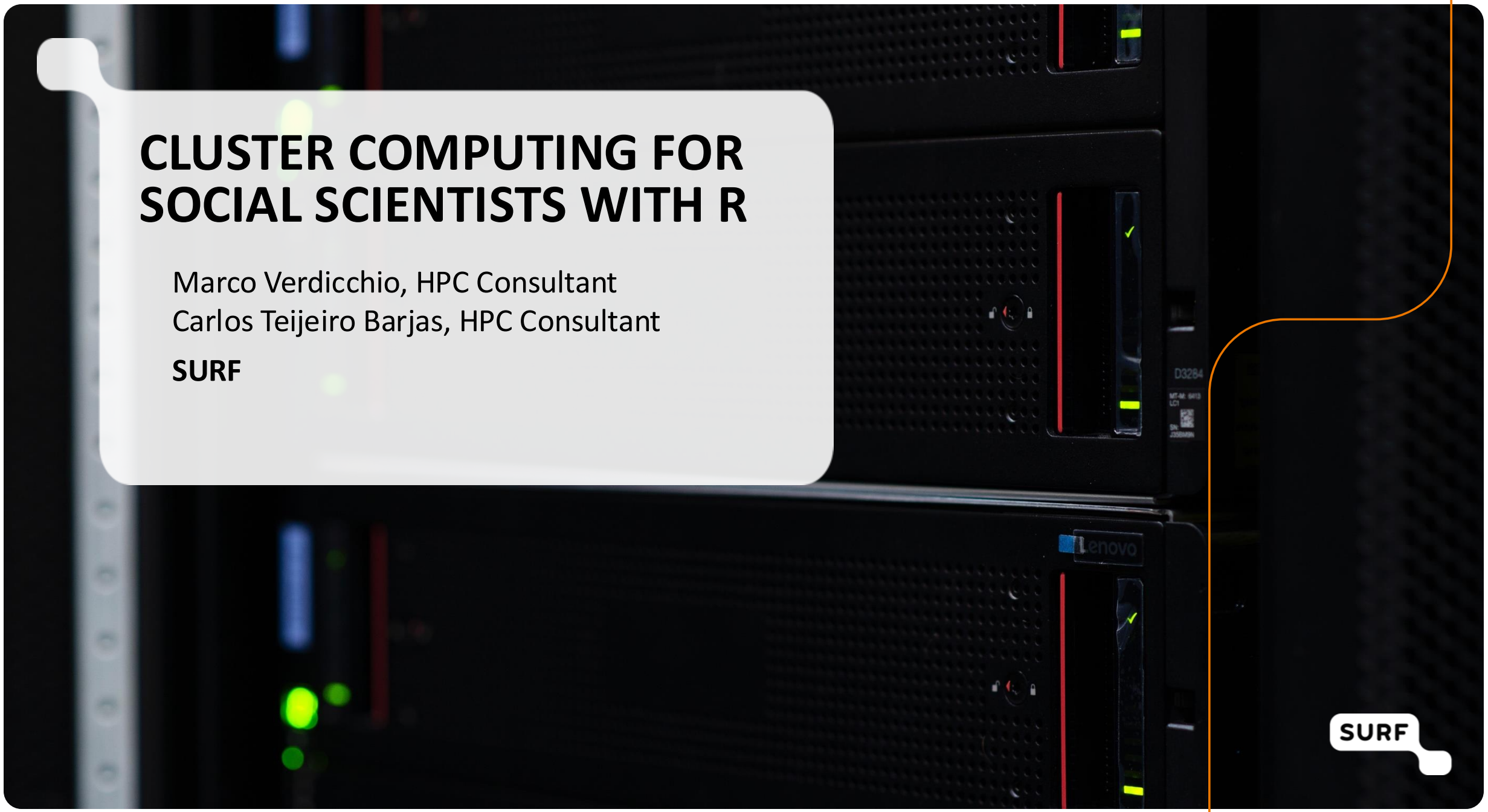


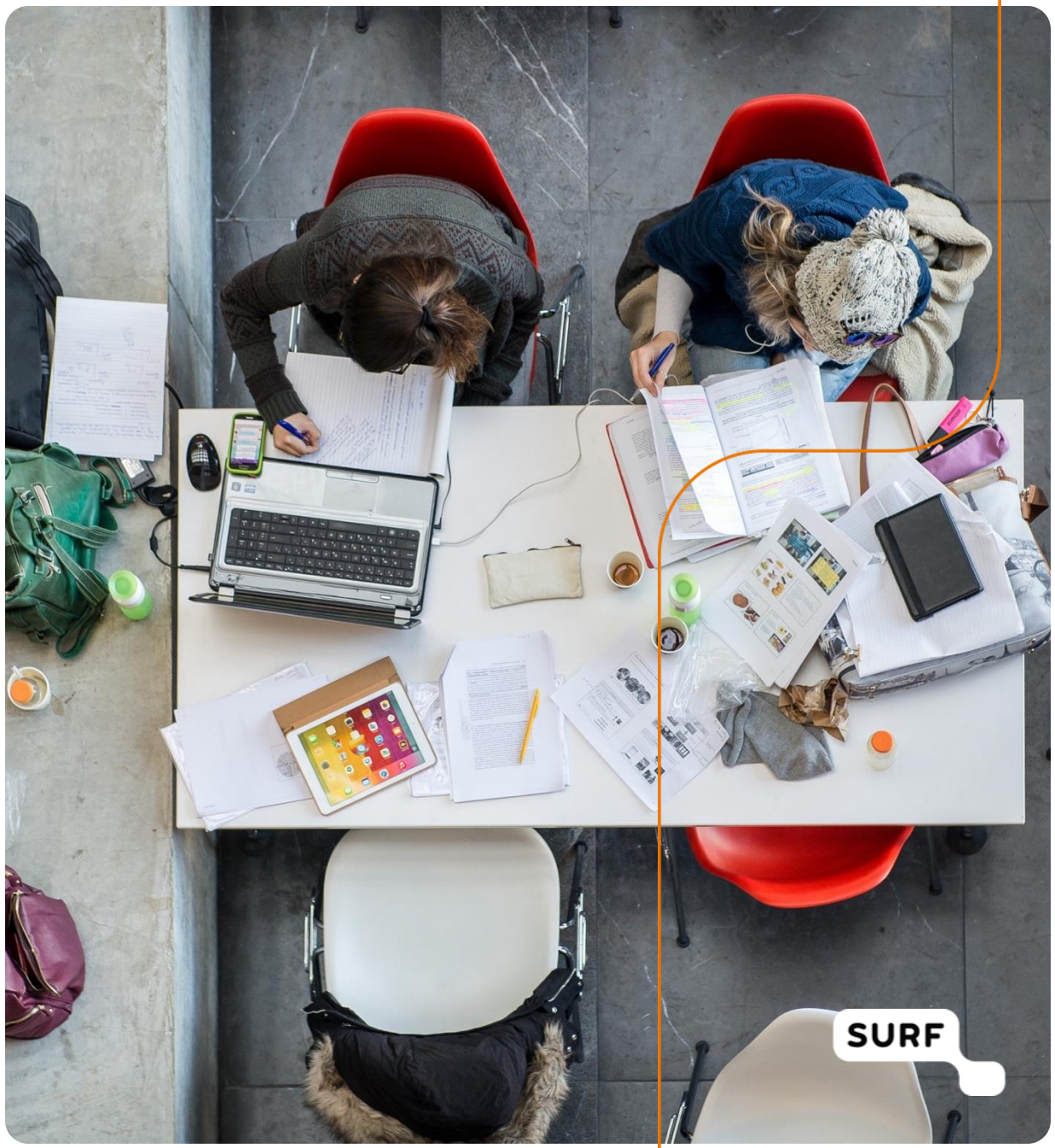
CLUSTER COMPUTING FOR SOCIAL SCIENTISTS WITH R

Marco Verdicchio, HPC Consultant
Carlos Teijeiro Barjas, HPC Consultant

SURF

SURF is an ICT cooperative for education and research

SURF is a cooperative association of Dutch educational and research institutions.
We work together to acquire or develop the best possible digital services, and to encourage knowledge sharing through continuous innovation.
with the aim: making education and research better and more flexible.



Amsterdam Science Park

AMOLF

CWI
Centrum Wiskunde & Informatica

netherlands eScience center

UNIVERSITY OF AMSTERDAM

egi

SURF

SURF

Reliable, secure and innovative ICT infrastructure

Digital innovation and transformation of education and research

Knowledge exchange and trainings

Services development and integration with EU initiates

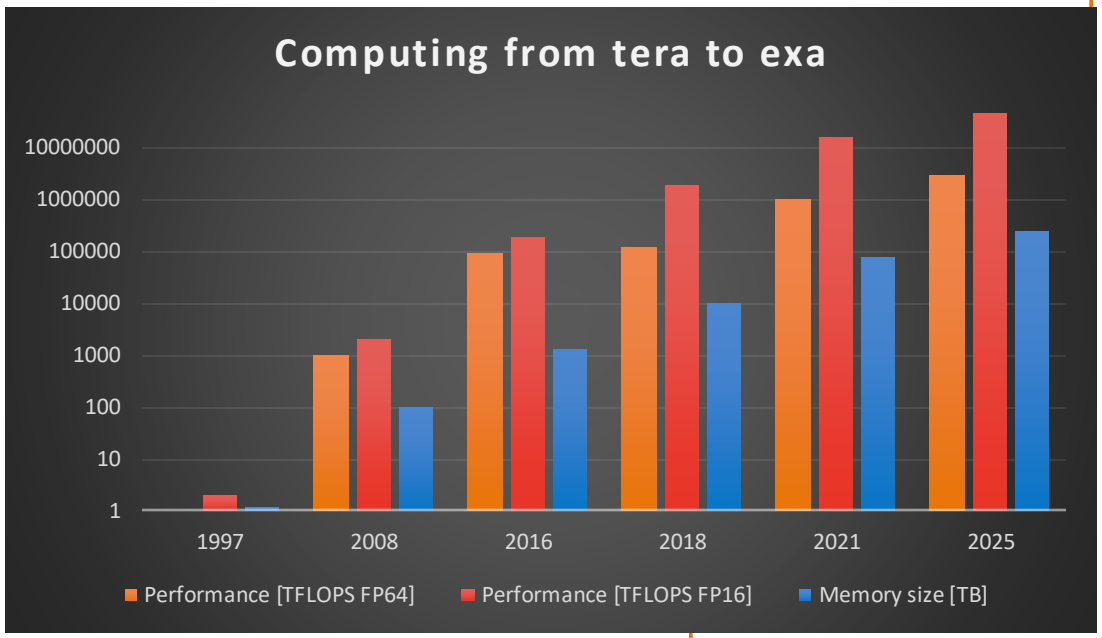
Traditionally driven by Scientific Challenges

- From High Energy Physics to atomic and molecular physics
- Life sciences (cell biology)
- Social science
- From the big bang to astronomy
- Theoretical chemistry and materials science
- Earth (climate and geophysics)
- Life and biodiversity

And further stimulated by the needs within the SURF cooperative

It’s far more than just big systems

- As scientific problems become more complex:
  - collaborations will grow
  - data sizes will grow
  - e-infrastructure HPC systems will grow
  - requires integration of compute and data



- For todays and future e-infrastructure development this means advances and collaborations in the complete e-infrastructure ecosystem:

  community services + software (system, middleware, libraries, applications) + algorithms + programming models + workflows + hardware (compute, data, network) + datacenter + operations + support + data management + training + education + integration + federation

SURF Infrastructure Ecosystem

Digital workflow management

Research Software Directory

Application enabling

AI / GPT-NL

Extended Reality

Futuring

Netherlight

Autonomy

Data Storage

HPC

HTC

Quantum

Neuromorphic

Data management

Sustainability

Network to Institutions

Expertise

Security

Identity

Infra Interoperability /collaboration

eosc

Jules Verne

LUMI

NL Regional TIER-2

aws

Azure

Google Cloud

SURF

Snellius, the Dutch National supercomputer

- Snellius is a cluster of heterogeneous nodes with capabilities for HPC, data-processing, ML/DL
  - 1,263 CPU node with 128/192 cores
  - 72 GPU node with 4 Nvidia A100
  - 124 High memory nodes (1-8TB mem/node)
  - High performance file system
  - Infiniband HDR100 connectivity
  - Optimised software stack

- Continuously update and extended

AMD Lenovo NVIDIA

Snellius, the Dutch National supercomputer

AMD Lenovo NVIDIA

580 + 21
AMD ROME CPU compute nodes

36 + 36
Nvidia A100 GPU compute nodes

76,8K + 2.7K
Cores

144 + 144
A100 GPUs

3PF +0.15PF
CPU Rpeak

2,8PF + 2,8PF
GPU Rpeak

Phase 1+1A

NL-SAS, NVMe and SSD
storage

>13.3PiB + 2PiB
Usable storage capacity

>300GiB/s
Sequential perfromance

Storage

714 + 72
AMD NG CPU compute nodes

137K + 14K
Cores

5,63PF + 0,5PF
CPU Rpeak

Phase 2+2A

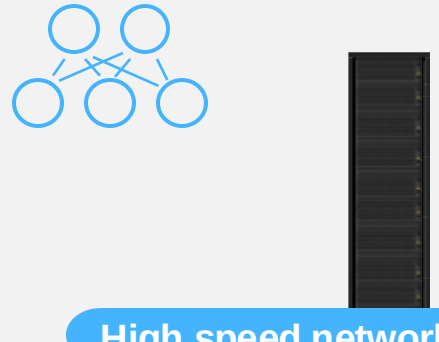
66 + 28
Nvidia Hopper GPU compute nodes

264 + 112
GPUs

20PF
GPU Rpeak

Phase 3

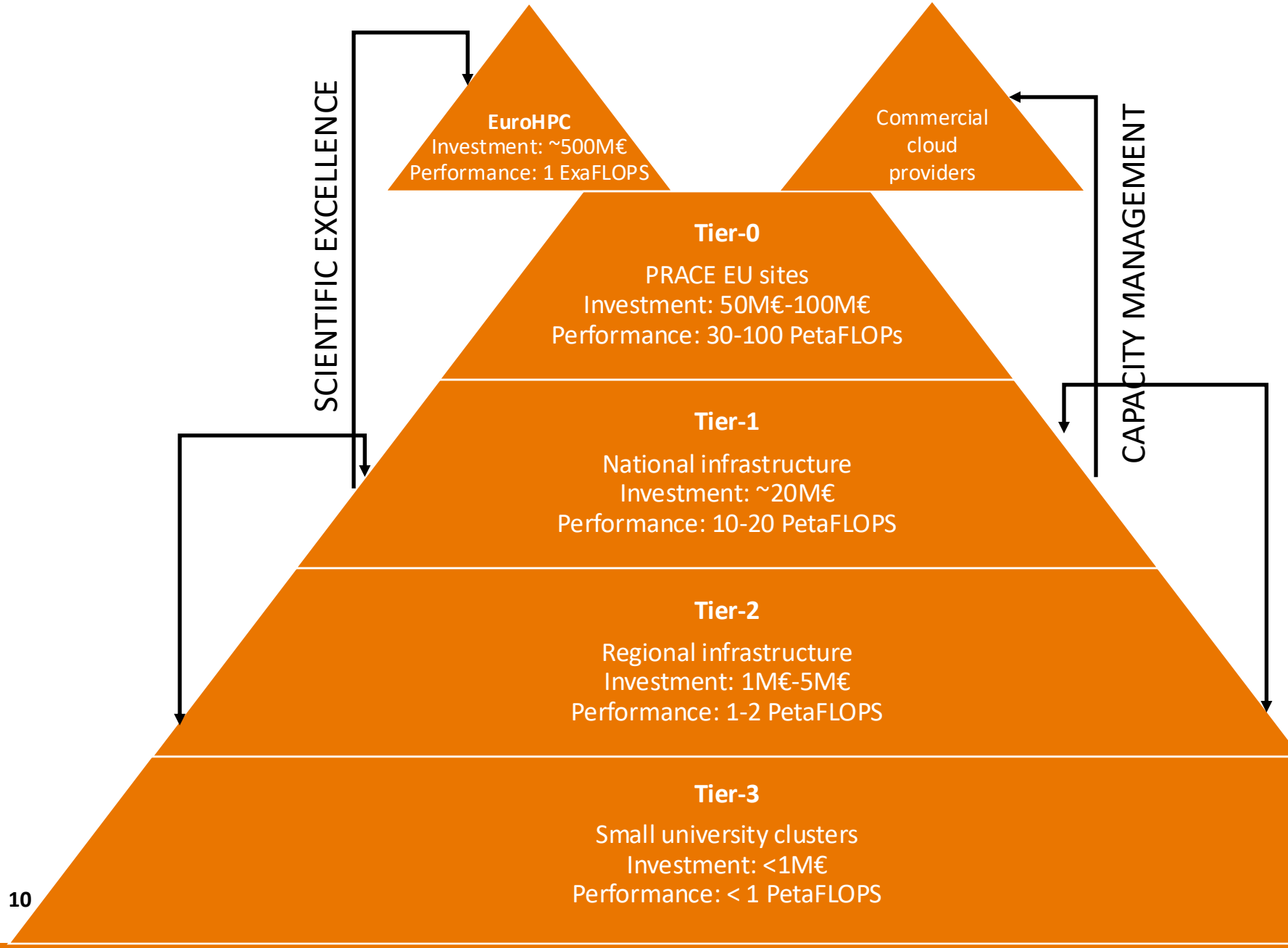
HDR and NDR
Fat-Tree Mellanox infiniband fabric



High speed network

SURF

Tiered HPC model – HPC federation



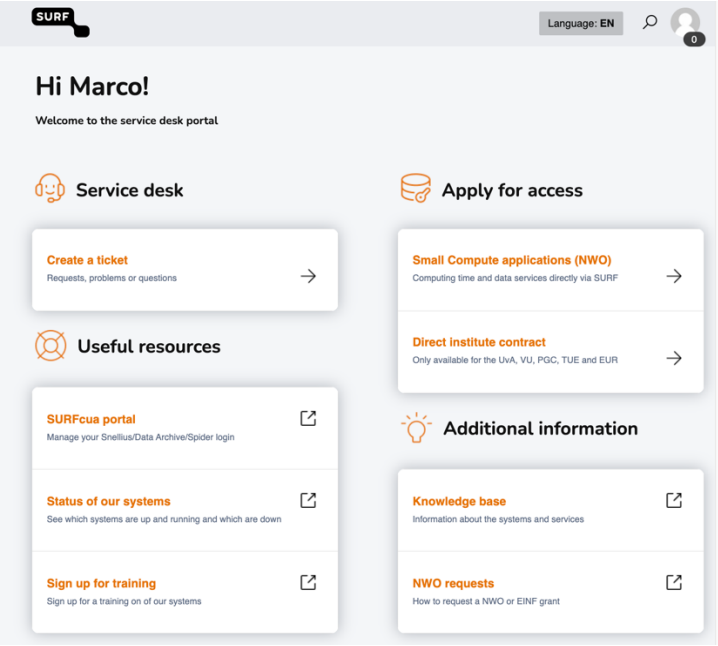
High-Performance Computing support

- **User Support and Issue Resolution**
  Assist users with troubleshooting technical problems and resolving issues related to the HPC system, services and applications.

- **System Access and Account Management**
  Handle user account creation, permissions, and access control to ensure secure and efficient use of the system.

- **System Usage support**
  Manage software installation and configuration and assist users in job submission and scheduling.

- **National and international projects and collaborations**



https://servicedesk.surf.nl/

Co-development research applications and workflows

- **Promising Applications programme**
  - Target Dutch HPC scientific applications
  - Support for codes optimization, benchmark and profiling
  - HPC specialists on CPUs, GPUs and specialized hardware

- **Applications Roadmap project**
  - Identify current challenges and bottlenecks of selected domains
  - Evaluate current status of scientific applications/workflows
  - Define medium/long term roadmap of applications evolution to support development and infrastructure planning

Innovation activities

- **EAR (energy aware runtime) to monitor and manage energy consumption**
  - Energy management framework on Snellius
  - Collaboration with BSC team and LRZ on testing and validation

- **Experimental Technology platform: Hardware assessment and Co-design**
  - Access to experimental hardware from AMD, Intel, IBM, Nvidia
  - Community driven technical environment for experiments
  - Culture of experimentation & Concept Validation

High-Performance Machine Learning Team

- **HPML team Activities**
  - Research support and outreach
    - Courses, bootcamps, workshops
  - General ML consulting on HPC systems
  - Algorithms scaling and efficiency
    - Efficient inference
    - Distributed and efficient data pipelines

- **Projects examples**
  - AI4Science. Applying AI models to boost classical HPC modelling
  - AI assisted operational tools and support
  - GPT-NL main contributor

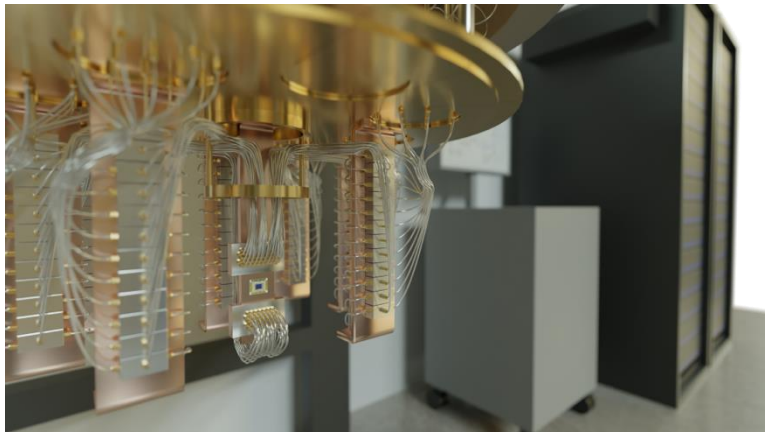
_ / Data sharing / Generative AI /

GPT-NL boosts Dutch AI autonomy, knowledge, and technology

Artificial intelligence Data sharing National safety Operations and human factors

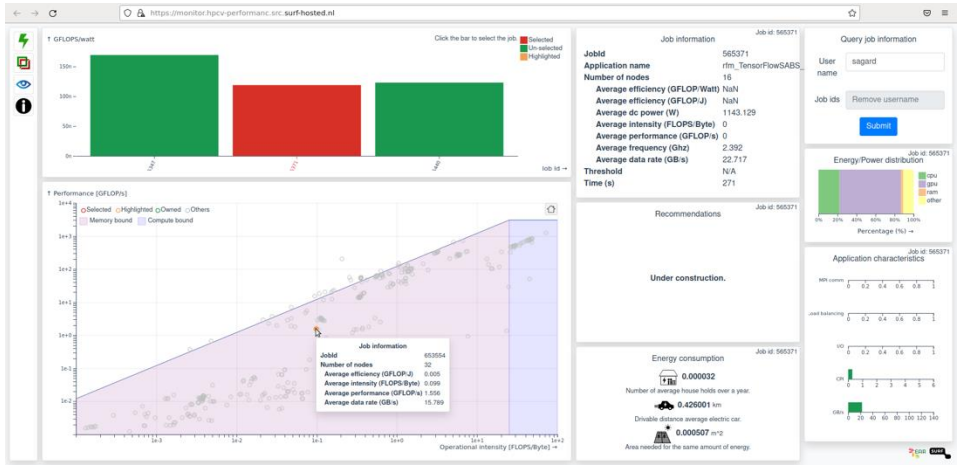
Large language models such as ChatGPT offer promising technical opportunities to address societal challenges. At the same time, there are concerns about the legal and ethical aspects of this development, as most of the models are developed by foreign big tech. TNO is therefore working with SURF and NFI to develop its own Dutch language model. GPT-NL should strengthen our strategic autonomy in – and knowledge of – AI, Data Science, and

Visualisation Team Activities



XR innovation

Trend reporting, XR Developer Network,
National XR Day, internal technical consulting,
Quantum Computing lab in VR demo, …



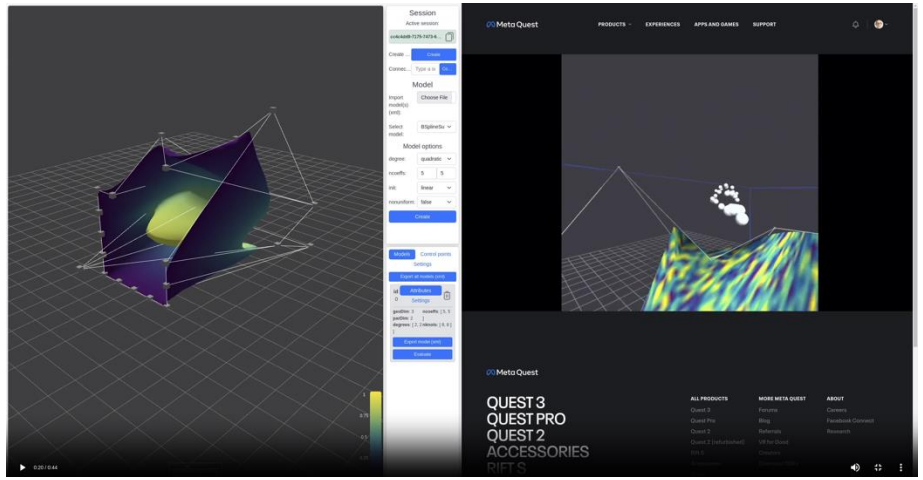
Snellius EAR and Reframe dashboards

Energy, performance, system tests

HEREDITARY

HEREDITARY project

(EC-funded 2024-2028, Uni Padua, TU Graz, RadboudUMC, SURF, …)
Develop secure distributed system for multimodal health data linkage, including federated machine learning.
Viz team working on visual analytics interface with TU Graz



IgAnet

Interactive Design-through-Analysis
& computational steering

HPC Training for researchers and industry

Training courses for research

Want to get started with our systems but lack the necessary knowledge? We regularly organize hands-on systems training courses at our offices in Utrecht and Amsterdam or at your education or research institution. You can also include the training courses in the educational programme of your institution.

Systems training

Learn how to work with our systems.

- Supercomputing
- HPC Cloud
- Data management

Technical skills

- Parallel programming
- Machine learning
- Big data
- Visualisation
- Software containers

PRACE

CompBioMed

SURF

New services / service components for improved usability

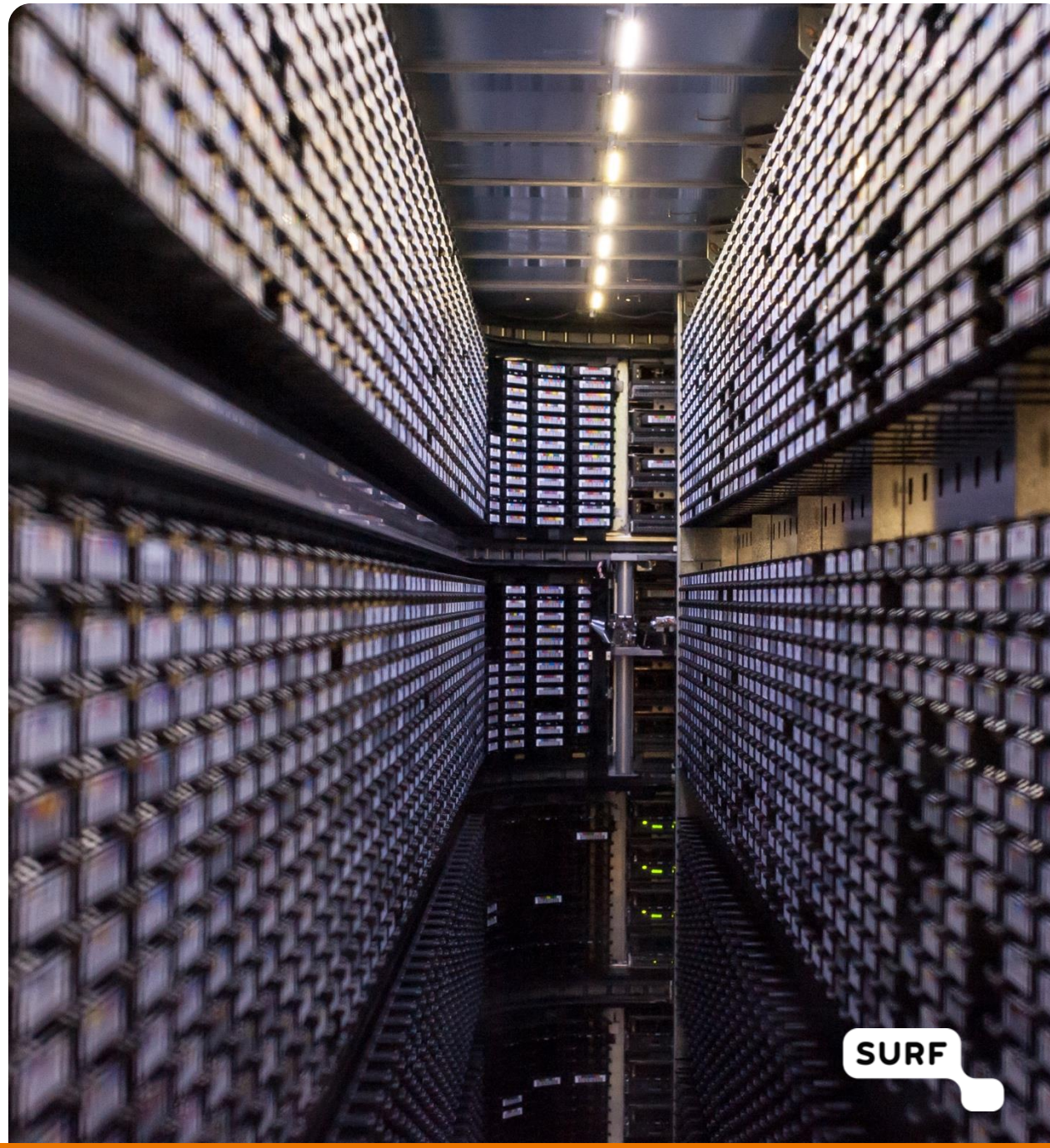
- **UI and Gateways to access HPC resources**
  - JupyterHub on Snellius
  - OpenOnDemand GUI for compute services

- **Secure Supercomputer for sensitive data processing**
  - CBS Microdata are linkable data at the level of individuals, companies and addresses (highly sensitive!!!)
  - HPC environment that meets the requirements of CBS in legal, technical and security requirements

OSSC the ODISSEI Secure SuperComputer

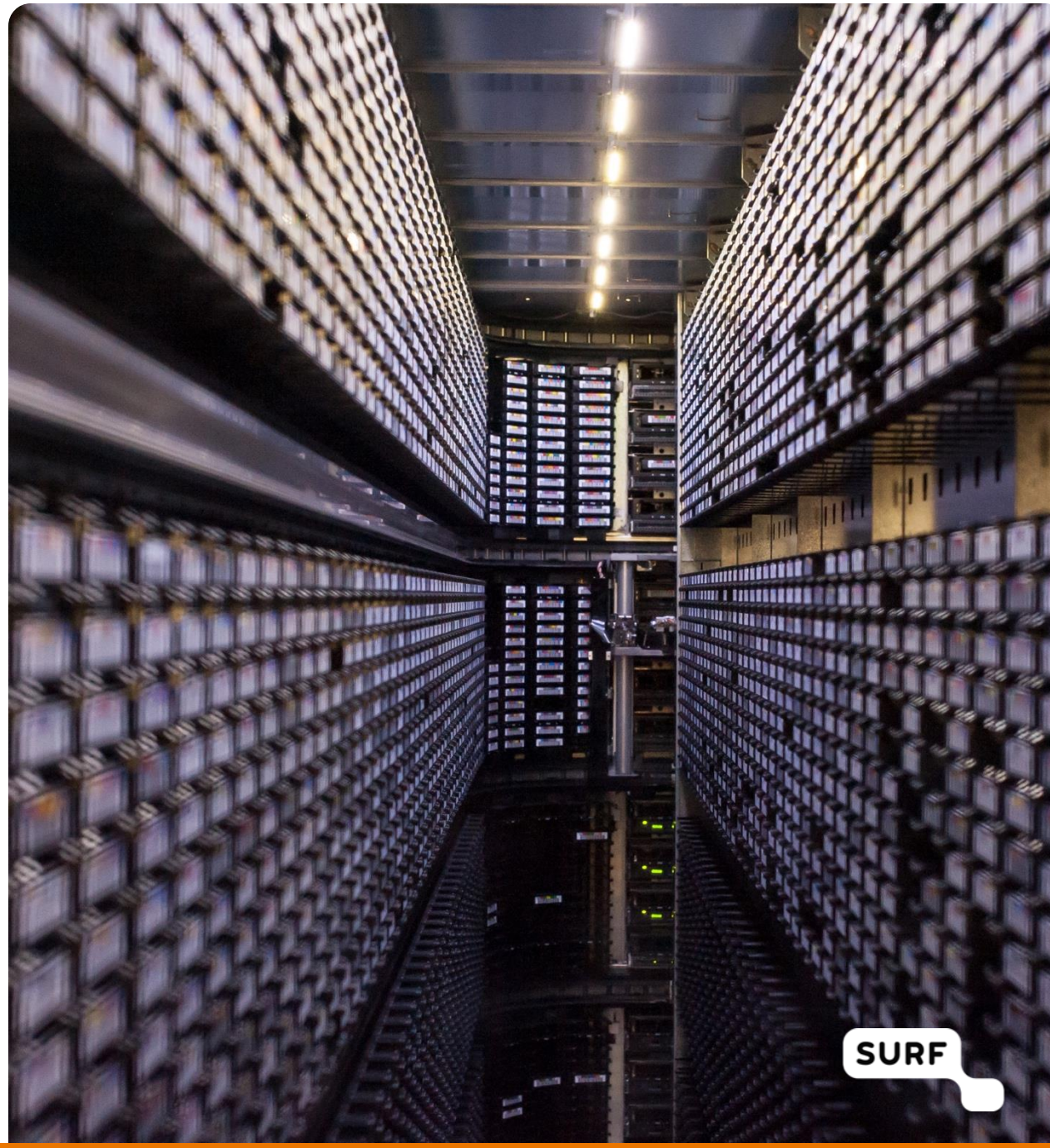
- ODISSEI, CBS and SURF co-creation project, PoC 2018

- Research with extremely sensitive
  - Microdata are linkable data at the level of individuals, companies and addresses
  - Access to highly sensitive CBS data gives great opportunities for globally competitive research

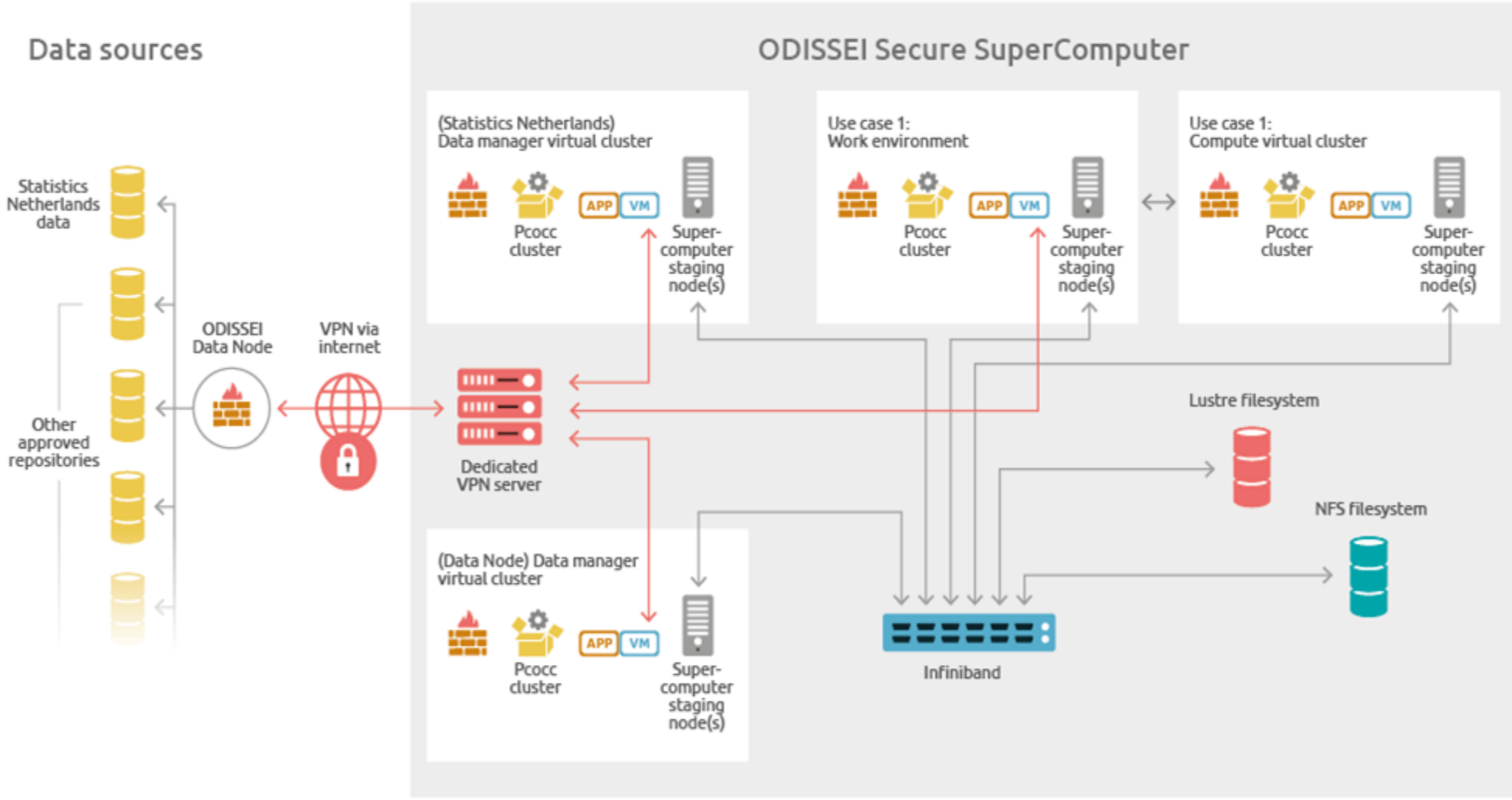


OSSC the ODISSEI Secure SuperComputer

- CBS microdata
  - available to Dutch universities, scientific organisations, planning agencies and statistical authorities within strict conditions
  - Access through CBS Remote Access secure but with limited compute capabilities
- HPC environment that meets the requirements of CBS in legal, technical and security requirements
- The ODISSEI Secure Supercomputer (OSSC) is an “enclave” of Statistics Netherlands within SURF



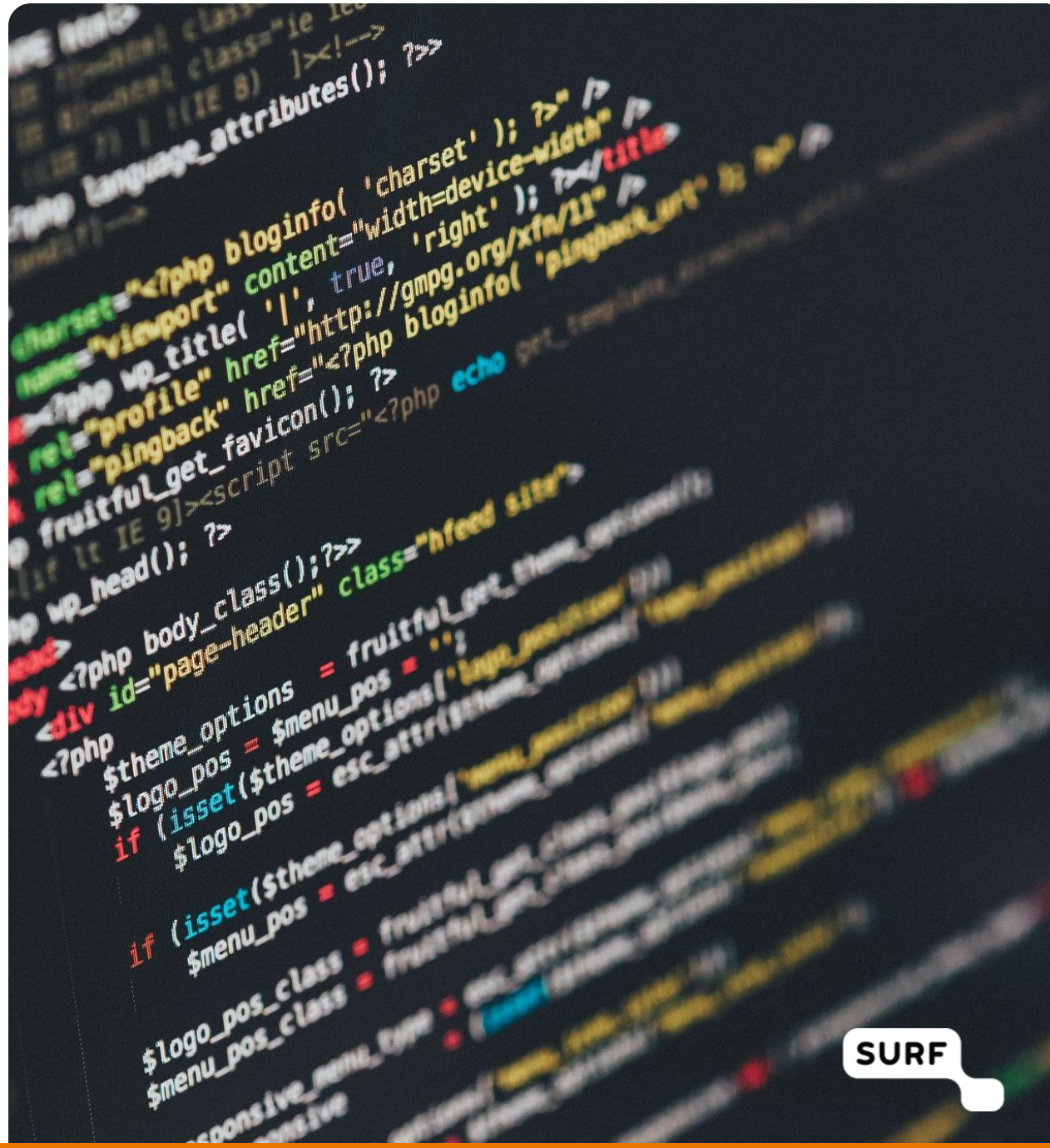
The OSSC structure



Using the OSSC

- CBS project
  - Login to the system is exclusively through CBS Resource Access portal
- ODISSEI and SURF members
  - available for use by researchers affiliated with one of the ODISSEI or SURF member organisations
- Allocation on SURF systems
  - NWO or Pilot Project
  - Direct contract

More info how to apply on the ODISSEI website.



Support for OSSC users

- **HPC support at SURF**
  - Dedicated Helpdesk (servicedesk.surf.nl)
  - Environment creation and management
  - Software installation and configuration
  - Promising Applications support

- **SoDa Team**
  - Support social scientists with data intensive & computational research
  - Consultation for Social Data science projects
  - https://odissei-soda.nl/

OSSC Use cases

Erasmus School of Economics

Utrecht University

University Medical Center Groningen

Erasmus University Rotterdam

VU VRIJE UNIVERSITEIT AMSTERDAM

Maastricht University

National Institute for Public Health and the Environment
Ministry of Health, Welfare and Sport

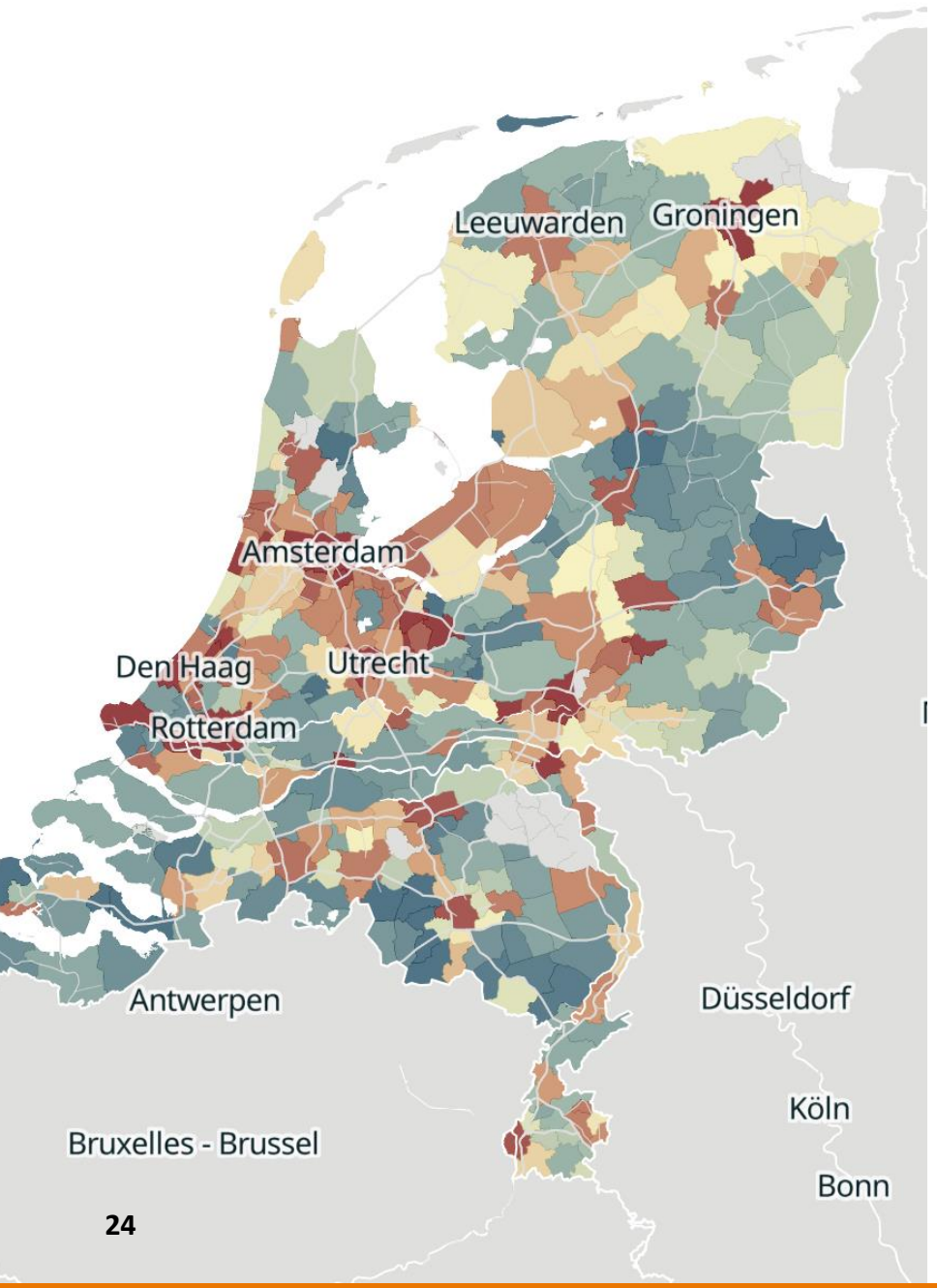
Centraal Planbureau

TILBURG UNIVERSITY

Amsterdam UMC

TU Delft

SURF



OSSC Use cases

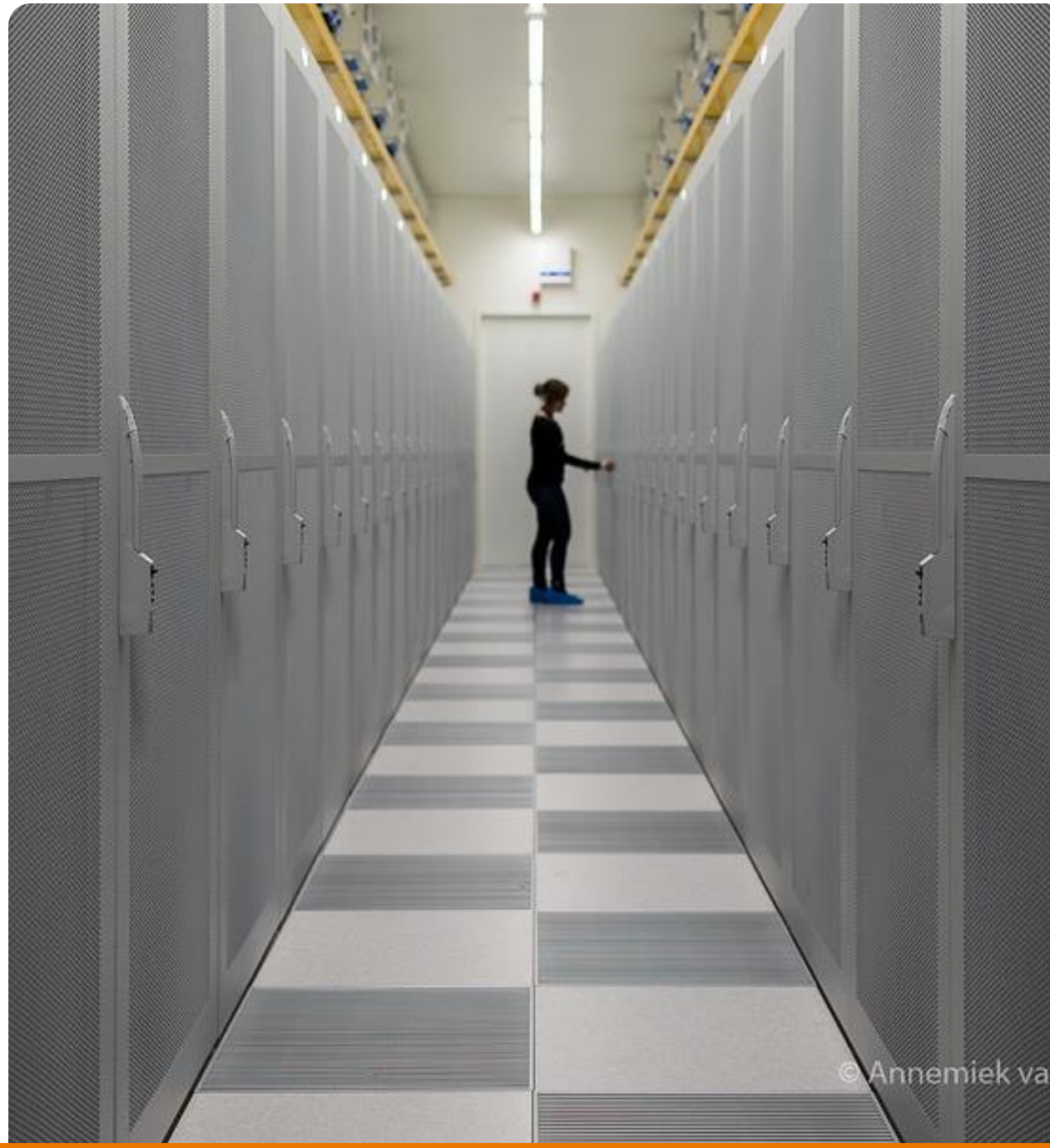
- [Kansenkaart project - https://kansenkaart.nl/](https://kansenkaart.nl/)

  Explore differences in opportunity, such as income, health, or education, using microdata from the central bureau of statistics (CBS).

  - Using R to perform ~150M regressions using CBS data
  - Single analysis run in parallel on multiple cores
  - Executed 100s of jobs each performing analysis of chunk of initial data
  - Distributed models over 50 Snellius compute nodes and managed load balance

Outline

- **Working with a Supercomputer**
  - What is a Supercomputer?
  - What are the differences for the OSSC?
- **Introduction to Linux**
  - Usage and basic commands
  - Shell script programming
- **Running jobs on the HPC system**
  - Interact with the batch scheduler
  - Run a “real” scientific workflow

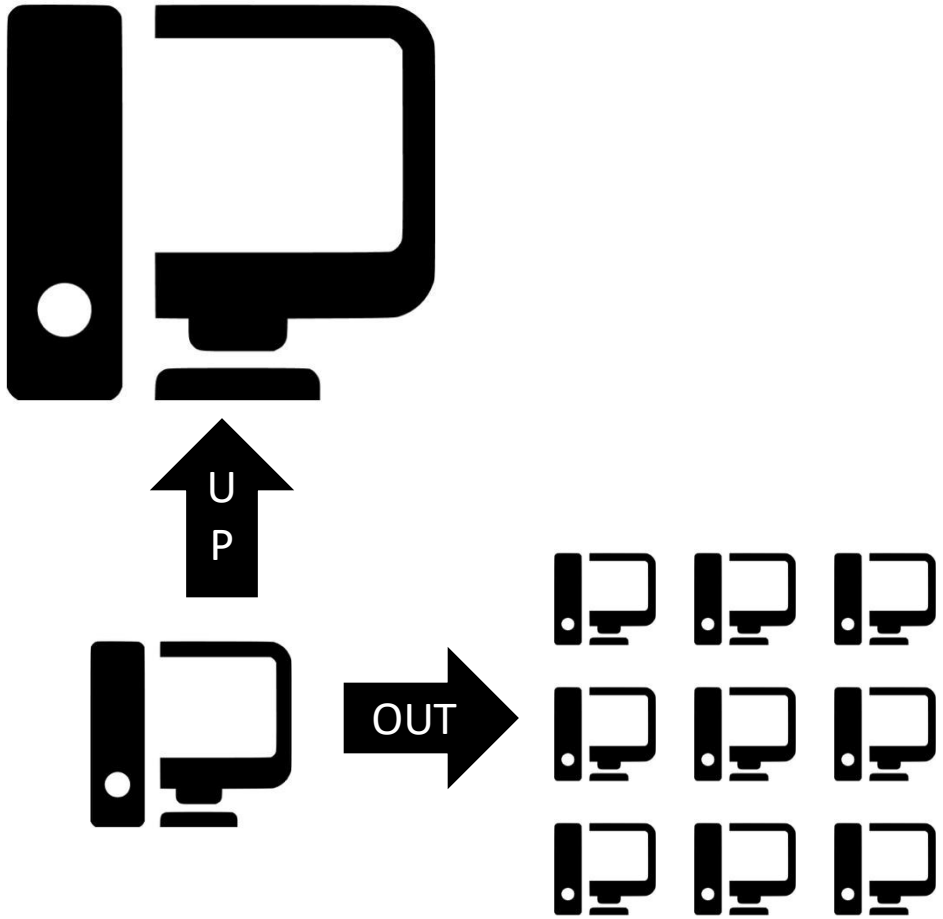


WORKING WITH A SUPERCOMPUTER

- What is a Supercomputer?
- OSSC differences

Why, or rather when, you need a Supercomputer?

- **Scale up**
  - Faster CPUs / GPUs
  - Large memories
  - Specialized Hardware/Software
- **Scale out**
  - Large parallel applications
  - Many small- to medium- size jobs



Working with a Supercomputer

Serial computing

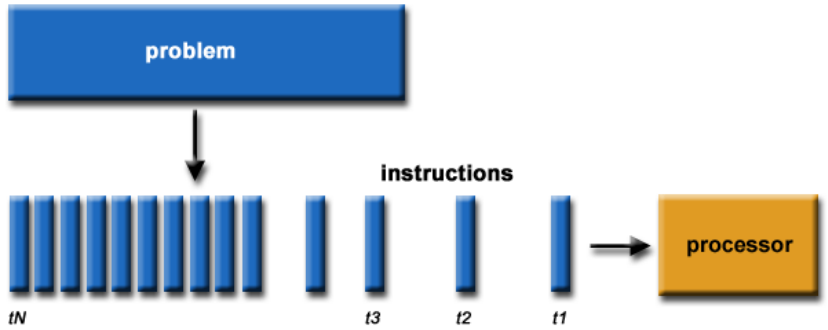
- A problem is broken into a discrete series of instructions, which are executed sequentially on a single processor.

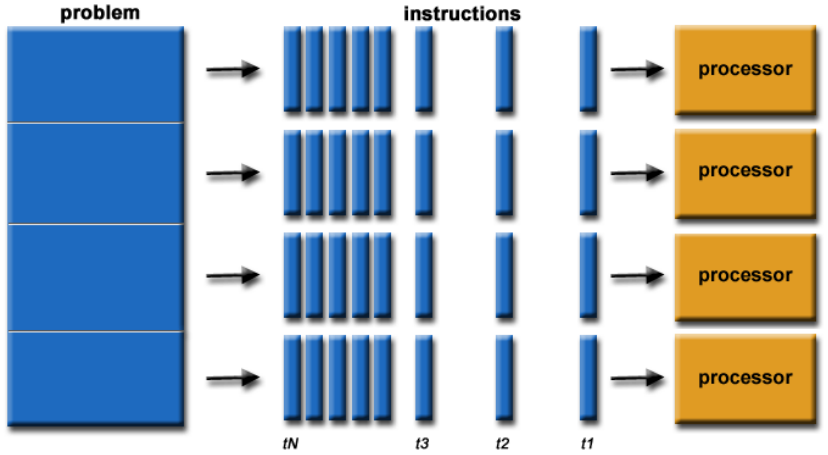
Distributed computing

- A problem is broken into discrete parts that can be solved concurrently using simultaneously multiple resources.

Parallel computing

- A problem is broken into discrete parts that are processed in parallel on multiple resources. Each process is in communication and share data with the others, allowing for multiple processes to work on the same problem.



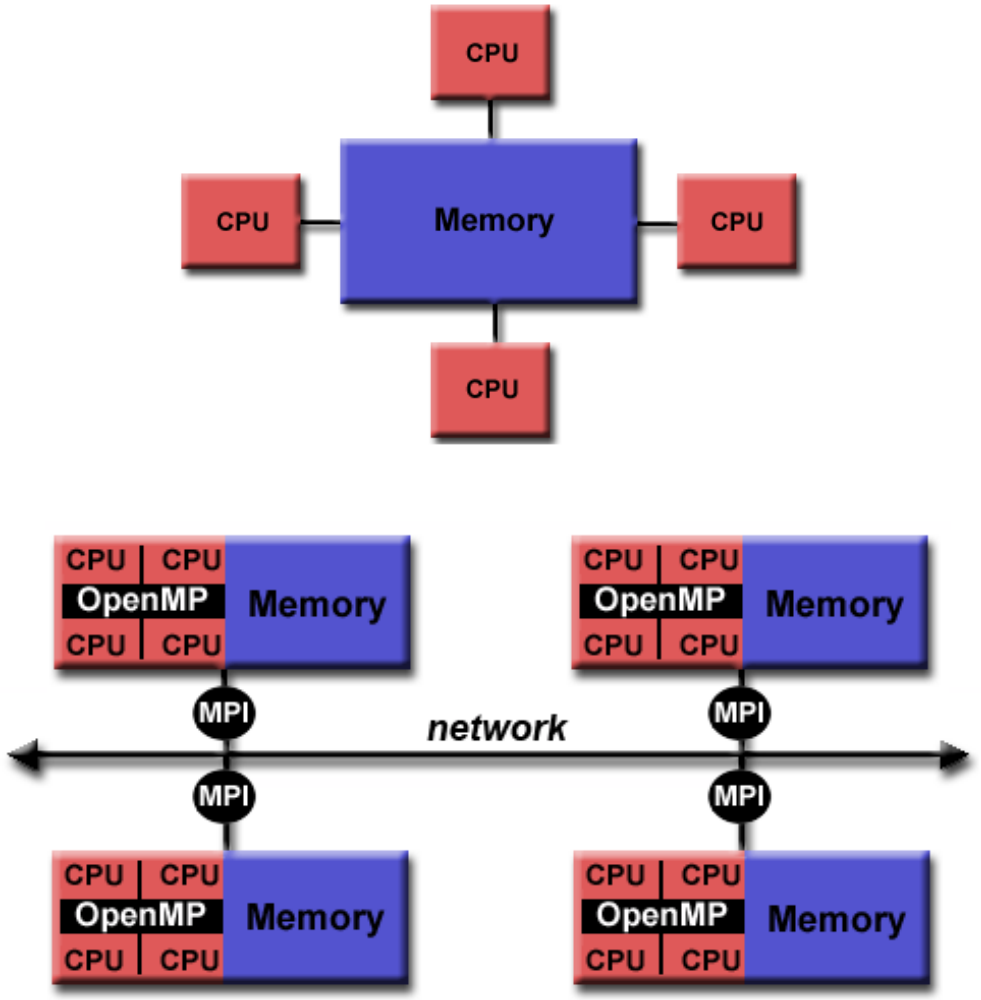


credits: https://computing.llnl.gov/tutorials/parallel_com

Working with a Supercomputer

Parallel computing

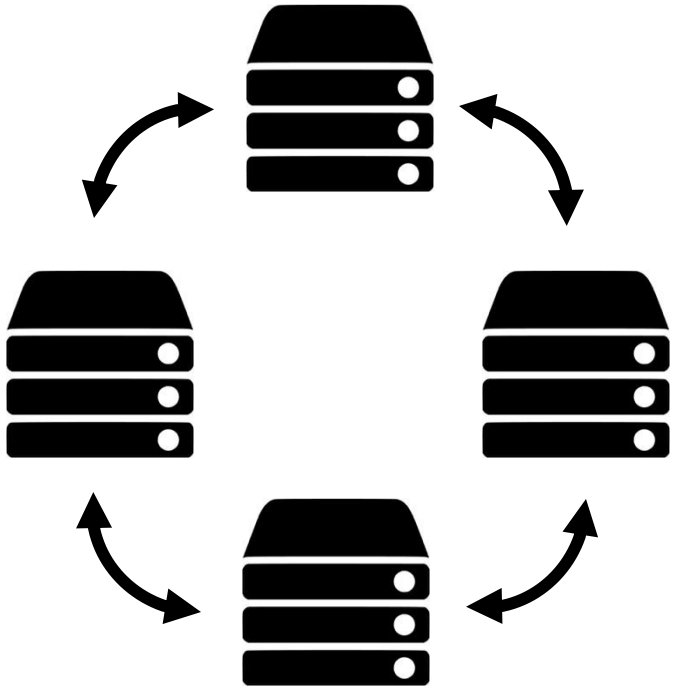
- Task parallel
  - many independent runs
  - needs orchestration
  - for monte-carlo, parameter sweeps
- Shared memory
  - always within one batch node
  - uses threads
  - often implicit
- Distributed memory
  - can use one or more batch nodes
  - uses separate processes
  - almost always using MPI
  - for PDE problems, time stepping



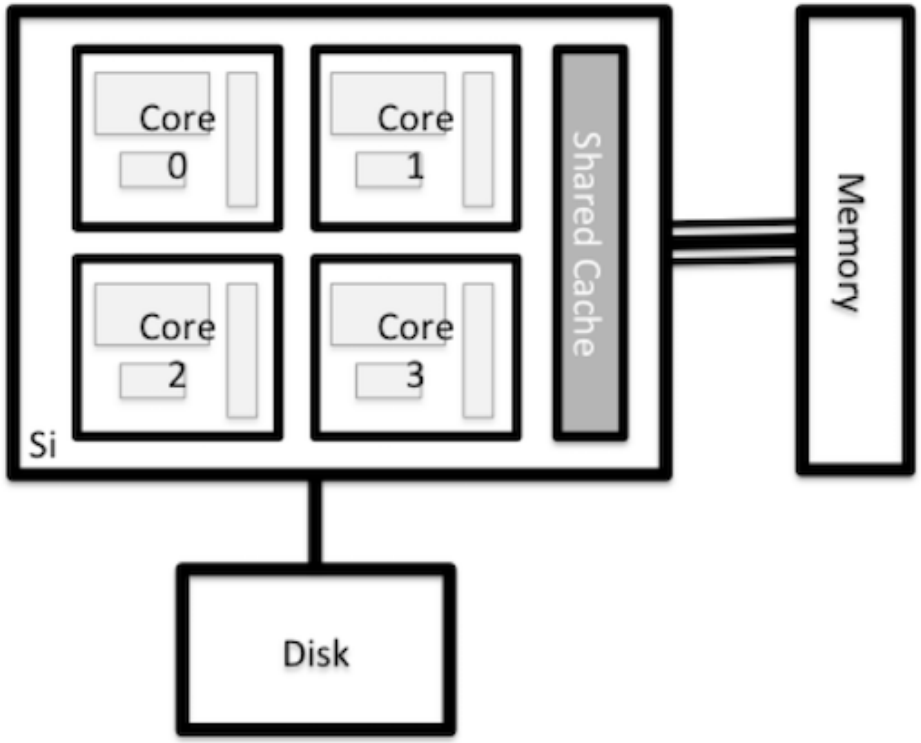
credits: https://computing.llnl.gov/tutorials/parallel_com

Working with a Supercomputer

The compute nodes



Interconnected
Compute Nodes



Source: https://epcced.github.io/hpc-intro

Snellius, the Dutch National supercomputer

AMD Lenovo NVIDIA

580 + 21
AMD ROME CPU compute nodes

36 + 36
Nvidia A100 GPU compute nodes

76,8K + 2.7K
Cores

144 + 144
A100 GPUs

3PF +0.15PF
CPU Rpeak

2,8PF + 2,8PF
GPU Rpeak

Phase 1+1A

NL-SAS, NVMe and SSD
storage

>13.3PiB + 2PiB
Usable storage capacity

>300GiB/s
Sequential perfromance

Storage

714 + 72
AMD NG CPU compute nodes

137K + 14K
Cores

5,63PF + 0,5PF
CPU Rpeak

Phase 2+2A

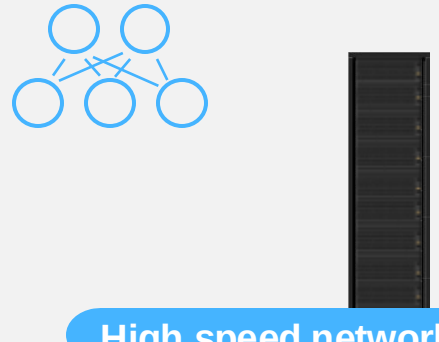
66 + 28
Nvidia Hopper GPU compute nodes

264 + 112
GPUs

20PF
GPU Rpeak

Phase 3

HDR and NDR
Fat-Tree Mellanox infiniband fabric



High speed network

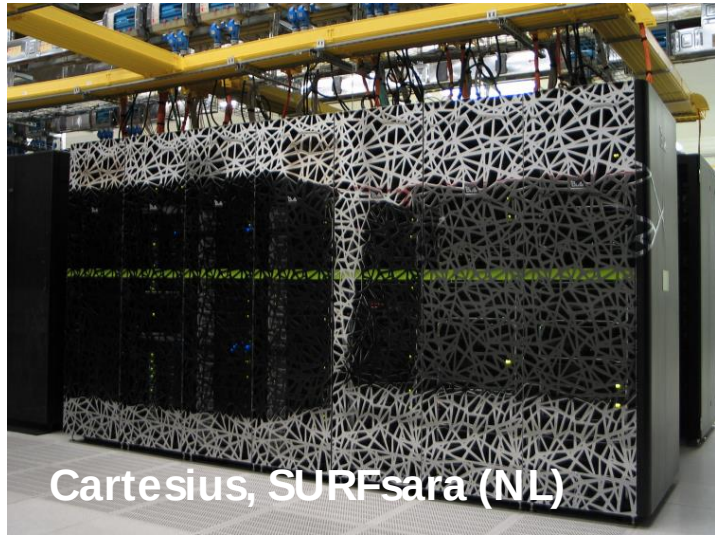
SURF

Working with a Supercomputer

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,699,904	1,206.00	1,714.81	22,786
2	**Aurora** - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel DOE/SC/Argonne National Laboratory United States	9,264,128	1,012.00	1,980.01	38,698
3	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	
4	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442.01	537.21	29,899
5	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	2,752,704	379.70	531.51	7,107
6	**Alps** - HPE Cray EX254n, NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11, HPE Swiss National Supercomputing Centre (CSCS) Switzerland	1,305,600	270.00	353.75	5,194
7	**Leonardo** - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, EVIDEN EuroHPC/CINECA Italy	1,824,768	241.20	306.31	7,494
8	**MareNostrum 5 ACC** - BullSequana XH3000, Xeon Platinum 8460Y+ 32C 2.3GHz, NVIDIA H100 64GB, Infiniband NDR, EVIDEN EuroHPC/BSC Spain	663,040	175.30	249.44	4,159

www.top500.org

Working with a Supercomputer



User Experience

- Multiuser system
- Unix OS
- Optimized software

Compute power

- Many CPUs system
- Specialized Hardware
- Low-latency/High bandwidth Connections

Storage

- Efficient I/O
- Large Memories

Working with a Supercomputer

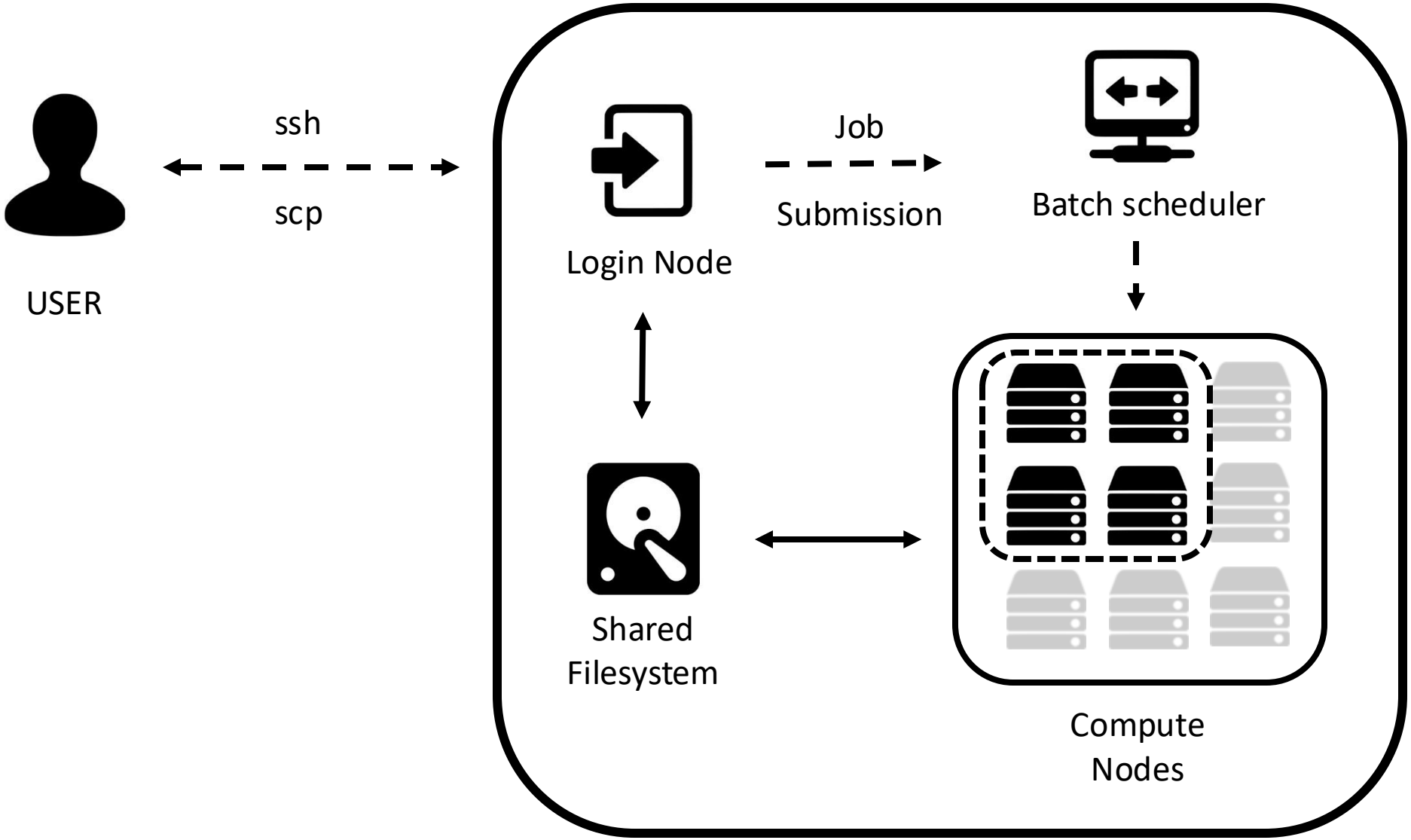
Is NOT like this….

Working with a Supercomputer

Or like this….

Working with a Supercomputer

Working with a Supercomputer



Working with a Supercomputer

Login node(s)

- Editing and transferring files
- Compile programs
- Prepare simulations

Compute nodes

- Multicore nodes
- Large memories
- High-speed interconnections

Batch scheduler

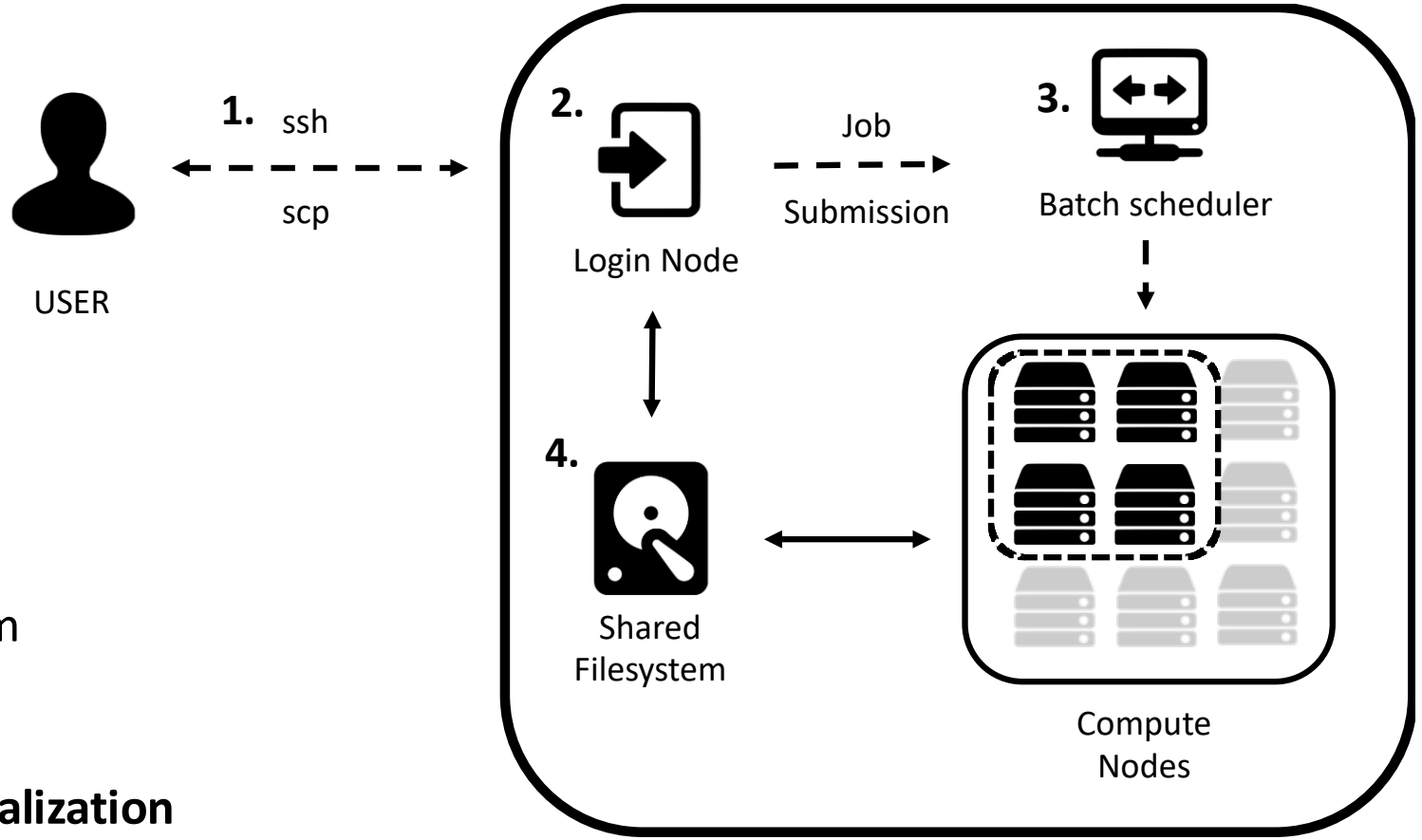
- Resource allocation
- Job queueing
- Accounting and

File system

- Parallel FS
- Efficient I/O
- Node local disks

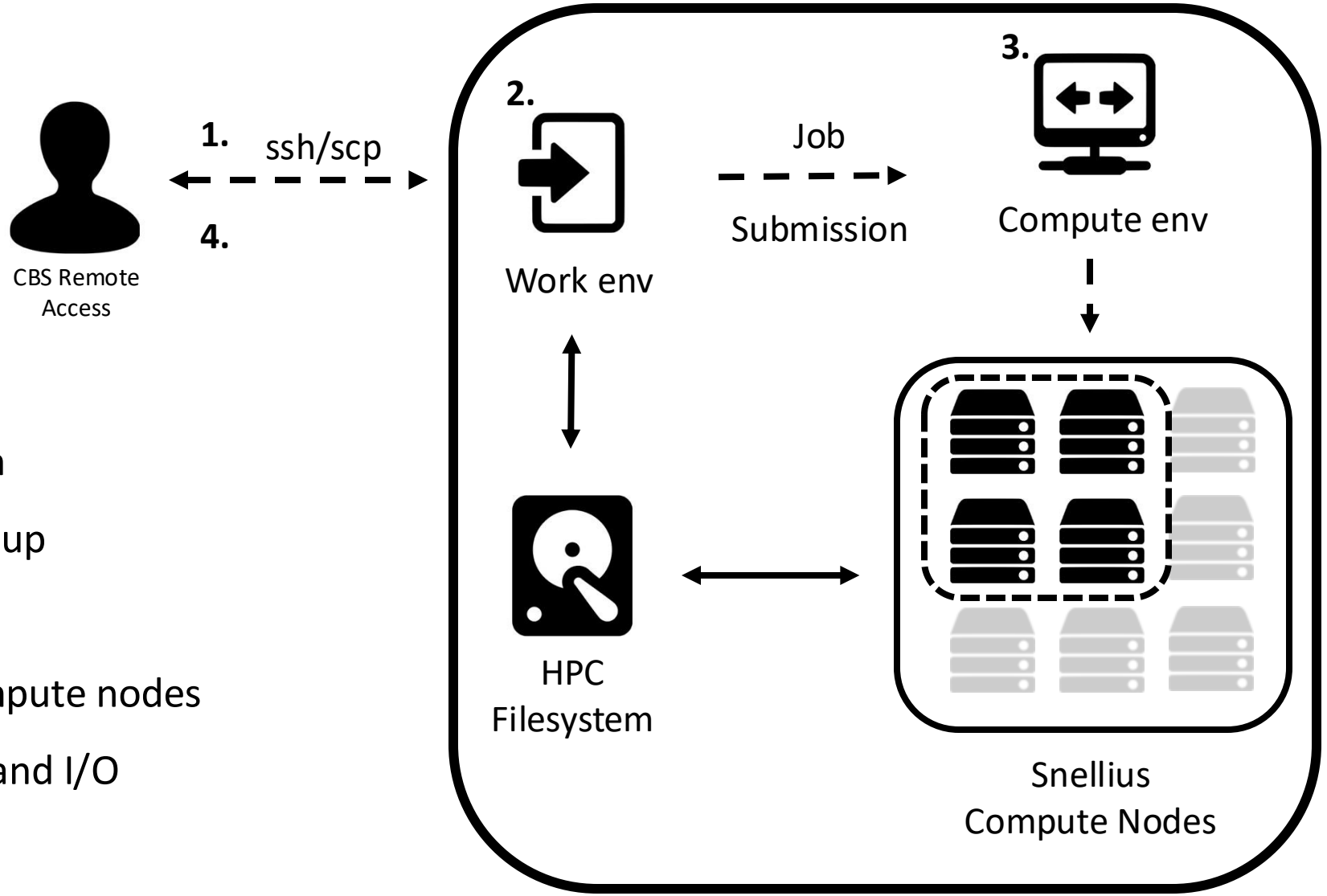
Working with a Supercomputer

1. **Login and transfer files**
   - ssh, scp/ftp
   - Command line, GUI
2. **Prepare your job(s)**
   - Input/Software preparation
   - Job submission script
3. **Submit your job(s)**
   - Submit job to the batch system
   - Monitor job
4. **Retrieve outputs / Remote visualization**



Working with the OSSC

1. **CBS secure connection**
   - Copy files in/out
   - Linked accounts
2. **Prepare your job(s)**
   - Input/Software preparation
   - Job submission test and setup
3. **Submit your job(s)**
   - Provision of virtualized compute nodes
   - High performance storage and I/O
4. **Retrieve outputs to CBS RA**

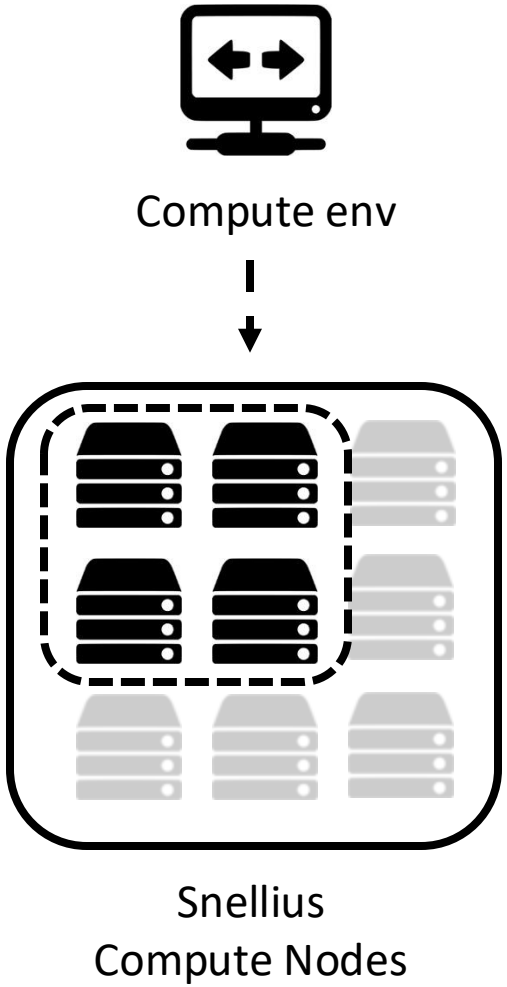


Working with the OSSC

Snellius vs OSSC

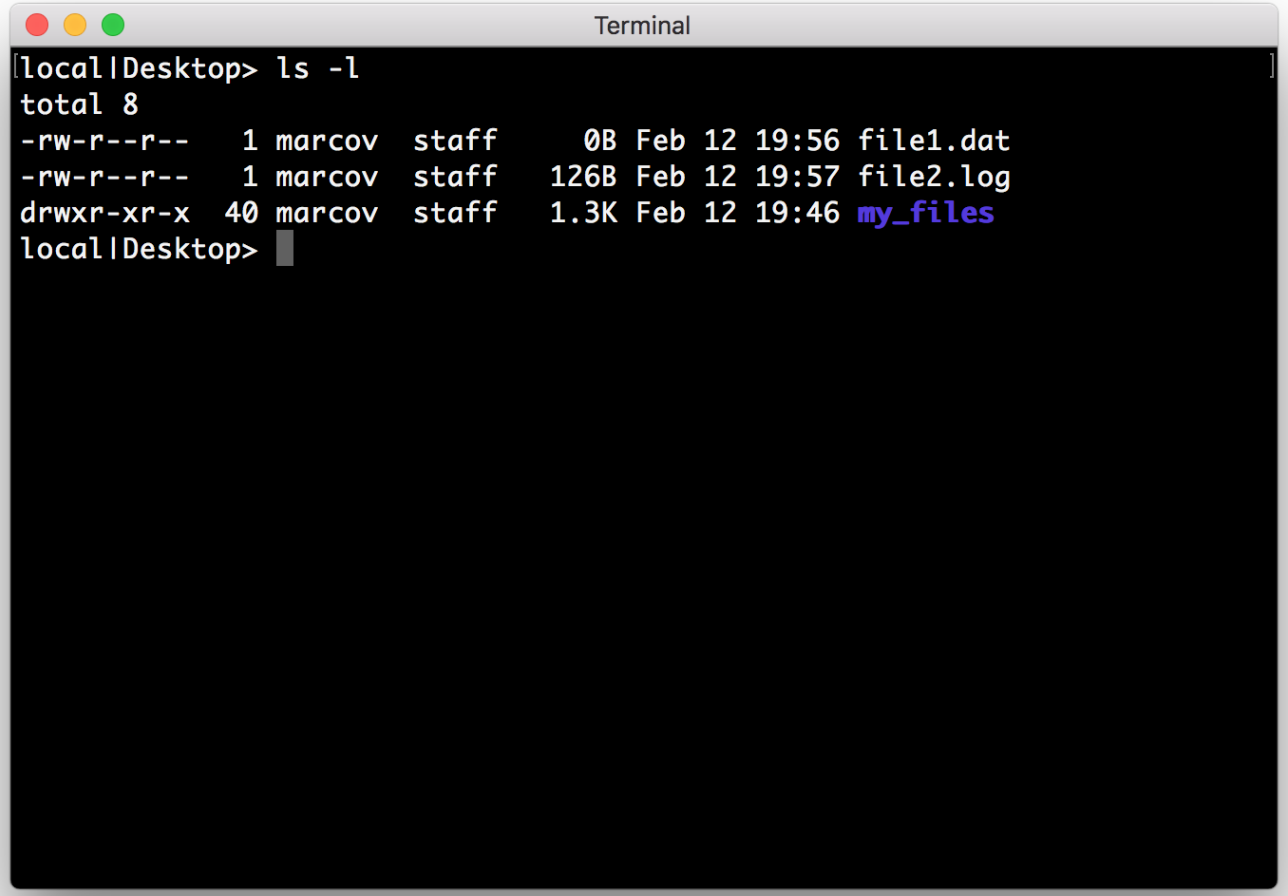
- Same Compute nodes
- Snellius optimized software available
- Same network and file system

- Compute env available on demand
- Work env running on dedicated compute nodes (cost budget!)
- No outside connections
- Access and data transfer only through OSSC RA



Working with a Supercomputer

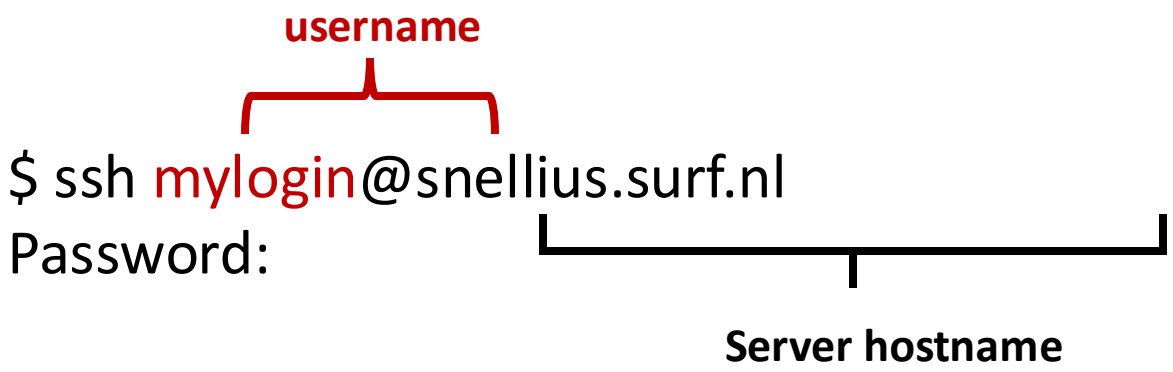
The Terminal and the command line



Working with a Supercomputer

SSH, or Secure SHell

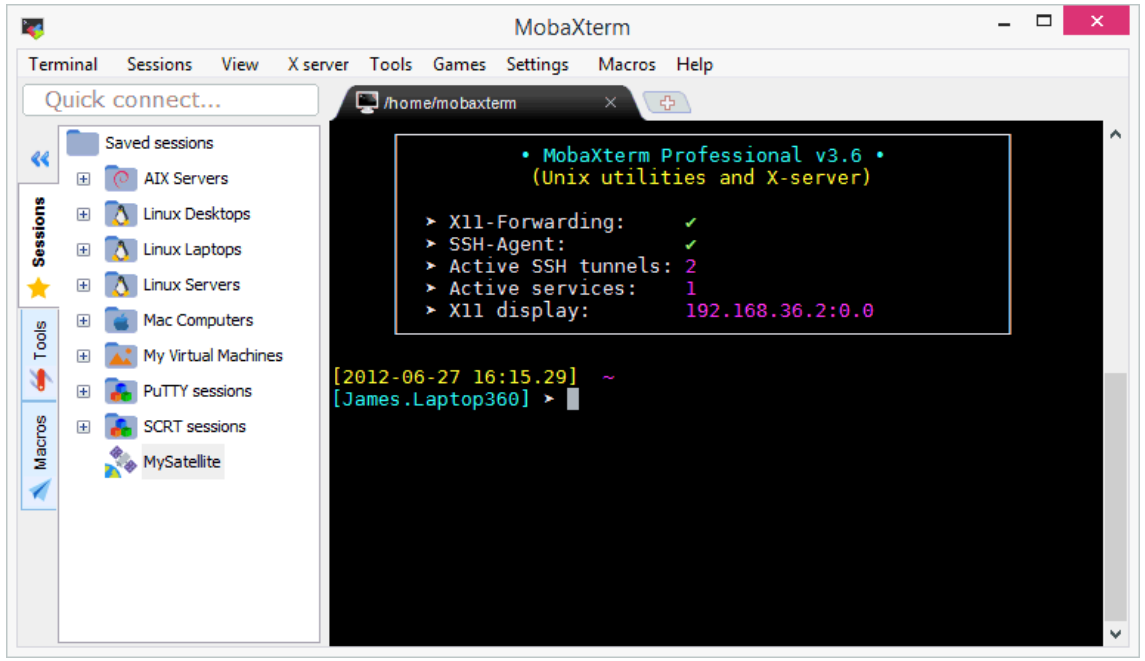
- establishing a cryptographically secured connection
- authenticating each side to the other
- passing commands and output back and forth



Working with a Supercomputer

Install UNIX tools on your local machine

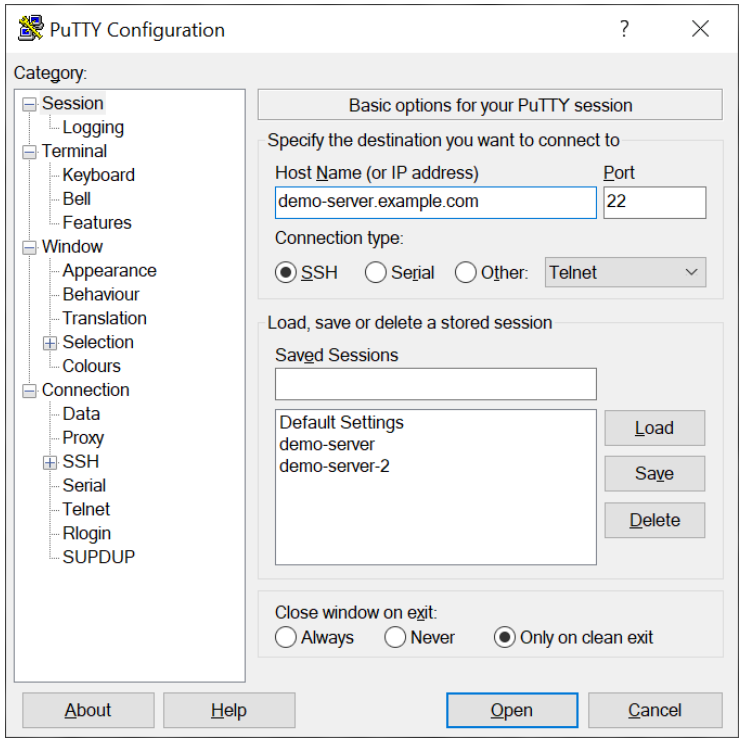
- Windows
  - Putty (https://www.putty.org/)
  - MobaXterm (http://mobaxterm.mobatek.net)
- Mac OSX
  - Terminal (pre-installed)
  - XQuartz (http://www.xquartz.org)
- Linux
  - You are already well equipped!

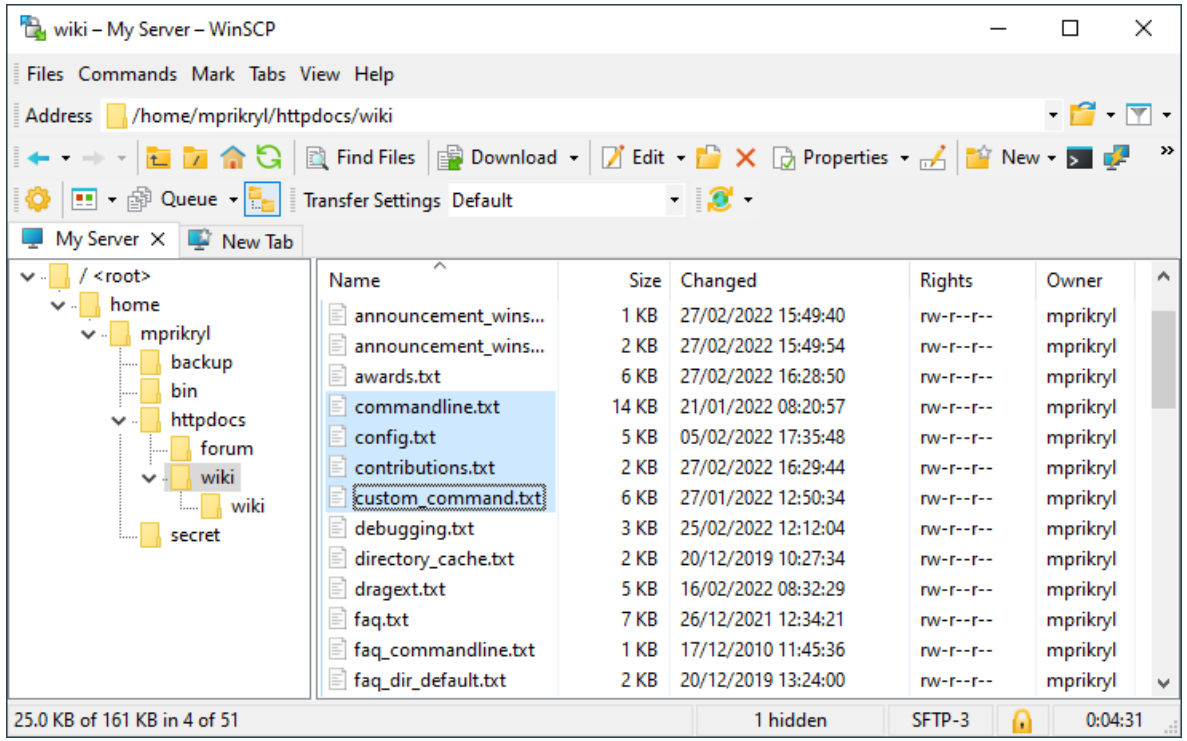


MobaXterm interface

Working with a Supercomputer

Connecting and moving data into the OSSC





Preconfigured Putty and WinSCP on the CBS RA

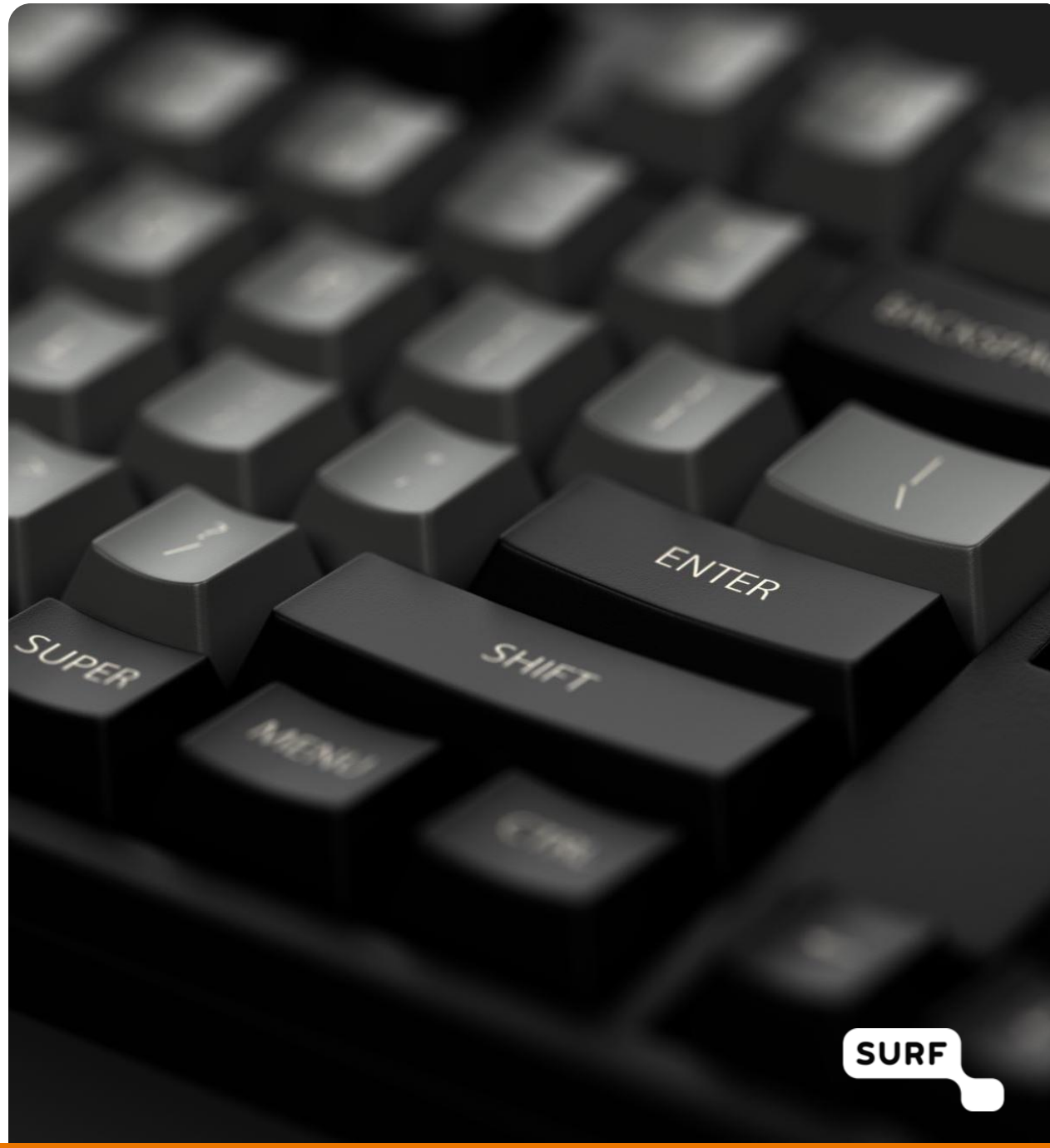
Working with a Supercomputer

You are logged in!!!

Working with a Supercomputer

What to you need to start using an HPC system

- Motivations (and patience!)
- Account and login on the system
- Budget to use the resources
  - Each account ”holds” a budget that can be spent by all users in the account
  - Use of the resource is credited depending on the type of hardware



Working with a Supercomputer

Get access to the system

- EU funded projects (EuroCC, CompBioMed, etc.)
- National initiatives
  - NWO, Computing Time on National Computer Facilities
  - Large/Small Compute Applications
- Special agreements with Universities or research centers
- Contact the helpdesk at SURF: **servicedesk.surf.nl**

INTRODUCTION TO LINUX

- Linux basic commands
- Shell script programming

Working with a Supercomputer

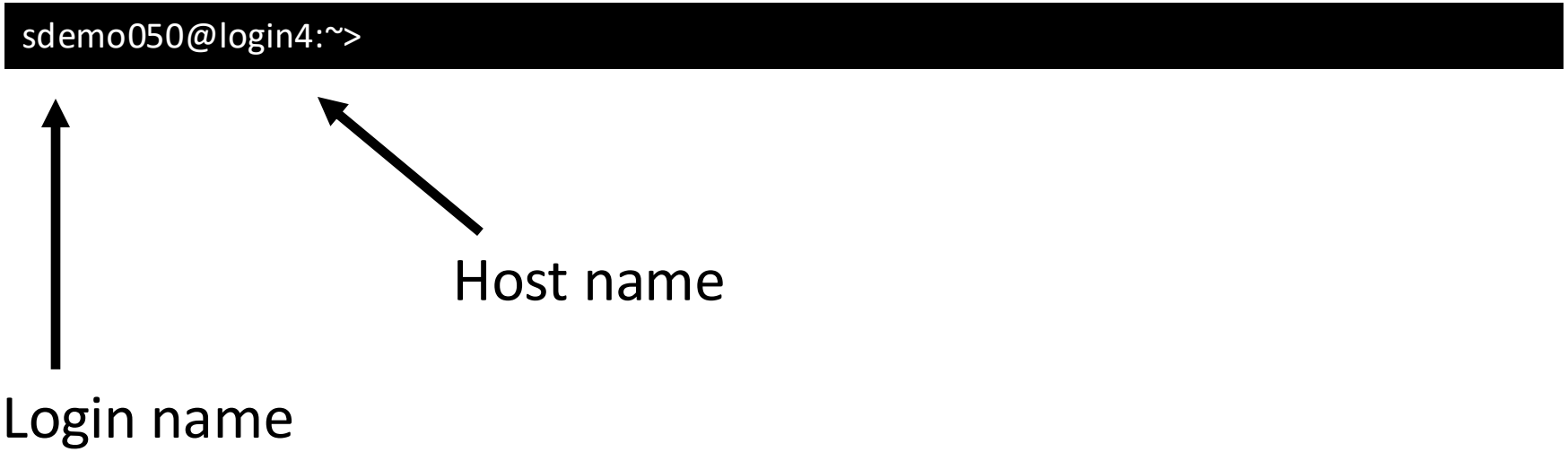
What is Linux?

- Operating System
  - Program that controls all other parts of a computer system
  - Allocates computer's resources and schedules tasks
  - Allows the user to use the facilities provided by the system
  - Essential to all computer systems
- Multi-User and Multi-Tasking
  - Multiple users have multiple tasks running simultaneously
- Designed to be machine independent
- Setup as a software development environment
- Suitable for scientific applications

Introduction to Linux

Linux basic commands

- After successful login we will interact with the shell program

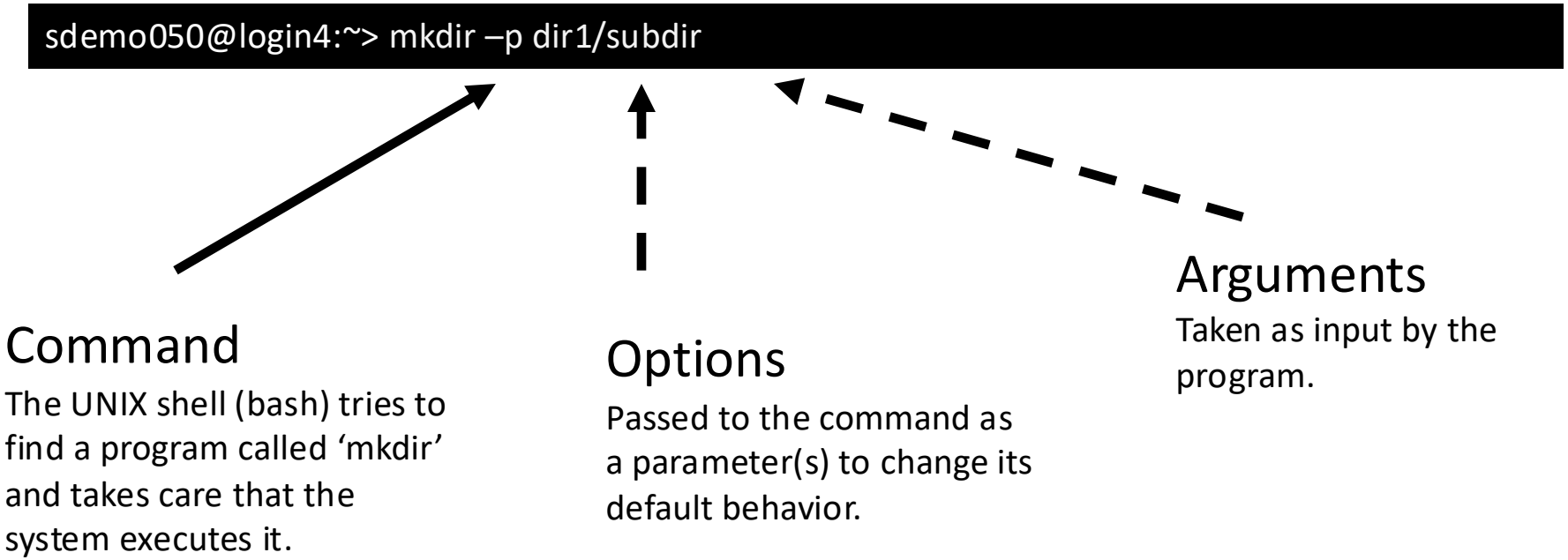


- Now the system is ready to accept commands

Introduction to Linux

Linux basic commands

- Structure of a Linux command



- Case-sensitive (everything!)
- Spaces used to separate command, options and arguments

Introduction to Linux

- **Explore the file system**
  - ls, cd, pwd
- **Create and edit files and directories**
  - mkdir, touch, cp, mv, nano, less, vi
- **Use variable (env. variables)**
  - $USER, $HOME
- **System tools**
  - history, find, zip, tar

```
sdemo050@login4:~> cd /mydata/

sdemo050@login4:~> ls
bin file1.txt file2.log

sdemo050@login4:~> nano README.md

sdemo050@login4:~> echo $USER

sdemo050@login4:~> history
```

Introduction to Linux

Linux basic commands

File editing/viewing

- touch: Creates a blank file with a specified name.
- less: View contents of specified file, page by page.
- cat: Display contents of a file.
- head/tail: Displays the first/last 10 lines of a file.
- ls: List files and directories.
- …

System tools

- history: Display a listing of the last commands you've run.
- find: Search files and directories.
- tar: Compress and extract files.
- top: Display processes running on the system.
- echo: Print a message at screen.
- …

Shell script programming

What if I want to run many bash commands?

…maybe in a workflow?

Bash scripts

- The “Shell” is the program which read commands and run other programs. Bash is a type of “Shell”.
- A *bash script* is a plain text file which contains a series of commands
- Any command you can run on the command line can be put into a script (v.v.)
- It will be executed like a normal program: ./script.sh

Introduction to Linux

- **Shell script example**

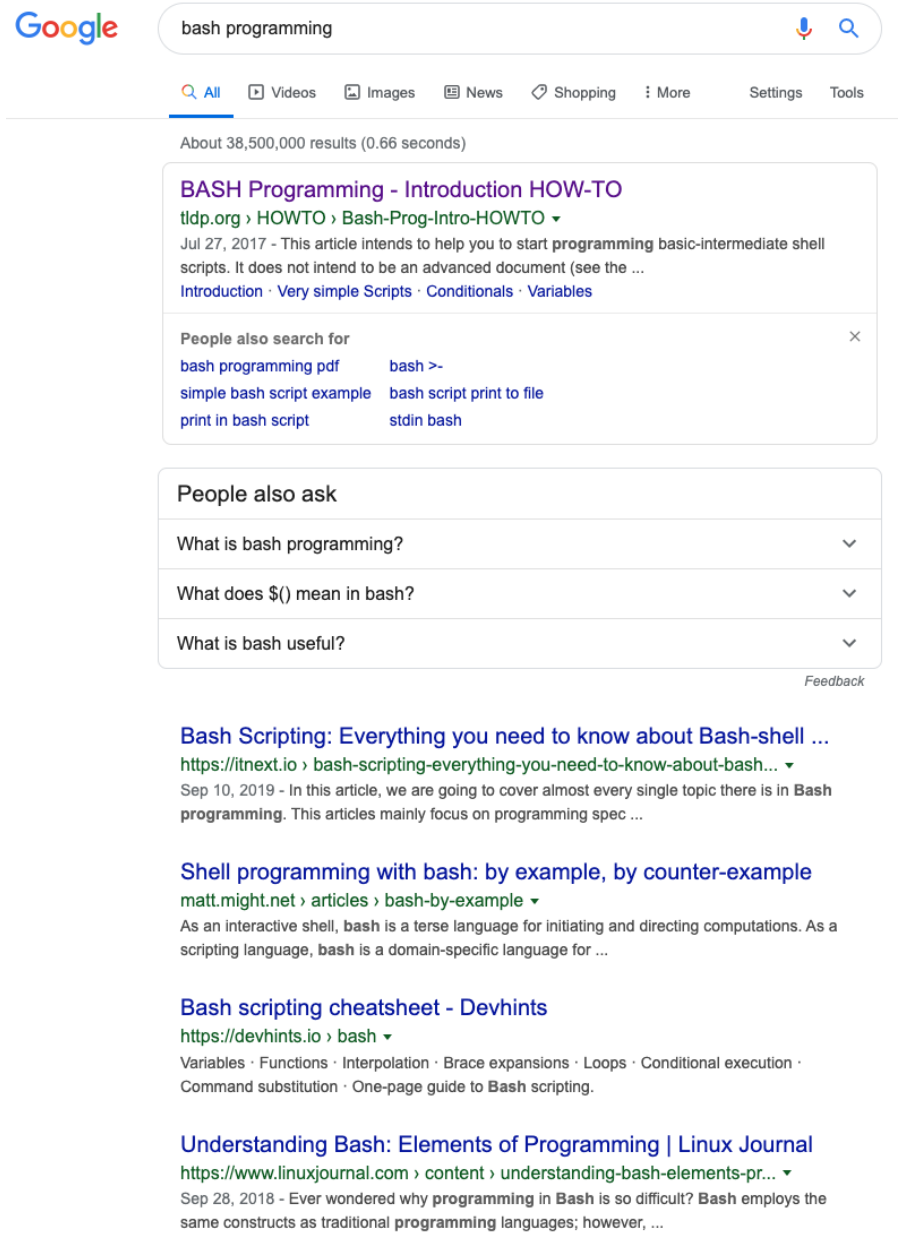
  - **echo** : prints text to screen
  - **lscpu**: list all cpus available on the node
  - **sleep**: put the cpu to sleep for N seconds
  - **date**: print current time and date

```
#!/bin/bash

echo "Running lscpu command."
echo "This is what it does:"
lscpu --help
echo

echo "Printing results in cpu.log..."
lscpu > cpu.log
sleep 1
date >> cpu.log
echo

echo "Done"
```

Introduction to Linux

Shell script programming

- Loops for, while, until
- Conditional statements: If, else, then
- Functions
- Variables
- and much more



Shell script programming

Standard input, output and error

Every program (bc, shell, ...) has three predefined input/output associated:

- Standard input (stdin): normally your keyboard
- Standard output (stdout): normally your screen
- Standard error (stderr): normally your screen

These can be redirected to a file or to another command.

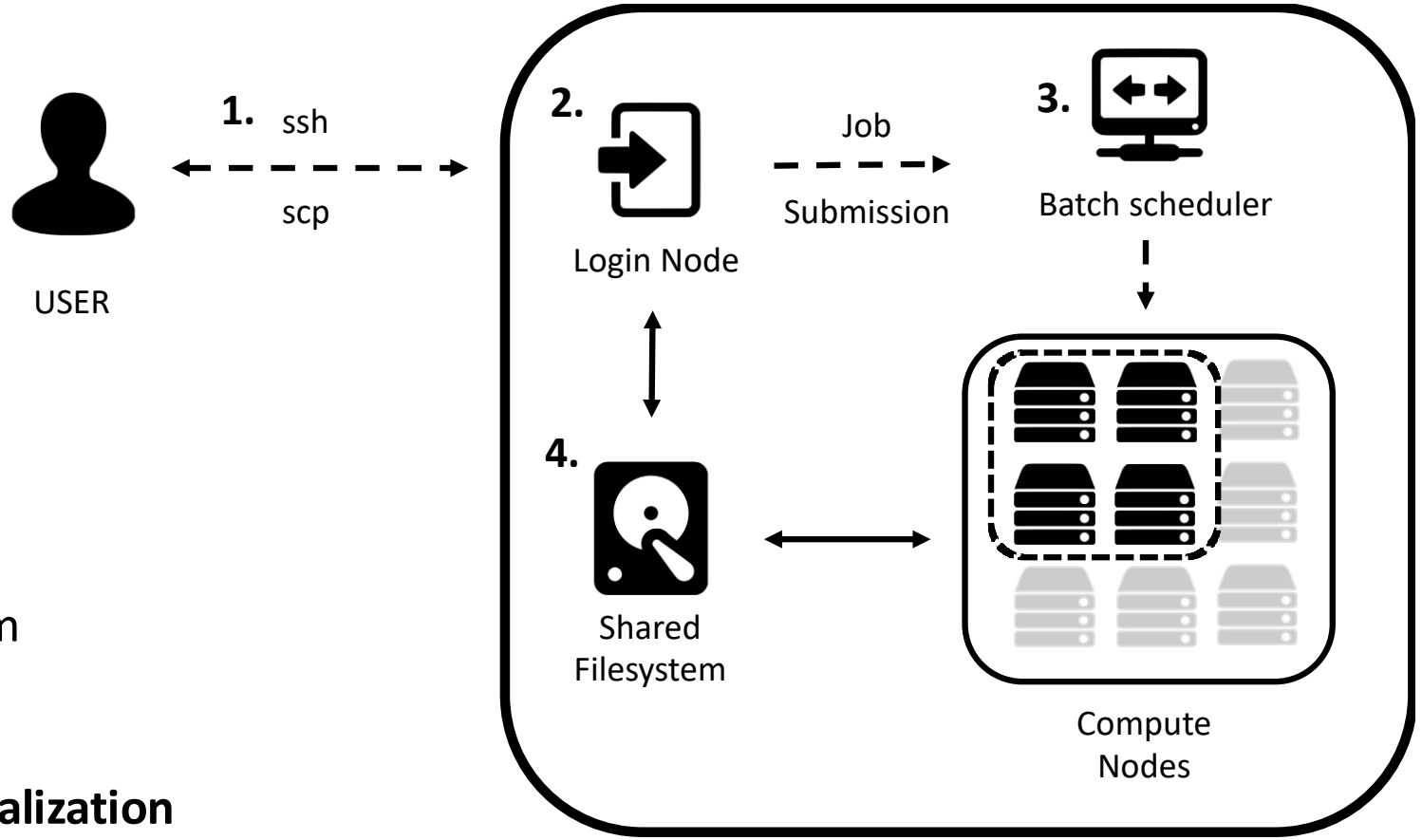
```
sdemo050@login4:~> echo "3/0" | bc > calc.log
```

RUNNING JOBS ON THE HPC SYSTEM

- Interact with the batch scheduler
- Run a scientific application or workflow

Running jobs on the HPC system

1. **Login and transfer files**
   - ssh, scp/ftp
   - Command line, GUI
2. **Prepare your job(s)**
   - Input/Software preparation
   - Job submission script
3. **Submit your job(s)**
   - Submit job to the batch system
   - Monitor job
4. **Retrieve outputs / Remote visualization**



Running jobs on the HPC system

The batch jobs scheduler

- Supercomputers use job schedulers to distribute computational tasks over the available nodes.

- Instead of executing commands interactively, you prepare a job script
  - Script containing the commands to execute
  - Resource characteristics (specific)

- The batch system is responsible for allocating cores, processors or nodes to a job.

Running jobs on the HPC system

The batch jobs scheduler

- It allows to run MANY jobs at the same time
    - The system takes care that they are run efficiently on the available resources.
- Multiusers, queue system
    - A batch system allows users to always submit jobs, even if a lot of people are using the system at the same time. In addition take care of budgeting and fair resource usage.
- System load balance
    - The system takes care of balancing the load across nodes and during time. In a batch system, most jobs may be submitted during office hours, but the scheduler will continue to start jobs at night as nodes become available.

The batch jobs scheduler

- Submit job to the queue:

```
sdemo050@login4:~> sbatch <job script>
```

- Show running and queued jobs:

```
sdemo050@login4:~> squeue –u $USER
```

- Remove a job from the queue or kill it if running:

```
sdemo050@login4:~> scancel –u $USER
```

Running jobs on the HPC system

The batch jobs scheduler

- Create multiple jobs:

```
Submit a job array with index values between 0 and 31
$ sbatch --array=0-31    -N1 tmp

# Submit a job array with index values of 1, 3, 5 and 7
$ sbatch --array=1,3,5,7 -N1 tmp

# Submit a job array with index values between 1 and 7
# with a step size of 2 (i.e. 1, 3, 5 and 7)
$ sbatch --array=1-7:2   -N1 tmp
```

- Job arrays will have additional environment variables set (e.g. $SLURM_ARRAY_TASK_ID)

Running jobs on the HPC system

- Batch schedulers distribute work to the *compute nodes*

- Workflow
  - **You** upload your data from your computer to the cluster system
  - **You** create a job script with the work steps
  - **You** submit the job script to the scheduler
  - **The scheduler** looks for available computers to run your work
  - When a batch node with the requirements you specified becomes available, your work runs
  - When the job is finished, **you** download the results to your computer

- Batch scheduler on Snellius: SLURM (http://slurm.schedmd.com)

Running jobs on the HPC system

SLURM job scripts

SLURM Directives

A job script is a shell script which contain directives to inform the batch system about the characteristics of the resources required to run the job.

This directives appear as comments (#SBATCH) in the job script and have to conform with the sbatch syntax.

See: https://slurm.schedmd.com/sbatch.html

```
#SBATCH --nodes=<num>
#SBATCH --ntasks=<num>
#SBATCH --time=DD-HH:MM:SS
#SBATCH --partition=<name>
#SBATCH --task=<num>
#SBATCH --output=<file>
#SBATCH --tasks-per-node=<num>
#SBATCH --cpus-per-task=<num>
#SBATCH --core-per-cpu=<num>
```

Running jobs on the HPC system

Software stack

Some software packages require certain settings in your user environment, like paths and environment variables.

- Programming and scripting languages: Python, R, Matlab
- Compilers for C, C++, Fortran
- Specialized libraries (MKL, OpenMPI)
- Scientific codes
- Tools etc.

Running jobs on the HPC system

Software stack

Snellius uses **module**, a user interface to the Modules package which provides for the dynamic and centrally supported software environment.

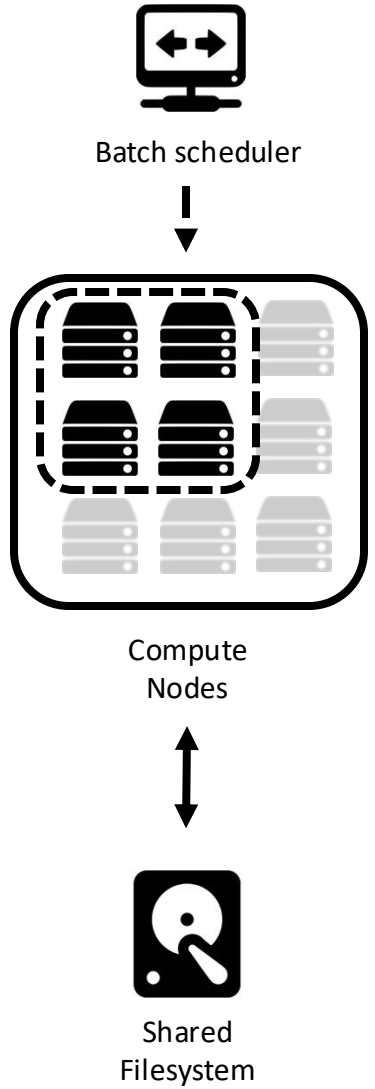
module avail	list modules
module avail Python	list all installed versions of python
module load Python	load the default python version
module load Python/3.11.3	load a specific version of python
module unload Python	unload python
module list	list currently loaded modules

Running jobs on the HPC system

File system

Each user has several areas of disk space for storing files. These areas may have size or time limits.

Choose carefully where to store your data!

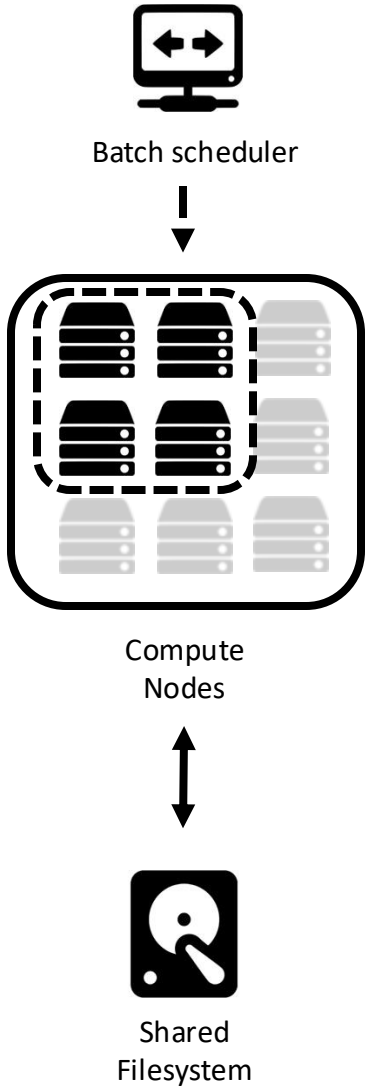


- **/home/user**
  - User home directory (quota - currently 200GB)
  - Backed up
  - Meant for storage of important files (sources, scripts, input and output data)
  - Based on NFS: not the fastest file system

Running jobs on the HPC system

File system

- **/scratch** (local & shared on Snellius)
  - Variable quota depending on disk
  - Not backed up
  - Meant for temporary storage (during running of a job and shortly thereafter)
  - Based on GPFS: the fastest file systems on Snellius
- **/project**
  - Large and fast
  - For special projects requiring lots of space (quota – as much as needed/possible)
  - Not backed up
  - Comparable in speed with /scratch on Snellius



Running jobs on the HPC system

- **Job scripts consist of:**
  - the “shebang” line: #!/bin/bash
  - scheduler directives
  - command(s) that load software modules and set the environment
  - command(s) to prepare the input
  - command(s) that run your main task(s)
  - command(s) to save your output

```
#!/bin/bash
#SBATCH --job-name="firsttest"
#SBATCH --nodes=1
#SBATCH --ntasks=10
#SBATCH --time=00:01:00
#SBATCH --partition=rome

module load 2023
module load Python/3.11.3

cp –r <my_folder> $TMPDIR
cd $TMPDIR

srun myexecutable.exe

cp –r $TMPDIR/* $HOME
echo "DONE"
```

Running jobs on the HPC system

Login and transfer files to remote machine

- ssh, scp/ftp
- Command line, GUI

Prepare your jobs

- Input + Software
- Job submission script

Submit your job and retrieve output

- Submit job to the batch system
- Monitor job, retrieve output

Best practices

- Give the scheduler a realistic *walltime* estimate
- Your home directory is slow. Use $TMPDIR.
- Load software modules as part of your job script this improves reproducibility
- Run parallel versions of your programs

https://servicedesk.surf.nl/wiki/display/WIKI/Snellius

Running jobs on the HPC system

- **Snellius specific documentation and guides**
  - https://servicedesk.surf.nl/wiki/display/WIKI/Snellius

- **More courses by SURF (for research and more)**
  - EuroCC Netherlands Agenda
    - https://eurocc-netherlands.nl/calendar/category/training-en/
  - SURF Agenda
    - https://www.surf.nl/en/agenda
  - SURF training mailing list
    - https://lists.surfsara.nl/listinfo/training-announce

The OSSC Team at SURF

- **OSSC team**
  - **Ahmad Hesam** (Project leader)
  - **Michel Scheerman** (OSSC & Snellius developer)
  - **Martijn Kruiten** (OSSC & Snellius developer)
  - **Dennis Stam** (OSSC & Snellius developer)
  - **Annette Langedijk** (OSSC Management board)
  - **Benjamin Czaja** (OSSC & Snellius Advisor)
  - **Stefan Wolfsheimer** (OSSC & Snellius Advisor)
- **Former members**
  - **Narges Zarrabi** (Former Project leader)
  - **Maxime Mogé** (Former OSSC & Cartesius Advisor)
  - **Lykle Voort** (Former OSSC & Cartesius Advisor)

Thank you.