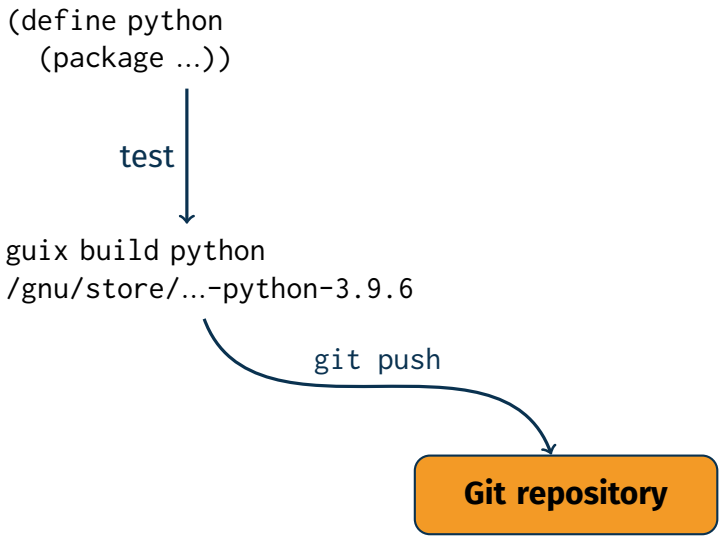
The audience of package managers wasn’t supposed to be sysadmins.

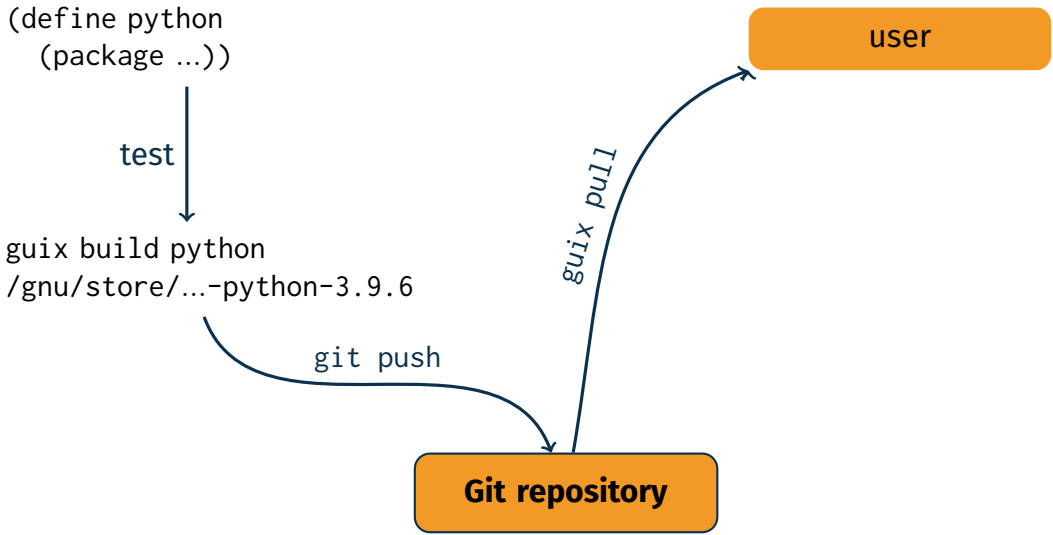

Time to make it
a reality.

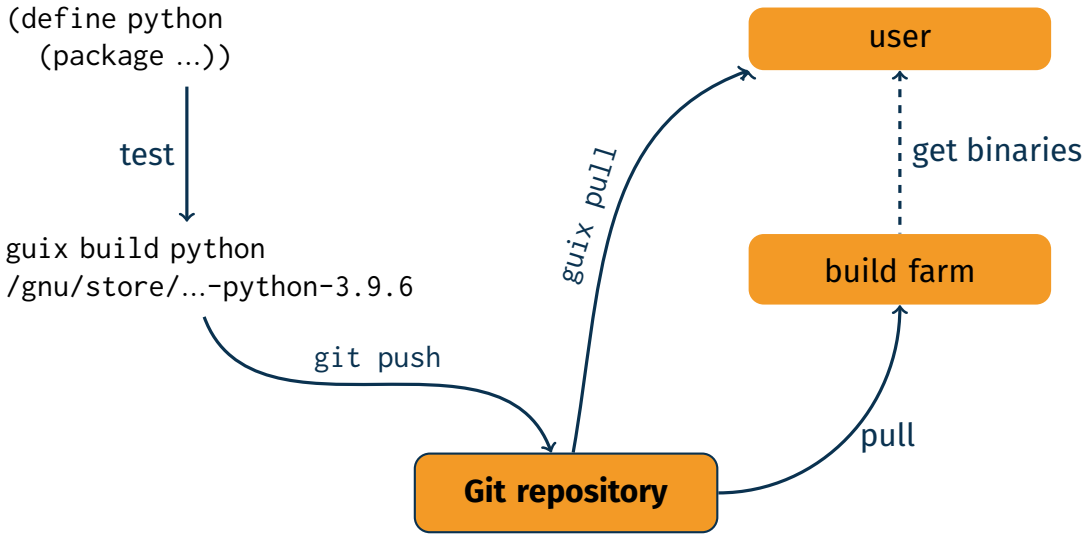
ludovic.courtes@inria.fr | @civodul@toot.aquilenet.fr

https://hpc.guix.info

Bonus slides!


(define python
(package ...))
test
guix build python
/gnu/store/...-python-3.9.6
git push
Git repository


(define python
(package ...))
test
guix build python
/gnu/store/...-python-3.9.6
git push
guix pull
user
Git repository


(define python
(package ...))
test
guix build python
/gnu/store/...-python-3.9.6
git push
guix pull
user
get binaries
build farm
pull
Git repository