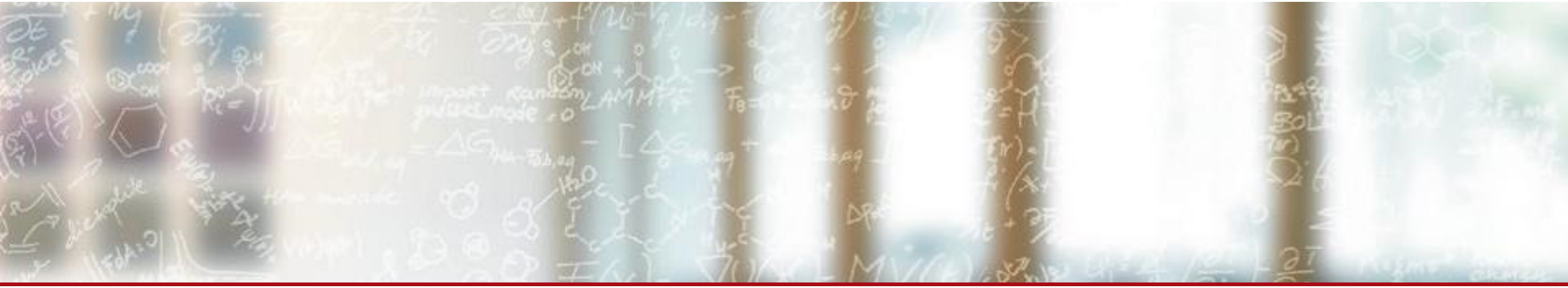
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



User Environments on Alps

Ben Cumming

September 4, 2023

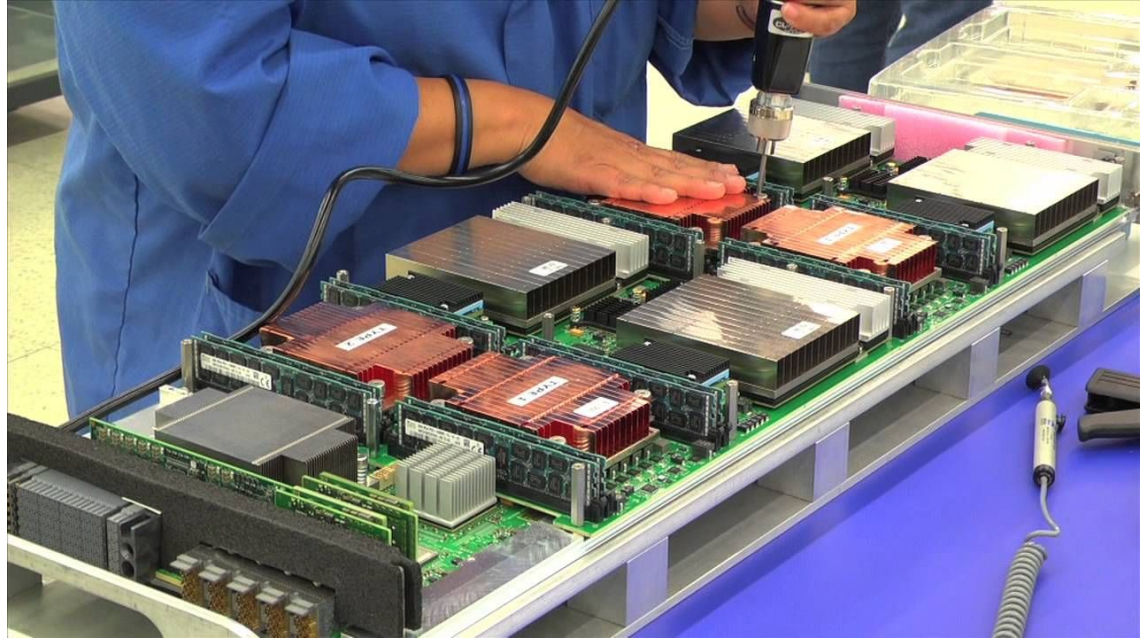
Overview

An overview of the new UENV method for providing programming tools, HPC libraries, applications and tools will presented:

- A (very) brief description of Grace-Hopper on Alps (what's changing?);
- Overview of the user environment on Daint and Eiger in September 2023;
- The Alps UENV for deploying applications and programming tools;
- A practical on-screen demonstration;
- Current status and future work.

Alps changes - the hardware

Daint-GPU nodes



Daint-GPU node has a simple architecture

- 1 Haswell CPU socket
- 1 P100 GPU
- PCI-E connection between host-device
- 1 network interface card (NIC)

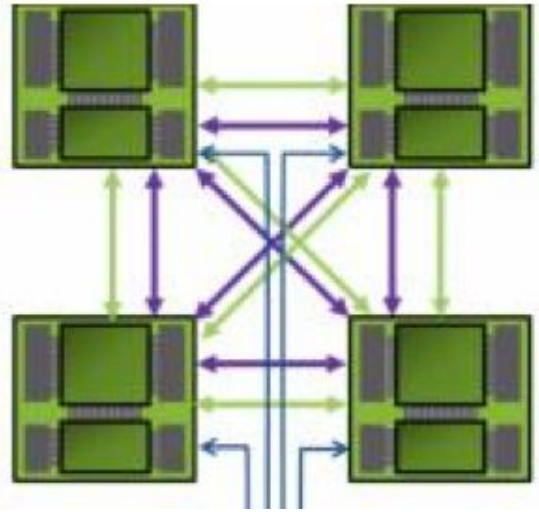
The ratio of 1-1 made allocating MPI ranks relatively simple:

- One rank per GPU + CPU
- Or multiple ranks sharing the GPU using CUDA MPS (multi-process service)

Assigning CPU resources to GPU on x86+A100/H100 nodes is more challenging

- A single CPU socket with multiple NUMA regions is divided between GPUs.

Alps Phase II nodes



Grace-Hopper modules are *conceptually similar*

- 1 Grace CPU socket and one Hopper GPU per module
- Cache-coherent NVLINK connection between host and device memory
- One NIC per module

Each node will have 4 Grace-Hopper modules

- All-to-all cache-coherent memory NVLINK between all host and device memory

The one-to-one CPU to GPU ratio remains

- The 4 modules on a node form an optimised communication network.

Feeds and speeds: Daint-GPU node vs. one GH module

Comparing the raw speeds and feeds of the CPU and GPU

GPU	P100	Hopper	Increase
Bandwidth	700 GB/s	4000 GB/s	5.7x
FP64	4.7 TFlops	34/67 TFlops	7-14x
Memory	**16 GB HBM**	**96 GB HBM3**	**6x**

CPU	Haswell	Grace	Increase
Cores	12	72	6x
Bandwidth	**60 GB/s**	**475 GB/s**	**8x**
FP64	0.49 TFlops	> 2.5 TFlops	5x
Memory	64 GB DDR3	128 GB LPDDR	2x

Data Movement	Daint-GPU	Alps Phase II	Increase
Host-Device	**22 GB/s**	**480 GB/s**	**20x**
Device-Device on node	-	900 GB/s	-
node-node	11 GB/s	23 GB/s	2x

- The Grace-Hopper module delivers 5-10x improvement across the board
- Speedup may be lower or higher depending on the existing bottlenecks.

Alps changes - the software stack

User Lab Transition to Alps

In Q1-Q2 2024 CSCS' User Lab production cluster(s) will move from Daint to Alps

- Daint-GPU workloads will be moved to the Grace-Hopper nodes that will be installed in the Phase 2 expansion of Alps
- This will mean some big changes for CSCS and User Lab community
- Early evaluation of a Grace-Hopper node by CSCS gives us confidence that software and applications will be ready on time.
- CSCS will provide multiple vClusters on Alps:
  - These offer better isolation of resources and specialisation than the daint-mc and daint-gpu SLURM partitions today.
- One key change will be to the user environment on Alps
  - CSCS will continue to provide the Cray Programming Environment (CPE)
  - CSCS will provide improved support for HPC containers
  - CSCS will deploy programming environments, applications and tools using a new UENV approach.

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

The User Lab Software Stack Today

The Piz Daint user environment: one size fits all

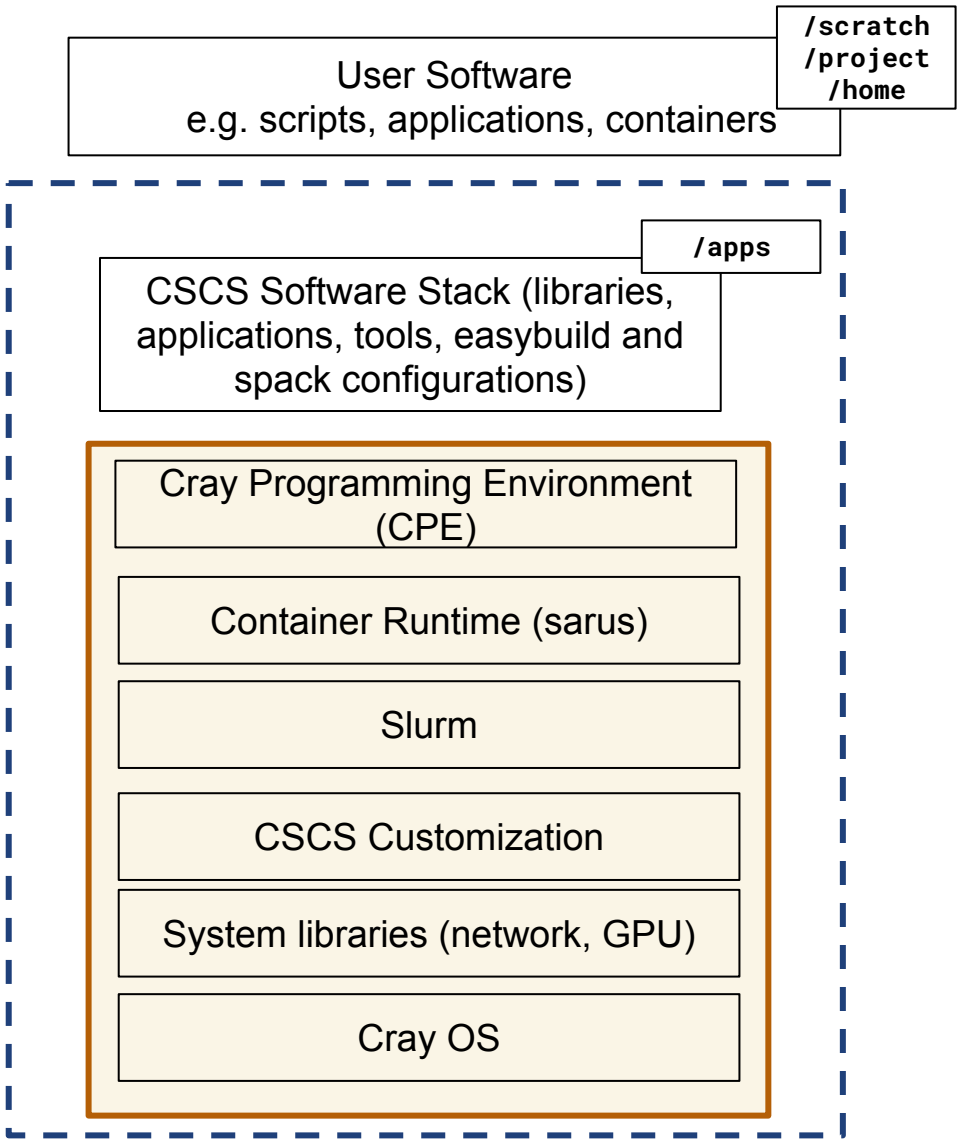
Interactions with Daint use the well-established, tried-and-trusted HPC workflow:

- Log in via ssh to a shell with a programming environment loaded
- **Manipulate the environment using modules**
  - Select Cray-provided compilers, libraries and tools
  - Select CSCS-provided libraries, tools and applications
- Use shared resources including storage and job scheduling.

As the number and variety of use cases expand

the “one size fits all model” does not scale

The user environment on Daint and Eiger: layer cake



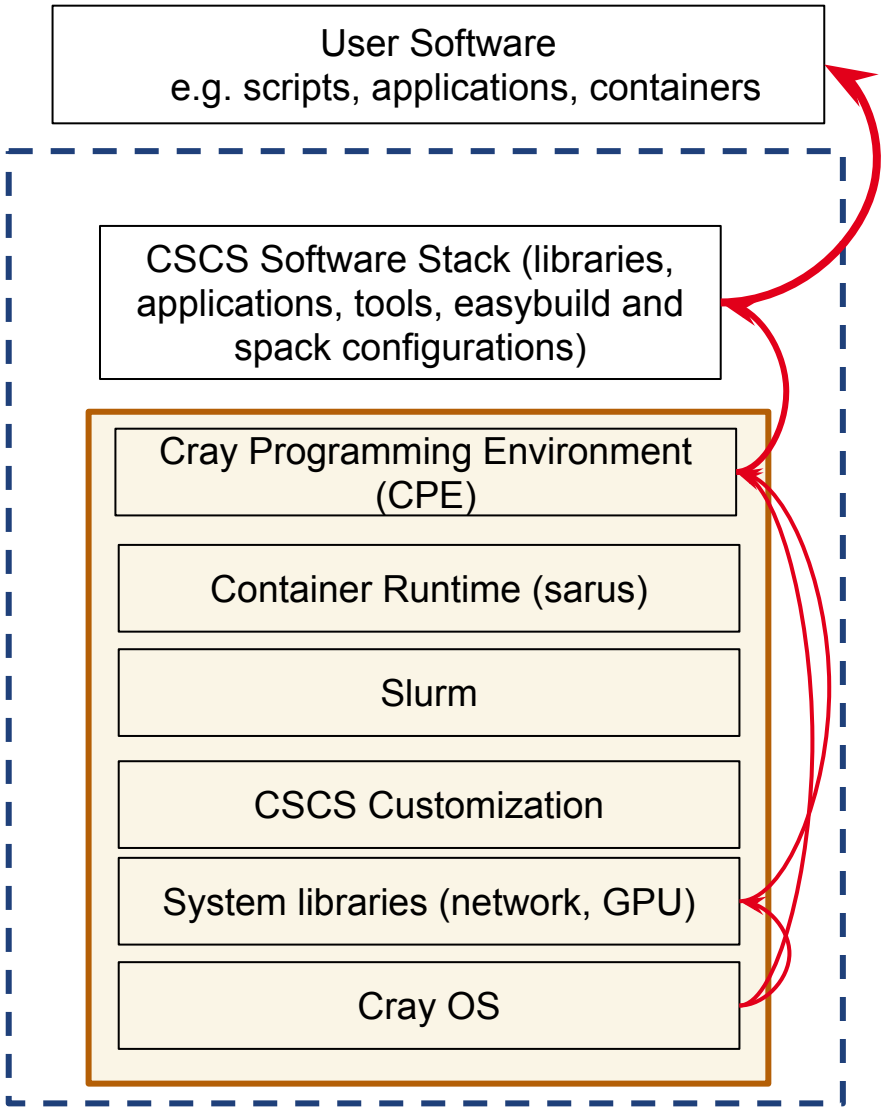
1. Essential services are installed on top of Cray OS
2. The Cray PE is installed as part of the underlying “node image”
3. CSCS provides software built with the Cray PE
4. Users build their software and workflows on top of that.

A change to one layer during an upgrade affects every layer above:

- A new Cray PE change often requires rebuilding the CSCS stack and user software

CSCS has developed tooling (ReFrame) and a very comprehensive test suite and CI/CD for maintaining the quality of this integration

Maintenance challenges



The Cray Programming Environment is complex by necessity:

- Modules provide a combinatorial set of libraries and tools that serve as many use cases as possible on an increasing number of hardware types.
- Integration is provided by HPE: once an issue is identified HPE have to fix the issue in a future release
  - Long latency between issues reporting and the fix available on Daint.
- Each new release requires extensive testing to check that issues have been fixed
  - And to identify the inevitable new issues

The Cray PE is the best configuration from any vendor in my experience.

These challenges affect all HPC clusters.

Stability vs. Bug Fixes and New Features

Any feedback that in your opinion can help us improve the HPC environment?

Would it be possible to keep older versions?

I am very satisfied with the HPC environment on Piz Daint

Compilers that support the newest C++ standards as well as possible

I need a stabler environment: older versions of the software tools disappear too quickly, which means I have to rebuild my stack every few months.

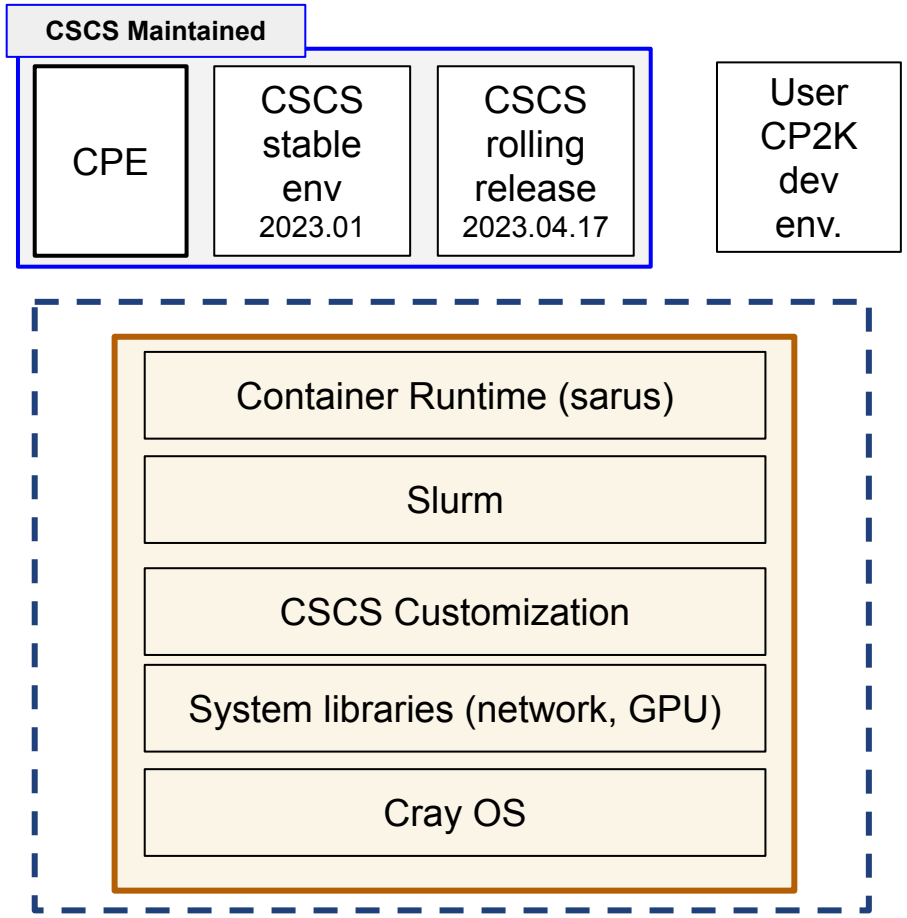
Please regularly update C++ and CUDA compilers

By providing an environment on CPE it is very difficult to meet all requirements

- Regular updates are required to fix bugs, maintain security and provide updated versions of tools.
- The latest versions of compilers can't be installed before they are packaged by HPE and tested by CSCS.
- It is impractical to maintain:
  - Full stacks on top of more than one CPE
  - More than 2-3 CPE on a system

UENV on Alps

The UENV approach



Provide multiple user environments

- Login to a simpler environment
  - Cray OS with slurm + container runtime + drivers
- Multiple environments are available:
  - The familiar CPE via modules: `module load cray/22.08`
  - CSCS-provided user environments
  - User-built user environments
- Each environment is contained in a single file
  - Shared in an artifactory or stored on a filesystem.

The environments are **independent**

The environments are built on top of the base-image - not the Cray PE.

Benefits of UENV

- Each environment is a single image (a SquashFS file)
  - Can be managed in a registry/artifactory (not on the file system)
  - Performance is decoupled from the file system
- Each environment is defined by a simple declarative recipe
  - alps-spack-stacks/blob/main/recipes/gromacs/a100/environments.yaml
  - Only key dependencies are libfabric and Slurm – the exact same environment can be rebuilt after major system upgrades.
- Each environment has a very small set of system dependencies
  - Only need rebuilding when **libfabric** or **SLURM** are changed.
- Deployed and updated independent
  - Of CPE releases
  - Of one-another

What is UENV

UENV are supported by a set of tools

- Stackinator: a tool for generating uenv images from a declarative recipe
  - Documentation: eth-cscs.github.io/stackinator/
  - Used by CSCS to build the environments
  - Available for advanced users to build their own images
- A GitHub repository with CSCS recipes
  - github.com/eth-cscs/alps-spack-stacks
  - CI/CD pipeline that builds, tests and deploys the images
  - Documentation of each environment is maintained alongside the recipe
    - eth-cscs.github.io/alps-spack-stacks/
- SLURM integration: a SLURM plugin that manages loading UENV images on compute nodes
  - github.com/eth-cscs/slurm-uenv-mount/
- CLI tools:
  - `sqashfs-mount`: low level tool for mounting environments
  - `uenv`: a command line tool for interacting with environments

Types of UENV

There are three categories of tools.

Mounted at `/user-environment`:

- Programming Environments
  - Compilers, MPI, libraries like HDF5, FFTW, OpenBLAS
  - Can be application specific, e.g. ICON
- Application Environments
  - E.g. GROMACS, CP2K, NAMD, ...
  - Provide applications and the tools required to build them

Mounted at `/user-tools`:

- Debuggers: e.g. DDT
- Visualisation: e.g. ParaView
- Profilers

Hands on Demo

Status of work

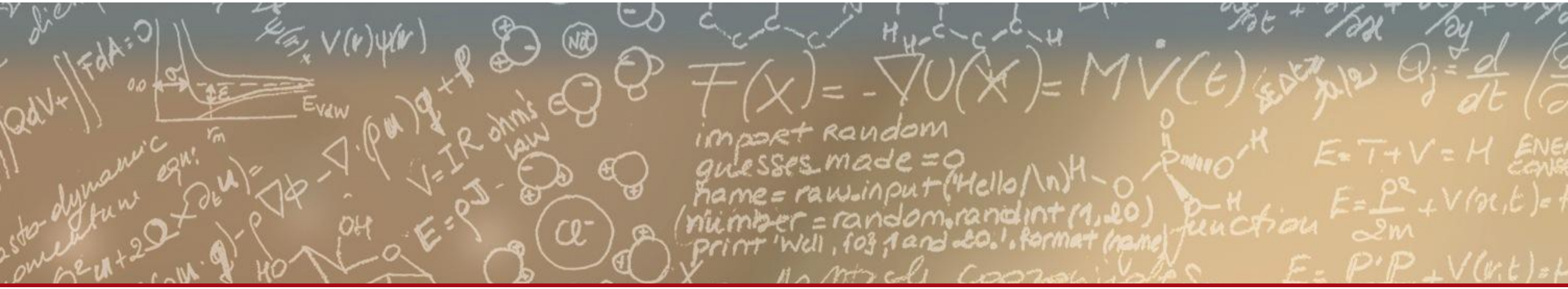
Status of work

- SLURM plugin and low-level tools are installed on Eiger and internal development vClusters
- Used on GPU and CPU partitions by internal development teams
- Deployed for MeteoSwiss
- CI/CD pipelines are generating and testing images

Under development:

- Deployment stage of CI/CD pipelines
- Documentation
- Comprehensive testing
- Support for Grace-Hopper

In Q4 2023 a vCluster will be created on Alps for users to test, evaluate and provide feedback on the UENV and workflows.


import random
guesses_made = 0
name = raw_input('Hello! What is your name?\n')
number = random.randint(1, 20)
print 'Well, ' + name + ', I am thinking of a number between 1 and 20.'

Thank you!