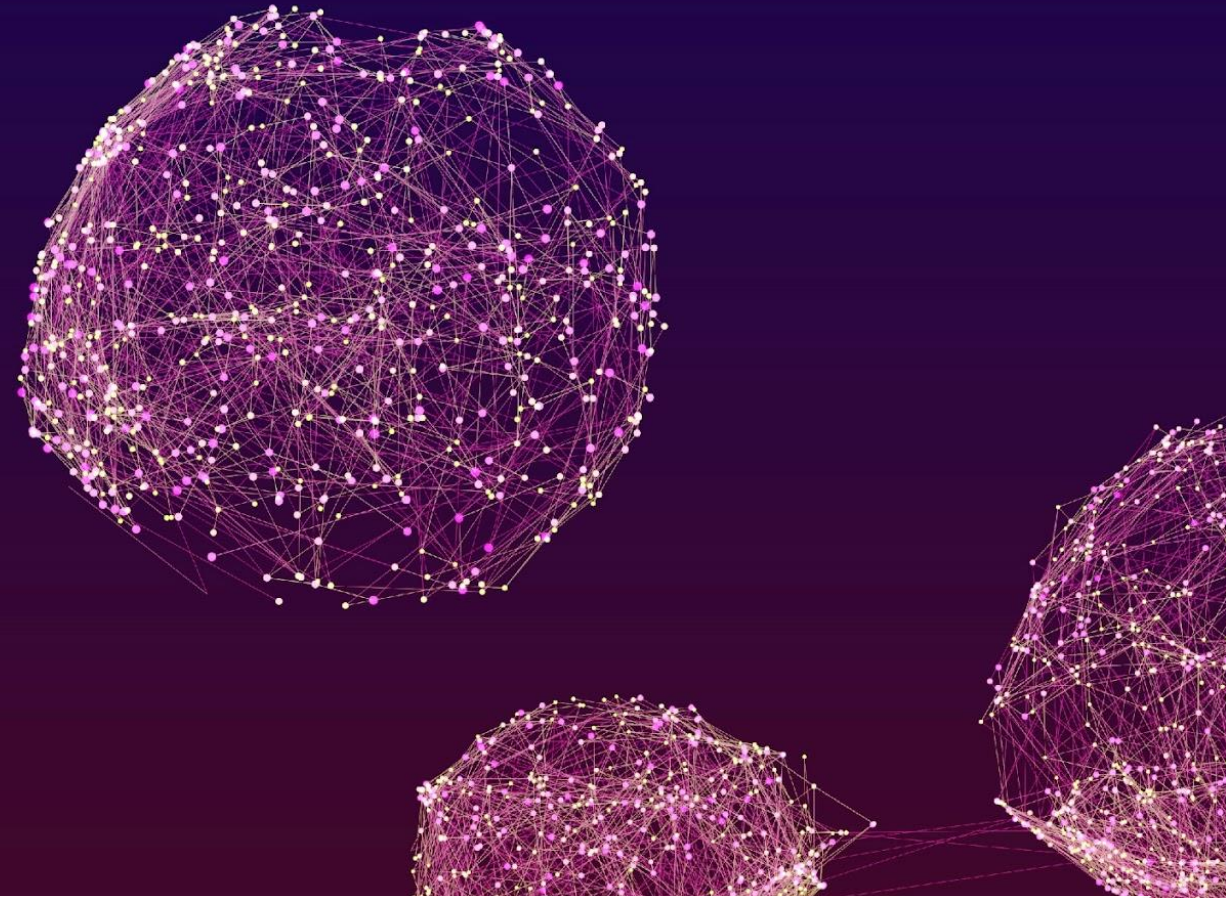


Software Environment on CLAIX

HPC Intro 2025

Felix Tomschi



TECHNISCHE
UNIVERSITÄT
DARMSTADT



NHR4
CES

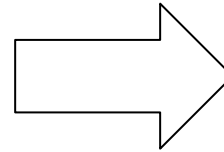
NHR for
Computational
Engineering
Science

Module system

```
export PATH=/opt/intel/bin:$PATH
make CC=icc CXX=icpc FC=ifort
```

```
export PATH=/opt/gcc/bin:$PATH
make CC=gcc CXX=g++ FC=gfortran
```

```
export PATH=/opt/clang/bin:$PATH
make CC=clang CXX=clang++ FC=flang
```



make

- Simplifies usage of software
 - different versions of same software
 - conflicting software, e.g. different compilers
- Software hidden behind modules
 - Handles conflicts & dependencies
 - Adjusts environment variables

Listing loaded modules

```
$ module list
```

```
Currently Loaded Modules:
```

```

1) GCCcore/.11.3.0          (H,C)  5) numactl/2.0.14          9) imkl-FFTW/2022.1.0
2) zlib/1.2.12              6) UCX/1.12.1             10) intel/2022a           (TC)
3) binutils/2.38            7) impi/2021.6.0         (M)
4) intel-compilers/2022.1.0 (C)  8) imkl/2022.1.0         (m)

```

Software usable “at the moment”

```
Where:
```

```

TC:      Toolchain, comprising a compiler, possibly an MPI implementation, and optional mathematical libraries
C:       Compiler
M:       MPI implementation
m:       mathematical libraries
H:       Hidden Module

```

Showing currently available modules

```

$ module avail
  ABAQUS/2017          iimpi/2020a          (TC)
  ABAQUS/2018          iimpi/2020b          (TC)
  ABAQUS/2019          iimpi/2021a          (TC)
  ABAQUS/2020          iimpi/2021b          (TC)
  ABAQUS/2021          iimpi/2022a          (TC,D)
  ABAQUS/2022          iimpi/2022b          (TC)
  ABAQUS/2023          (D) iimpi/2023a          (TC)
  Advisor/2023.0.0     iimpi/2023b          (TC)
  ANSYS/2021R2-test    ImageNet/ILSVRC2012
  ANSYS/2021R2         ImageNet/Winter21_whole (D)
  ...

```

Only shows modules loadable with currently loaded modules

Case sensitive

```

$ module avail CMake
  CMake/3.21.1  CMake/3.22.1  CMake/3.23.1  CMake/3.24.3  CMake/3.26.3 (D)

```

Loading modules

```
$ module load intel-compilers/2023.1.0
```

```
[INFO] Module intel-compilers/2023.1.0 loaded.
```

```
[INFO] Module numactl/2.0.14 loaded.
```

Modules no longer available with newly loaded module

```
Inactive Modules:
```

```
1) UCX/1.12.1      2) imkl-FFTW/2022.1.0    3) impi/2021.6.0
```

```
Due to MODULEPATH changes, the following have been reloaded:
```

```
1) numactl/2.0.14
```

Automatically (re)load dependencies of newly loaded module

```
The following have been reloaded with a version change:
```

```
1) GCCcore/.11.3.0 => GCCcore/.12.3.0      3) intel-compilers/2022.1.0 => intel-compilers/2023.1.0
2) binutils/2.38 => binutils/2.40          4) zlib/1.2.12 => zlib/1.2.13
```

```
$ module list
```

```
1) imkl/2022.1.0 (m)      3) GCCcore/.12.3.0 (H,C)  5) binutils/2.40          7) numactl/2.0.14
2) intel/2022a (TC)      4) zlib/1.2.13           6) intel-compilers/2023.1.0 (C)
```

Module hierarchy

```

$ module avail
----- MPI dependent Modules -----
AlphaFold/2.3.1-CUDA-11.7.0      netCDF/4.9.0
Biopython/1.79                   networkx/2.6.3
...
----- Compiler dependent Modules -----
ACTC/1.1                          libsndfile/1.1.0
at-spi2-atk/2.38.0               LibTIFF/4.3.0
...
----- Core Modules -----
ABAQUS/2017                        iimpi/2020a      (TC)
ABAQUS/2018                        iimpi/2020b      (TC)
...

```

Independent of any other modules (no dependencies)

Toolchains

```
$ module load GCC OpenMPI
```

```
[INFO] Module GCC/11.3.0 loaded.
```

```
[INFO] Module OpenMPI/4.1.4 loaded.
```

Cannot use two MPI
implementations
simultaneously

```
Lmod is automatically replacing "impi/2021.6.0" with "OpenMPI/4.1.4".
```

```
Lmod Warning:
```

```
-----  
The following dependent module(s) are not currently loaded: imkl-FFTW/2022.1.0 (required by: intel/2022a),  
impi/2021.6.0 (required by: intel/2022a)  
-----
```

```
Inactive Modules:
```

```
1) imkl-FFTW/2022.1.0
```

Toolchains

Includes GCC & OpenMPI

```
$ module switch intel foss
```

```
[INFO] Module zlib/1.2.12 loaded.
[INFO] Module binutils/2.38 loaded.
[INFO] Module numactl/2.0.14 loaded.
[INFO] Module foss/2022a loaded.
```

```
$ module list
```

1) GCCcore/.11.3.0	(H,C)	7) OpenSSL/1.1	13) OpenMPI/4.1.4	(M)	19) foss/2022a	(TC)
2) GCC/11.3.0	(TC)	8) libevent/2.1.12	14) OpenBLAS/0.3.20		20) zlib/1.2.12	
3) XZ/5.2.5		9) UCX/1.12.1	15) FlexiBLAS/3.2.0		21) binutils/2.38	
4) libxml2/2.9.13		10) libfabric/1.15.1	16) FFTW/3.3.10		22) numactl/2.0.14	
5) libpciaccess/0.16		11) PMIx/4.1.2	17) FFTW.MPI/3.3.10			
6) hwloc/2.7.1		12) UCC/1.0.0	18) ScaLAPACK/2.2.0-fb			

- Defines a set of modules
- Included modules are (un)loaded when (un)loading the toolchain

Switching modules may leave some inactive modules loaded

- Undesired side-effects
- Purge before loading completely different module version / toolchain

```
$ module purge
```

```
$ module load foss
```

```
[INFO] Module foss/2022a loaded.
```


User module collections

```
$ module purge
$ module load foss CMake Ninja Autotools CUDA Boost Python
$ module save dev
```

No manual purge needed

```
$ ssh login23-1
$ module restore dev
```

- Convenient when having fixed environments for different projects
- Better alternative to (un)loading in shell config (zshrc, bashrc)
 - Should be avoided, might lead to login problems

Finding modules

```
$ module avail CMake
```

```
CMake/3.21.1    CMake/3.22.1    CMake/3.23.1    CMake/3.24.3    CMake/3.26.3 (D)
```

Case insensitive

```
$ module spider cmake
```

Description:

CMake, the cross-platform, open-source build system. CMake is a family of tools designed to build, test and package software.

Versions:

```
CMake/3.15.3
CMake/3.16.4
CMake/3.18.4
CMake/3.20.1
CMake/3.21.1
CMake/3.22.1
CMake/3.23.1
CMake/3.24.3
CMake/3.26.3
CMake/3.27.6
```

Find out dependencies of
specific version

```
$ module spider CMake/3.27.6
```

You will need to load all module(s) on any one of the lines below before the "CMake/3.27.6" module is available to load.

```
GCCcore/.13.2.0
```

Module categories

```
$ module category
```

```
----- List of Categories -----
```

```
cfid chem devel io material math perf tools
```

```
$ module category chem
```

```
----- chem -----
```

```
CP2K (4) GAMESS-US (1) ORCA (2) QuantumESPRESSO (3) VASP (2)
```

```
CREST (1) Gaussian (7) PETSc (2) Siesta (1) xtb (3)
```

Module overview

```

$ module overview
----- MPI dependent Modules -----
ABINIT          (4)  GAMESS-US (1)  mpi4py          (1)  OpenBabel (1)  SciPy-bundle (1)  Wa
CGAL            (1)  HDF5      (7)  mpiP           (1)  OpenMX      (1)  Score-P       (5)
...

----- Compiler dependent Modules -----
ACTC            (1)  Ghostscript    (2)  libdwarf      (1)  METIS        (1)  Qhull         (1)
at-spi2-atk    (1)  giflib        (1)  libelf        (1)  MPC          (1)  Qt5          (2)
...

----- Core Modules -----
ABAQUS         (7)  Dymola        (1)  gomkl         (3)  iompi        (4)  Mathematica  (2)  STAR-CCM+   (6)
Advisor        (1)  EasyBuild    (8)  gompi         (8)  itac         (2)  MATLAB      (9)  Tcl         (1)
...

----- Container Image Modules -----
datascience-notebook (2)  PyTorch (1)  rapids (1)  tensorflow (1)  TensorFlow (1)

```

As spider command,
independent of
currently loaded
modules

Prefer provided software over own installations

Provided software

- Configured and tuned for CLAIX
 - Compiled for host architecture with AVX512 etc.
 - Make use of GPUs if available
 - Better performance
- Has probably already been tested
 - Should work out-of-the-box
 - Less effort for you and supportive staff

Examples

- Especially Python packages via pip
- TensorFlow, PyTorch, Horovod, NumPy, SciPy
 - Available as ordinary modules or containers

Advise

- Search via `module spider <name>`
- Search under help.itc.rwth-aachen.de

```
$ module load TensorFlow
[INFO] You can access the TensorFlow image via $TENSORFLOW_IMAGE
$ aptainer exec -nv $TENSORFLOW_IMAGE <command>
```

Enable
CUDA/GPU
support

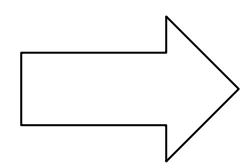
Environment variables

```
$ icc \
  -qopenmp \
  -L/cvmfs/software.hpc.rwth.de/Linux/RH8/x86_64/intel/skylake_avx512/software/FFTW/3.3.10-intel-2022a \
  -lfftw3 \
  main.c
```

Only works with specific intel toolchain

```
$ gcc \
  -fopenmp \
  -L/cvmfs/software.hpc.rwth.de/Linux/RH8/x86_64/intel/skylake_avx512/software/FFTW/3.3.10-GCC-11.3.0 \
  -lfftw3 \
  main.c
```

Only works with specific foss toolchain



```
$ $CC \
  $FLAGS_OPENMP \
  -L$EBROOTFFTW/lib \
  -lfftw3 \
  main.c
```

Works with any intel & foss toolchain

\$EBROOT<module> always refers to base directory of loaded <module>

- Write version- and toolchain-independent scripts
 - Use environment variables if possible
 - Avoid hardcoding names, versions, paths etc.

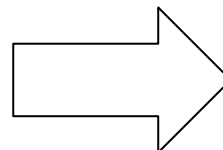
Compiling MPI programs

Only works with Intel MPI

```
$ make \  
CC=mpiicc \  
CXX=mpicpc \  
FC=mpiifort
```

Only works with Open MPI

```
$ make \  
CC=mpicc \  
CXX=mpicxx \  
FC=mpifort
```



Works with Intel & Open
MPI on CLAIX

```
$ make \  
CC=$MPICC \  
CXX=$MPICXX \  
FC=$MPIFC
```

Summary

Key takeaways

- Module system: simplifies management & usage of different software/versions
- Use toolchains (set of modules) when possible
- Prefer provided software over own installations
- Use environment variables in scripts instead of hardcoded names, paths etc. where possible

Commands

- `module list`
- `module (un)load <module>`
- `module avail [<module>]` (case sensitive)
- `module spider <module>` (case insensitive)
- `module switch <module1> <module2>`
- `module purge`
- `module overview`
- `module category [<category>]`

Environment variables

- Compiler: CC, CXX, FC
- MPI: MPICC, MPICXX, MPIFC
- Misc.: FLAGS_OPENMP, FLAGS_FAST
- Generic: EBROOT<module>

More information & help

- help.itc.rwth-aachen.de
- mod.readthedocs.io/en/latest/010_user.html
(user guide for the module system)
- servicedesk@itc.rwth-aachen.de