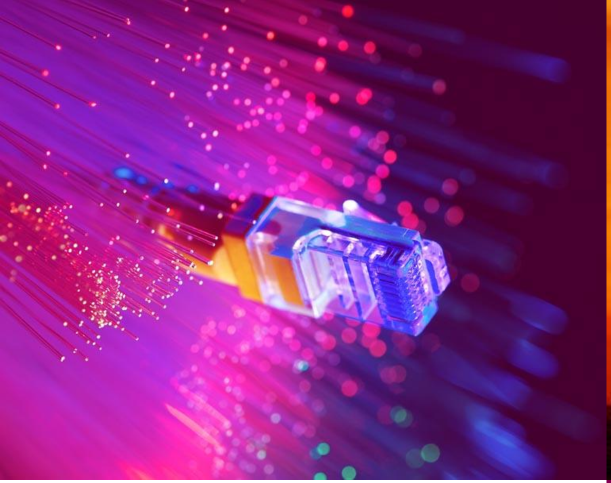


Using CSC Environment Efficiently

30.10.2017

CSC – Finnish expertise in ICT for research, education, culture and public administration

Program

09:00–09:15 Introduction to the course

09:15–09:45 Getting access
User account, project and services, web-based access to CSC's services

09:45–10:00 Coffee break

10:00–11:00 How to connect
How to access CSC's computers, NX client, taito-shell

11:00–12:00 CSC's computing environment
Different platforms, module system, licensing, storage and data transfer

12:00–13:00 Lunch break

13:00–14:30
Running your jobs, resource-management (a.k.a. batch job) systems

14:30–14:45 Coffee break

14:45–15:30 Compiling your program
What's inside a makefile and how to use **`make`** to install a program yourself

15:30–15:45 Science services at CSC
A short introduction

15:45–16:15 Troubleshooter + Installation session: Helping with installation of NX client, PuTTy, Virtual appliance,…

Practicalities

- Keep the name tag visible
- Lunch is served in the same building
  - Room locked during lunch (lobby open, use lockers)
- Toilets are in the lobby
- Network:
  - WIFI: eduroam, Haka authentication
  - Ethernet cables on the tables
  - CSC-Guest accounts
- Username and password for workstations: given on-site

- Bus stops
  - Other side of the street (102,103) ⟶ Kamppi/Center
  - Same side, towards the bridge (194,195/551) ⟶ Center/Pasila
  - Bus stops to arrive at CSC at the same positions, just on opposite sides
- If you came by car: parking is being monitored - ask for a temporary parking permit from the reception (tell which workshop you're participating)
- Visiting outside: doors by the reception desks are open

Non-profit state enterprise with special tasks

Turnover in year 2015

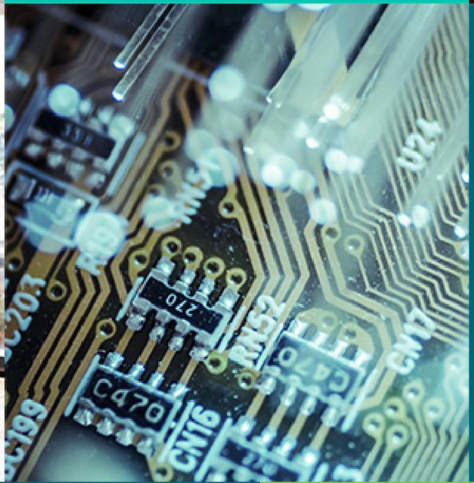
36.8M€

Headquarters in Espoo, datacenter in Kajaani, Finland

Owned by state

(70%)

and all Finnish education higher institutions (30%)



Circa

290

employees in year 2016

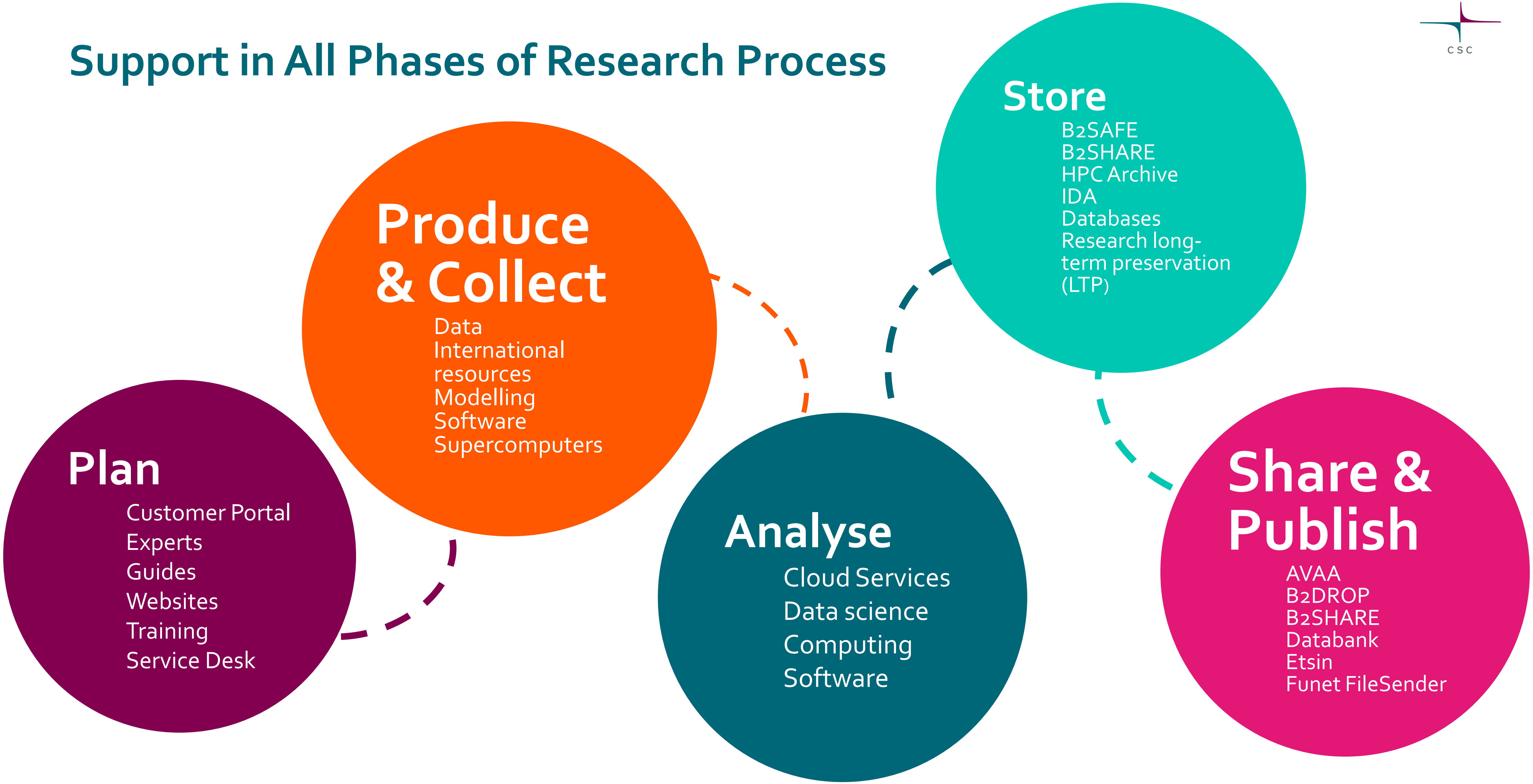
Our Customers

Research institutes and organizations

Organizations providing education

Memory organizations, state and public organizations

Support in All Phases of Research Process



Plan

Customer Portal
Experts
Guides
Websites
Training
Service Desk

Produce & Collect

Data
International resources
Modelling
Software
Supercomputers

Analyse

Cloud Services
Data science
Computing
Software

Store

B2SAFE
B2SHARE
HPC Archive
IDA
Databases
Research long-term preservation (LTP)

Share & Publish

AVAA
B2DROP
B2SHARE
Databank
Etsin
Funet FileSender

Supporting flexible pathways in education

When applying

VIRTA as a source of information (Opintopolku.fi)

Applicant statistics (Vipunen)

Enrolment (OILI)

During studies

VIRTA as a source of information:
allocation of Study Grant (Kela),
qualifications during studies (Valvira),
entitlement to healthcare (YTHS)

Education statistics (Vipunen)

Tailored course platforms in cloud environments (Notebooks), elctronical examination system (EXAM), collaborative platforms (Eduuni), online e-learning videotools (Funet Tiimi, Funet Etuubi Kaltura), register for student and study information (Oodi), sharing large files (Funet FileSender), identity management (Haka-luottamusverkosto, Eduuni-ID), worldwide wireless campus network (eduroam)

Flexible studying

VIRTA as a source of information:
studies in another higher education institution,
exchange studies,
transferring transcript records (JOO-liikkuvuus, Puro, EMREX)

Graduation and life after studies

VIRTA as a source of information:
professional competences (Valvira)

Statistics and monitoring (Vipunen)

Feedback and career surveys (Arvo)

Lifelong learning

Mangement of student's own digital data and consent to pass it on (My Data)

Funet – National and International Networks and Services

FUNET YHTEYDET

csc

Services included in Funet membership

- Funet Network Connections
- Funet CERT Information Security Service
- Vulnerability Scanner
- Certificate Service
- eduroam Roaming Access Service
- Funet FileSender File Sharing Service

Services with additional costs

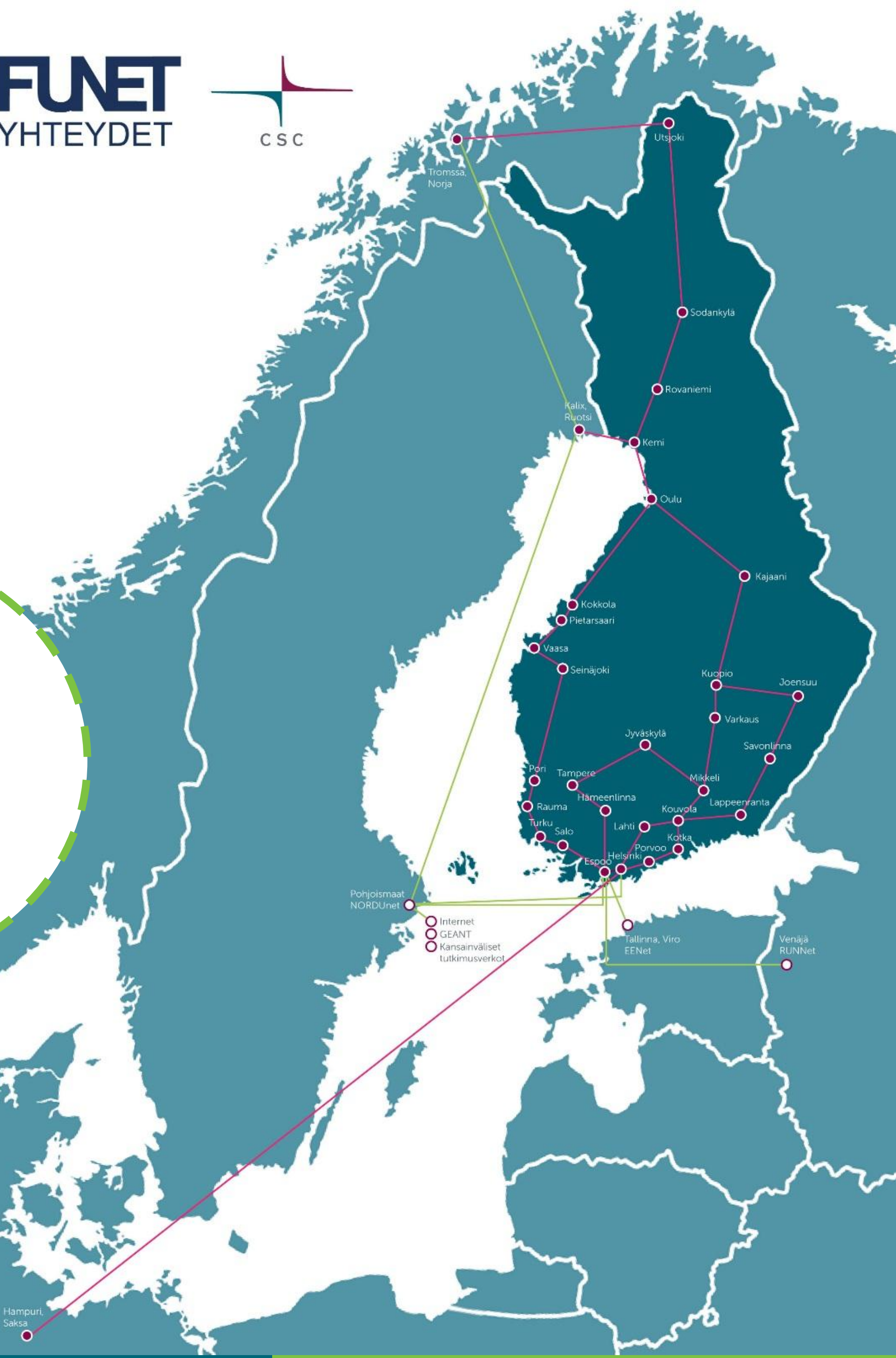
- Funet Etuubi Video Management System
- Funet Silta Video Conferencing MCU Service
- Funet Tiimi Web Conferencing System
- Funet light Paths
- Router Service
- Streaming Service

Ca 80

Funet members

360 000

end-users

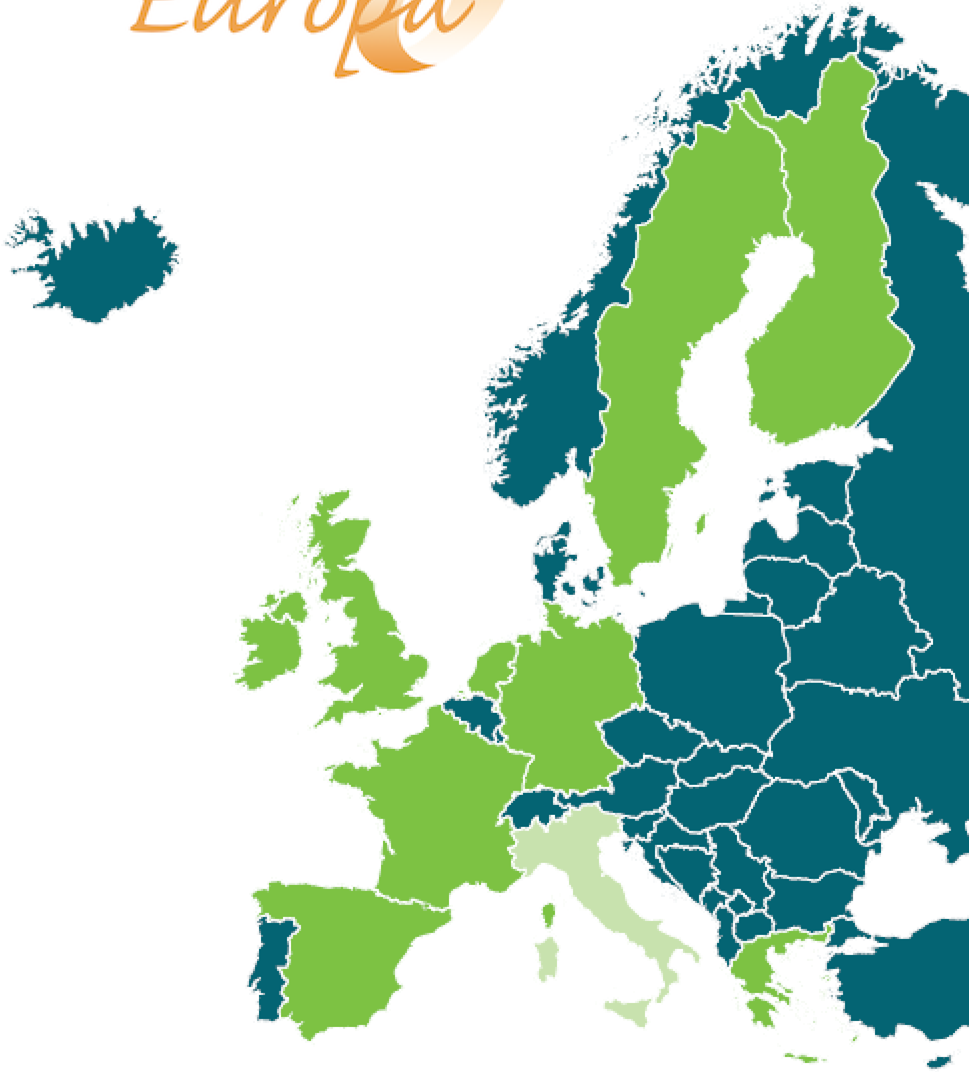


Internationally competitive research environments and e-Infrastructures

Collaboration with majority of European computing centers

- International research network organizations:
  - *NORDUnet, eduGAIN, GÉANT (GN3)*
- European research infrastructures and supporting projects:
  - *ELIXIR, CLARIN, ENVRI*
- International HPC projects and GRID-organizations:
  - *Nordic e-Infrastructure Collaboration (NeIC), PRACE, EGI-Inspire*
- European centres of excellence:
  - *NOMAD, E-CAM*
- European e-Infrastructure policy initiatives :
  - e-Infrastructure Reflection Group (*e-IRG*), *RDA*

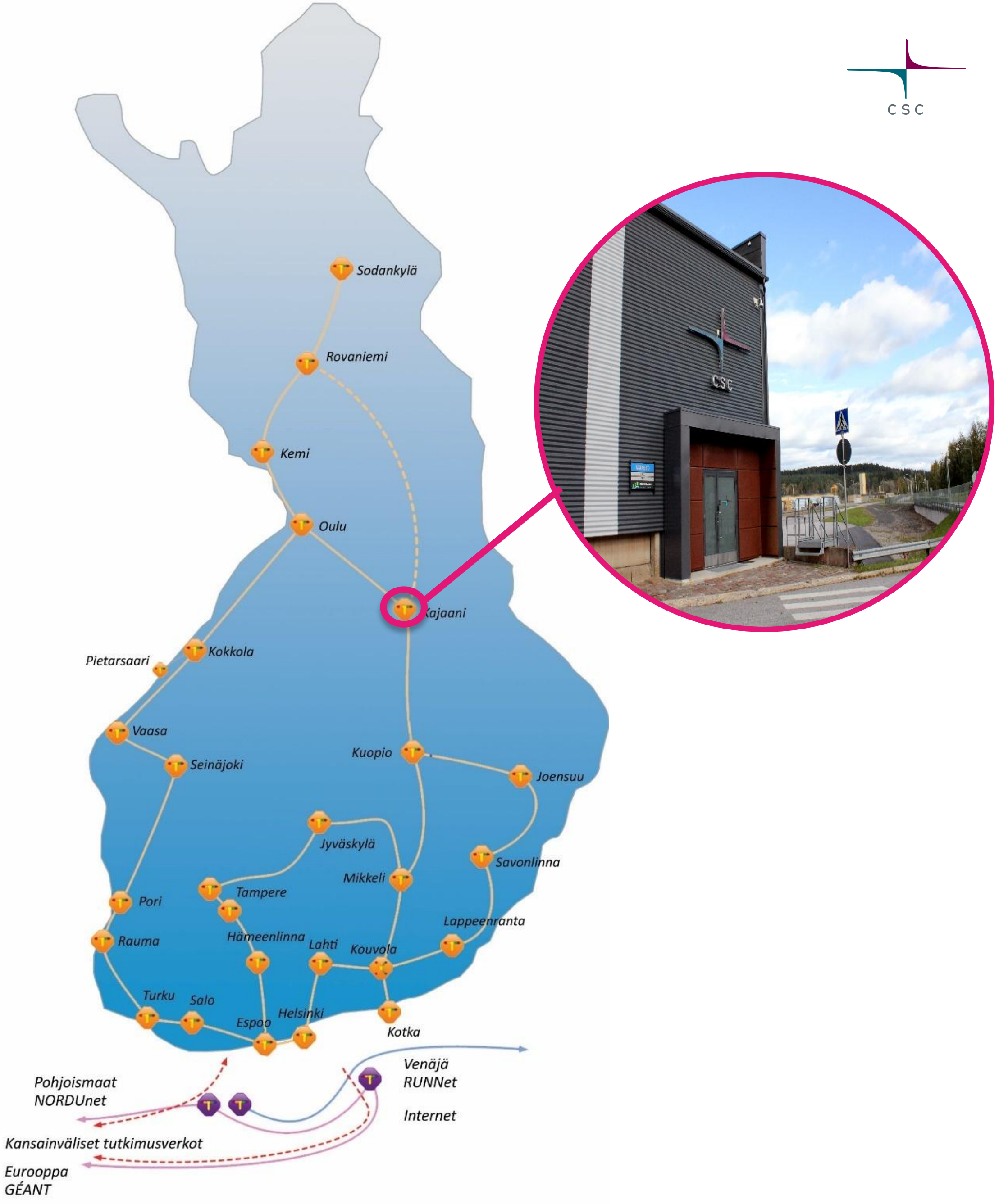
HPC-Europa3 - travel, learn, network



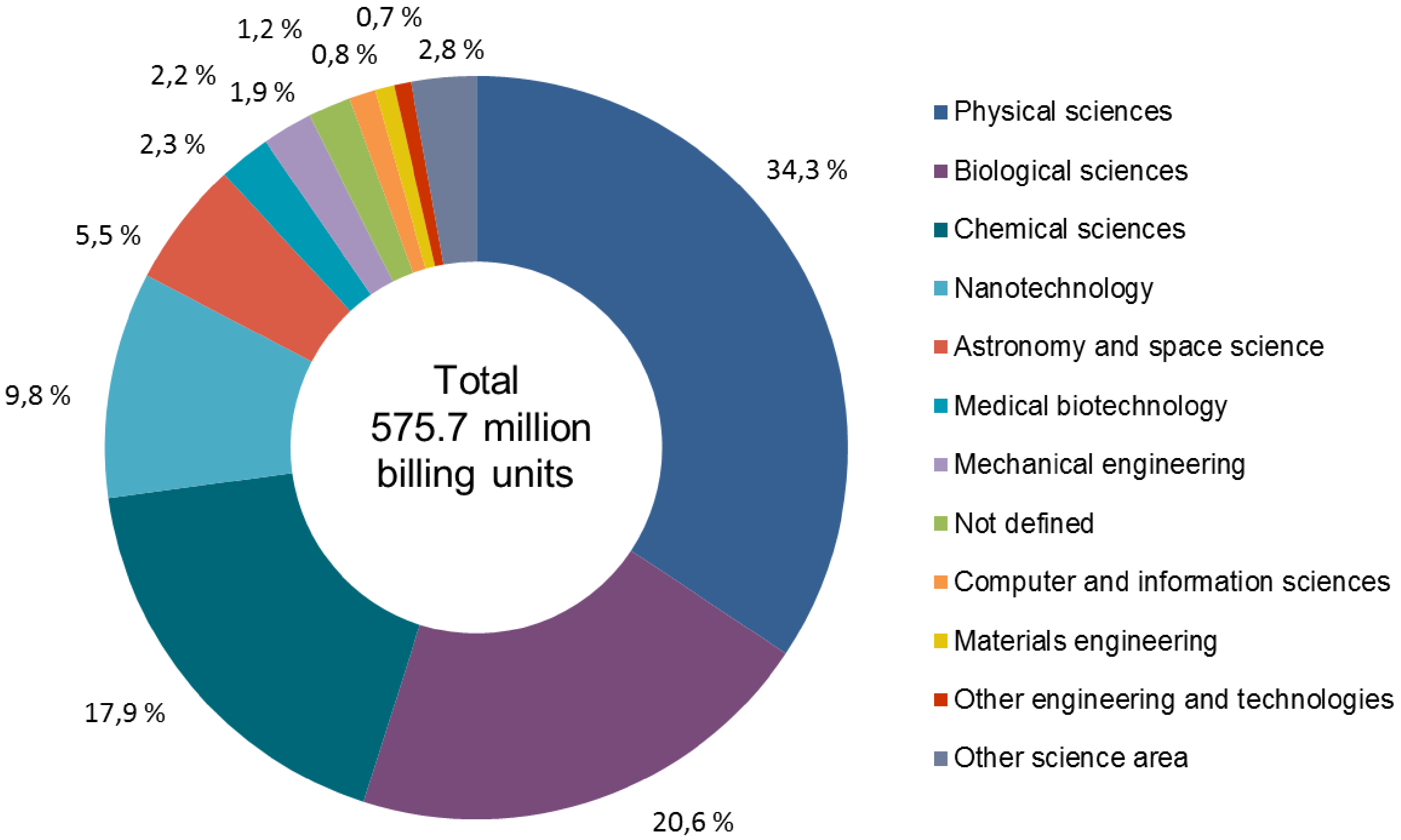
- Four calls every year until April 2021
  - Visit a group or invite a (coming) collaborator to your group
  - Next DL for applications is 16th November
- Requirements:
  - Non-proprietary research
  - Affiliated at an EU-country, associated country or other
  - Project needs/benefits of HPC resources
- Provided:
  - Support for accommodation, travel costs for 3-13 weeks and likely a small daily allowance
  - Resources and support from local HPC-center
- http://www.hpc-europa.org/

CSC’s data center in Kajaani

- CSC’s modular data center in Kajaani. Modern and reliable infrastructure (national power grid, roads, airline connections, data networks)
- The Funet network ensures excellent networking capabilities around the world
- Place for CSC’s next supercomputers with other CSC customer systems
- Cost-Efficient solution – Sustainable and green energy supply

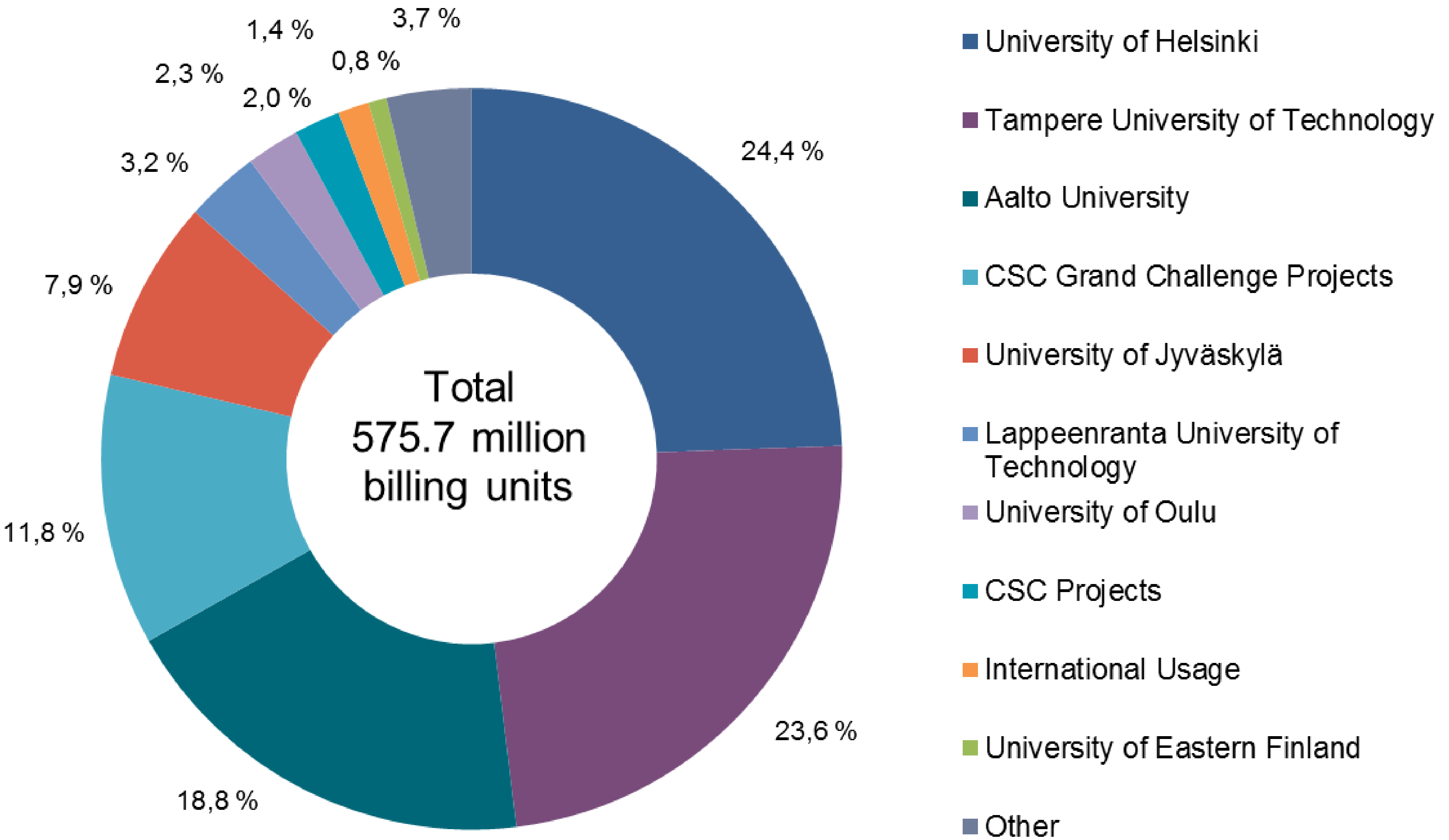


Computing Server Usage by Science Area



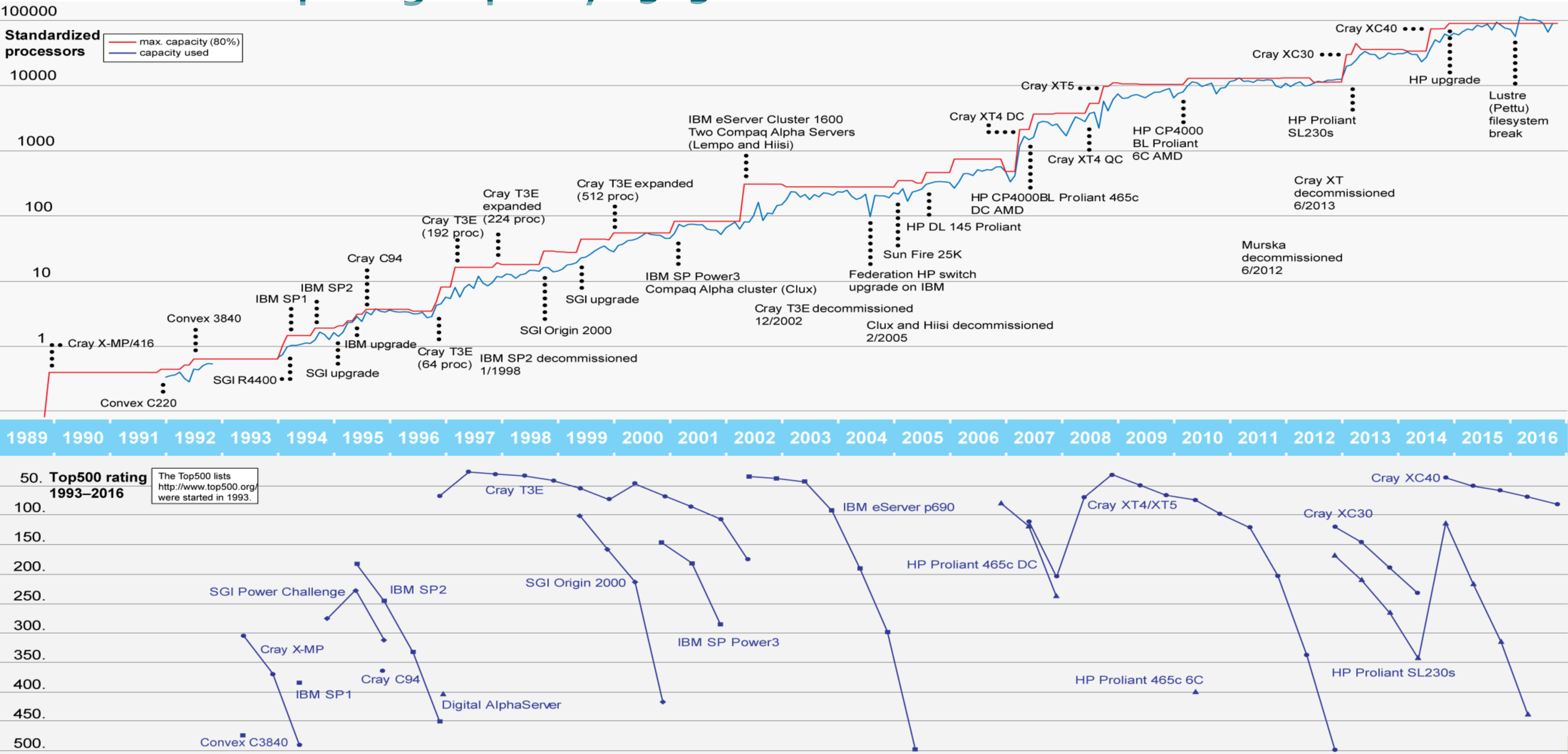
Q1-Q3/2016

Computing Server Usage by Organization



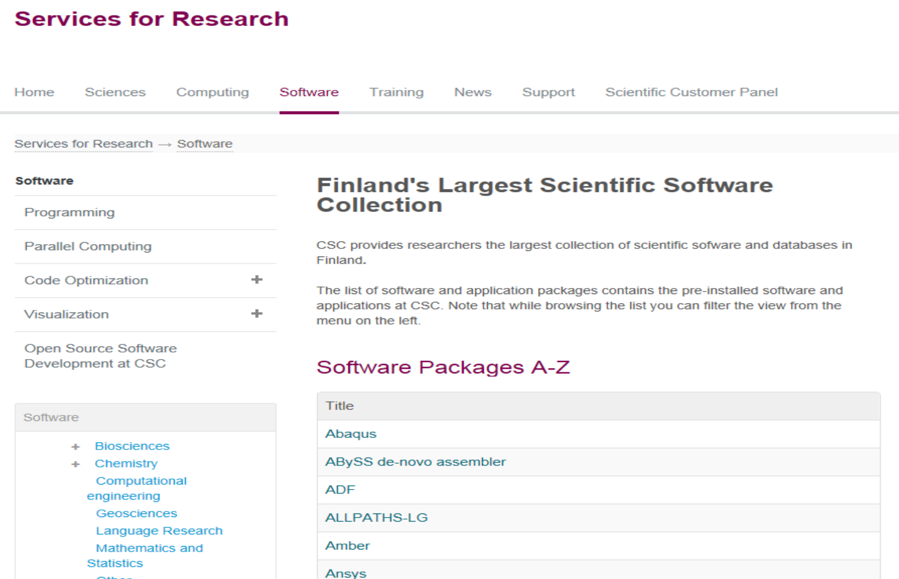
Q1-Q3/2016

CSC's Computing Capacity 1989–2016



Software and database offered by CSC

- Large selection (over 200) of software and database packages for research research.csc.fi/software
- Mainly for academic research in Finland
- Centralized national offering: software consortia, better licence prices, continuity, maintenance, training and support



Services for Research

Home · Sciences · Computing · Software · Training · News · Support · Scientific Customer Panel

Services for Research → Software

Software

- Programming
- Parallel Computing
- Code Optimization +
- Visualization +
- Open Source Software Development at CSC

Software

- + Biosciences
- + Chemistry
- Computational engineering
- Geosciences
- Language Research
- Mathematics and Statistics

Finland's Largest Scientific Software Collection

CSC provides researchers the largest collection of scientific sofware and databases in Finland.

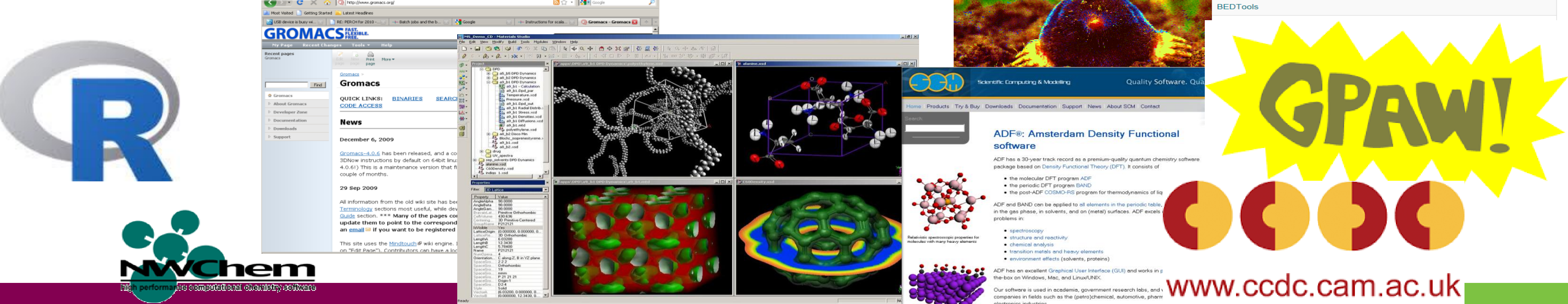
The list of software and application packages contains the pre-installed software and applications at CSC. Note that while browsing the list you can filter the view from the menu on the left.

Software Packages A-Z

Title
Abaqus
ABySS de-novo assembler
ADF
ALLPATHS-LG
Amber
Ansys
ArcGIS
Babel
BEAST
BEDTools

OpenFOAM

The Open Source CFD Toolbox



http://www.gromacs.org/

Most Visited · Getting Started · Latest Headlines

GROMACS FAST. FLEXIBLE. FREE.

My Page · Recent Changes · Tools · Help

Recent pages
Gromacs

Find

- Gromacs
- About Gromacs
- Developer Zone
- Documentation
- Downloads
- Support

Gromacs >

Gromacs

QUICK LINKS: BINARIES SEARC
CODE ACCESS

News

December 6, 2009

Gromacs-4.0.6 has been released, and a co
3DNow instructions by default on 64bit linu
4.0.6!) This is a maintenance version that fi
couple of months.

29 Sep 2009

All information from the old wiki site has bee
Terminology sections most useful, while dev
Guide section. *** **Many of the pages co
update them to point to the correspond
an email if you want to be registered**

This site uses the Mindtouch wiki engine.
on "Edit Page"). Contributors can have a loc

MS_Demo_CD - Materials Studio

Scientific Computing & Modelling · Quality Software. Qua

Home · Products · Try & Buy · Downloads · Documentation · Support · News · About SCM · Contact

Search:

ADF®: Amsterdam Density Functional software

ADF has a 30-year track record as a premium-quality quantum chemistry software package based on Density Functional Theory (DFT). It consists of

- the molecular DFT program ADF
- the periodic DFT program BAND
- the post-ADF COSMO-RS program for thermodynamics of liq

ADF and BAND can be applied to all elements in the periodic table, in the gas phase, in solvents, and on (metal) surfaces. ADF excels problems in:

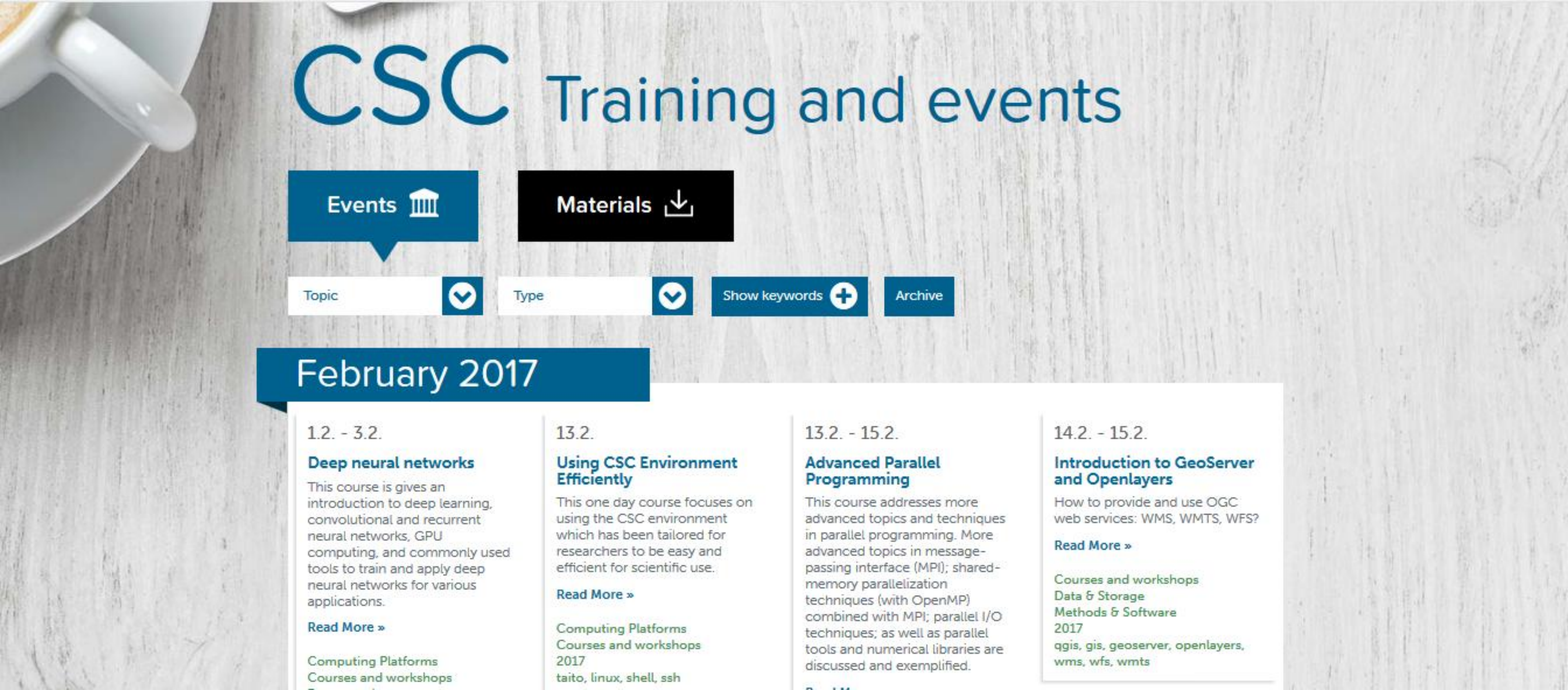
- spectroscopy
- structure and reactivity
- chemical analysis
- transition metals and heavy elements
- environment effects (solvents, proteins)

ADF has an excellent Graphical User Interface (GUI) and works in p the-box on Windows, Mac, and Linux/UNIX.

Our software is used in academia, government research labs, and companies in fields such as the (petro)chemical, automotive, pharm

Relativistic spectroscopic properties for molecules with many heavy elements

Courses



CSC Training and events

February 2017

1.2. - 3.2.

Deep neural networks

This course is gives an introduction to deep learning, convolutional and recurrent neural networks, GPU computing, and commonly used tools to train and apply deep neural networks for various applications.

Read More »

Computing Platforms
Courses and workshops

13.2.

Using CSC Environment Efficiently

This one day course focuses on using the CSC environment which has been tailored for researchers to be easy and efficient for scientific use.

Read More »

Computing Platforms
Courses and workshops
2017
taito, linux, shell, ssh

13.2. - 15.2.

Advanced Parallel Programming

This course addresses more advanced topics and techniques in parallel programming. More advanced topics in message-passing interface (MPI); shared-memory parallelization techniques (with OpenMP) combined with MPI; parallel I/O techniques; as well as parallel tools and numerical libraries are discussed and exemplified.

14.2. - 15.2.

Introduction to GeoServer and Openlayers

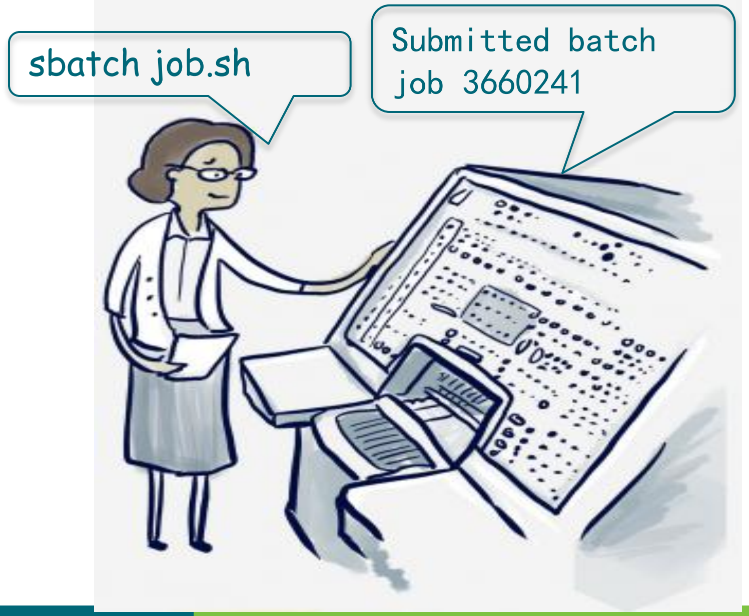
How to provide and use OGC web services: WMS, WMTS, WFS?

Read More »

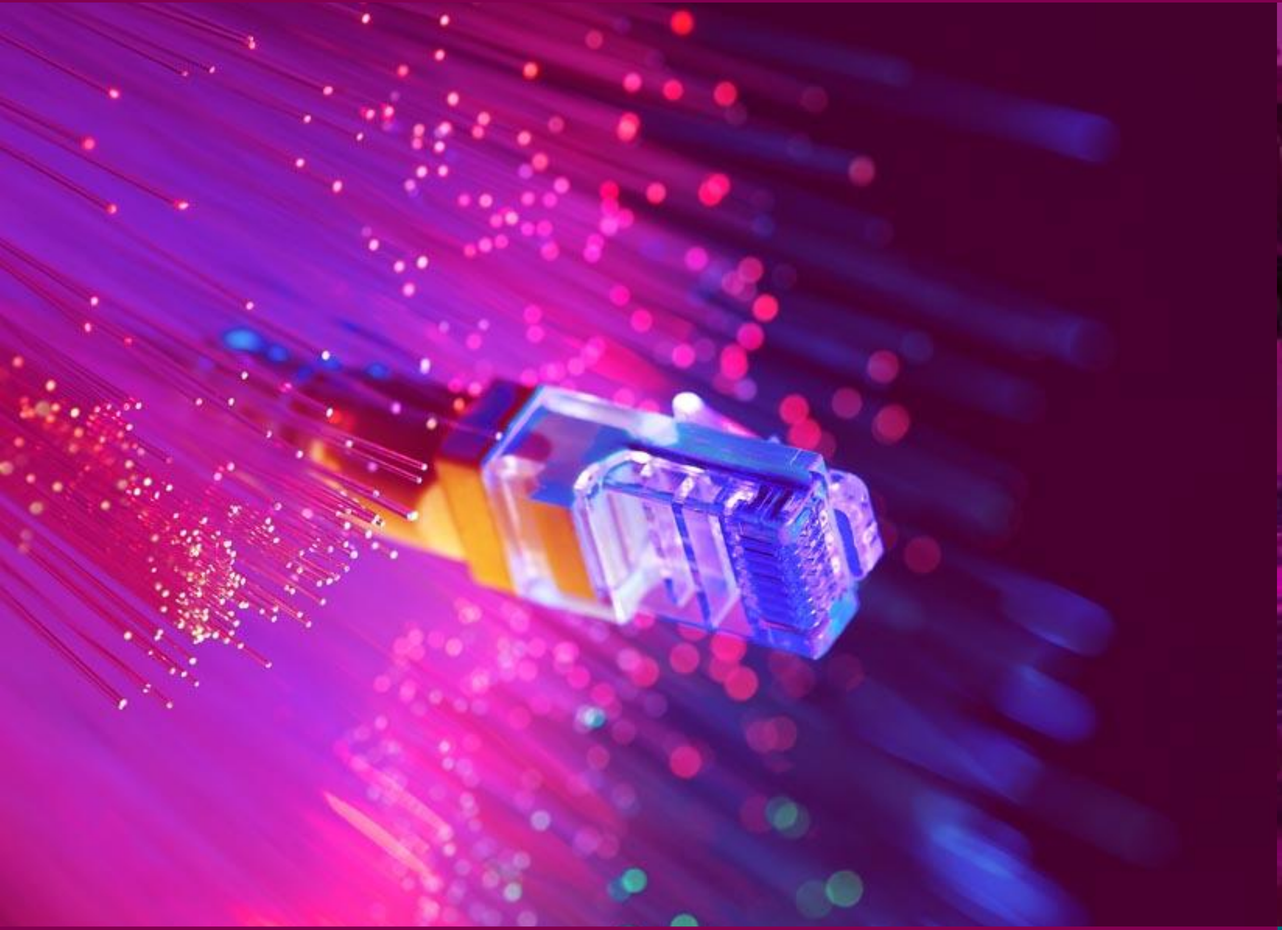
Courses and workshops
Data & Storage
Methods & Software
2017
qgis, gis, geoserver, openlayers, wms, wfs, wmts

How to get started?

- research.csc.fi
- research.csc.fi/csc-guide
- research.csc.fi/faq-knowledge-base
- www.csc.fi/web/training/materials → CSC-Environment
- Service Desk: servicedesk@csc.fi



Getting access to CSC resources



Getting access: in short

User Account

- Register to get a CSC user account
  - Self service
  - You get a user account and a Personal Project
  - Your university account is different (used e.g. in HAKA)

Academic Project

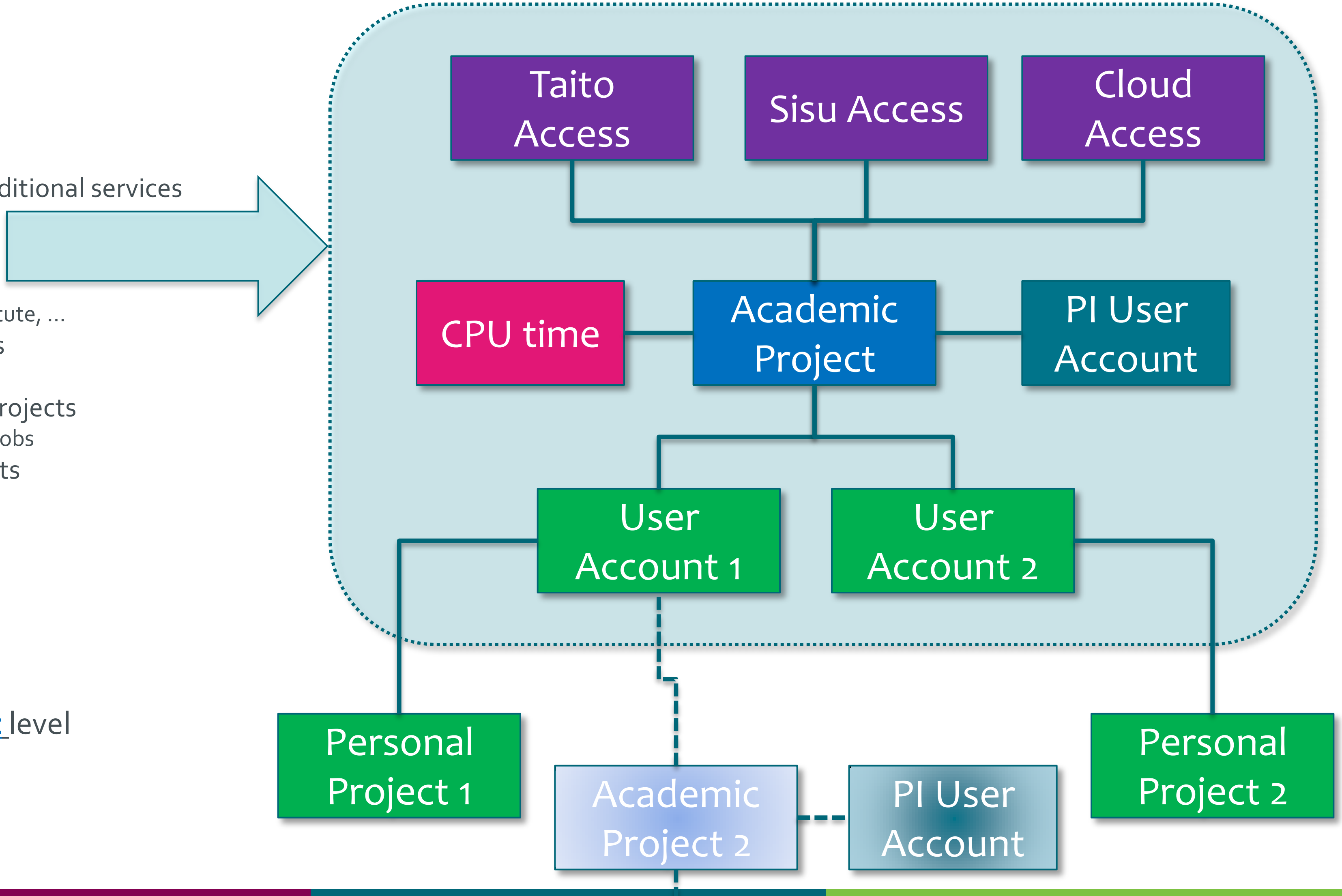
- Apply for an Academic Project
  - Your supervisor (PI) needs to do this
  - Or if your supervisor has a project, ask him/her to invite you to it
  - Set it as your primary billing project

Taito Access

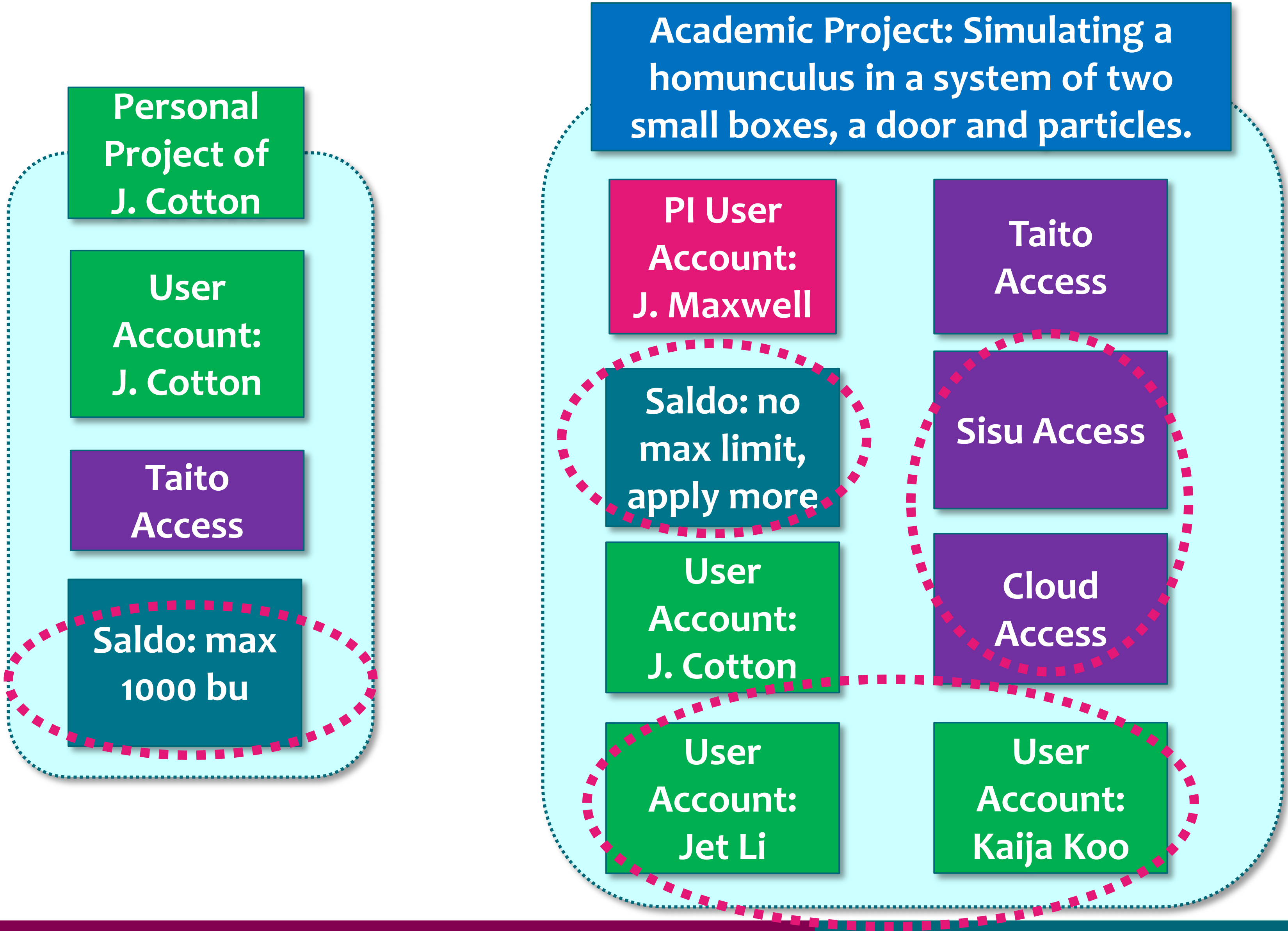
- Apply for a Service *e.g.* Taito cluster access
  - Your supervisor needs to do this
  - If the project existed, it likely already had services

Getting access: The framework

- CSC User Account
  - Is attached to a *Personal Project*
  - Used to log in at CSC
  - Each researcher should have only one
  - Includes access to Taito and little CPU but no additional services
- Academic Project
  - Created by your Principal Investigator (PI)
    - Theses supervisor, professory at your institute, …
  - A research group can have one or many Projects
    - E.g. one per major topic
  - Resources (CPU time, disk space) are parts of Projects
    - This CPU time is consumed when you run jobs
  - Academic Project usually has many user accounts
  - Your account can belong to many projects
- Services
  - E.g. access to Taito, Sisu or cloud
  - Services are properties of Academic Projects
  - PI Accept Terms of Use (link via email)
- Resources are managed at Academic Project level
  - More CPU time is applied to Academic Project



Getting access: Personal vs. academic projects


Personal Project of J. Cotton
User Account: J. Cotton
Taito Access
Saldo: max 1000 bu
Academic Project: Simulating a homunculus in a system of two small boxes, a door and particles.
PI User Account: J. Maxwell
Taito Access
Saldo: no max limit, apply more
Sisu Access
User Account: J. Cotton
Cloud Access
User Account: Jet Li
User Account: Kaija Koo

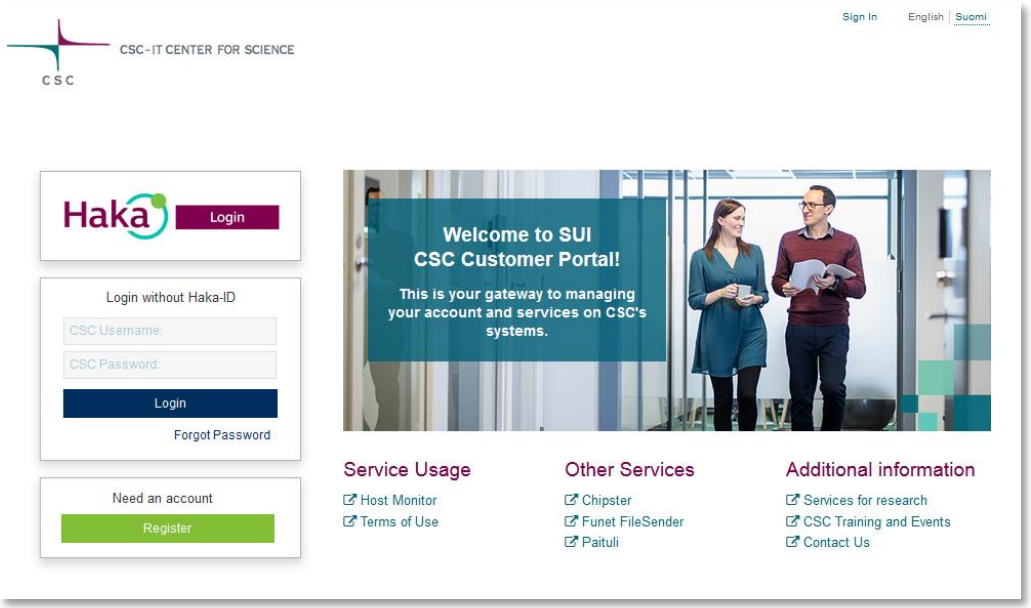
How much is 1000 bu?

It depends.

Using a GUI, you'll be fine for a year. On the other hand, one Gromacs simulation with 100k particles (~10 nm wide system) for 1000 ns will use ~50000 bu in five days on ~150 cores.

1. Register: User account

- https://research.csc.fi/csc-guide-getting-access-to-csc-services
- https://sui.csc.fi
  - Use the green box ”Register” and login with your HAKA account
- This will get you an initial computing quota
  - Sending computation job consumes processor cores
  - User gets a Personal Project with 1’000 billing units (500 core-hours) and access to Taito cluster.
    - The personal project is just for piloting, not for large jobs and you cannot apply for additional computing quota or services



Sign In English | Suomi

CSC - IT CENTER FOR SCIENCE

Haka Login

Login without Haka-ID

CSC Username:

CSC Password:

Login

Forgot Password

Need an account

Register

Welcome to SUI
CSC Customer Portal!

This is your gateway to managing your account and services on CSC's systems.

Service Usage
- Host Monitor
- Terms of Use

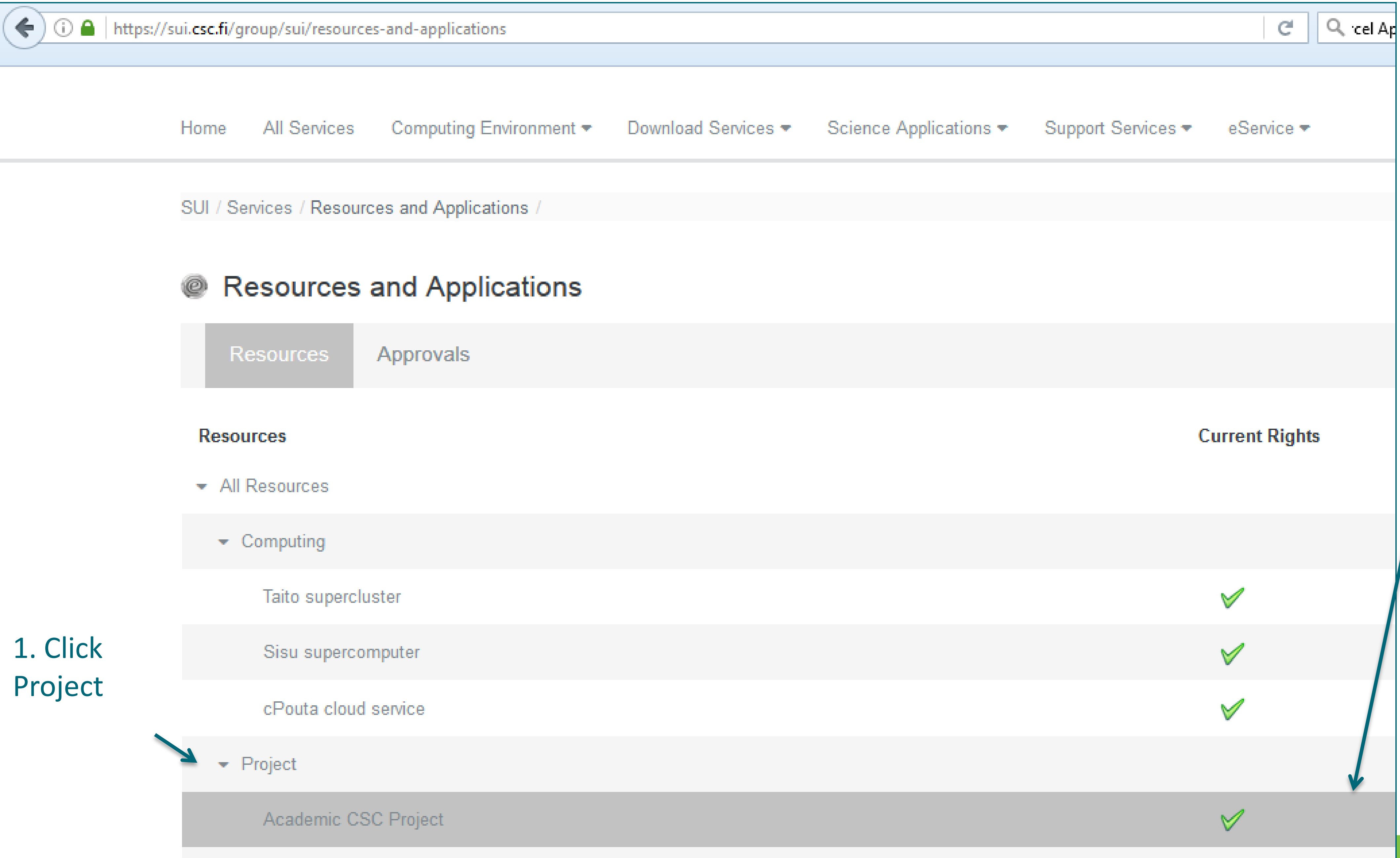
Other Services
- Chipster
- Funet FileSender
- Paituli

Additional information
- Services for research
- CSC Training and Events
- Contact Us

2. Apply for an academic Project

- Professors and PIs can apply for an Academic Project. Note, your PI may already have a project, where your account can be added.
  1. Login via HAKA authentication to SUI https://sui.csc.fi
  2. From eService menu Resources and Applications tool
  3. Fill the application form for the Academic project
     - A screenshot in the next slide
- https://research.csc.fi/csc-guide-projects-and-resource-allocation
- You (the Project) will get 10000 Billing Units by default
- You (your user account) can be a member of several projects

Academic project application form



Resources and Applications

Resources	Current Rights
All Resources	
Computing	
Taito supercluster	✓
Sisu supercomputer	✓
cPouta cloud service	✓
Project	
Academic CSC Project	✓

1. Click Project

2. Click Academic CSC Project

3. Scroll down to see the form

To select the active billing project

- You can select *which project's billing units* is accounted
- In SUI in eService menu select My Projects tool
    1. Select the project from the list
    2. Click "Set as Billing Project" button

Change **the default billing project** from
your Personal Project to the Academic Project when you get it!

To apply for more Billing Units (CPU time)

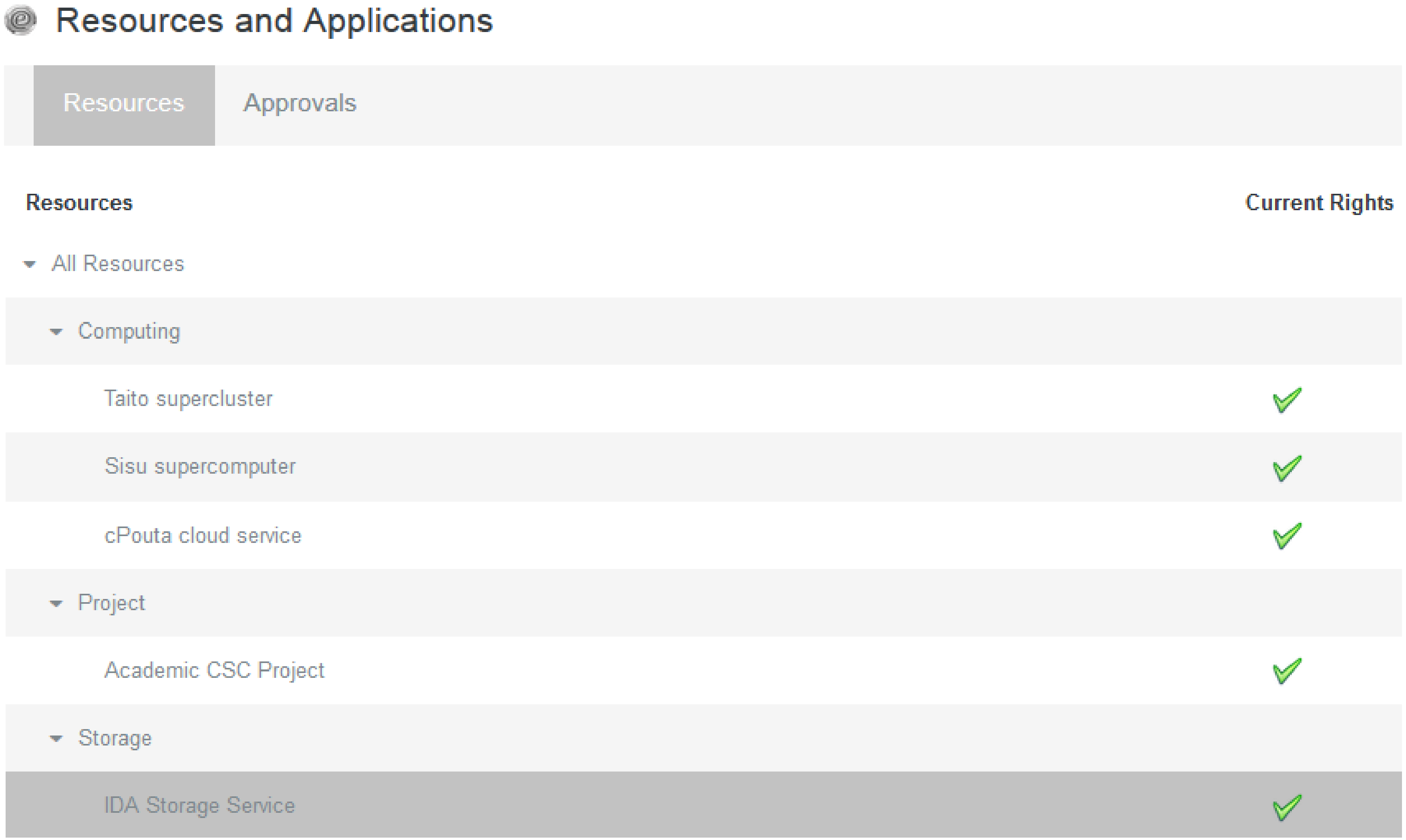
- Any Project Member can apply for more billing units for an Academic Project i.e. not for a Personal Project

- To apply with My Projects tool:
  1. https://sui.csc.fi/group/sui/my-projects or in SUI’s menu select: eService – My Projects
  2. Select the Project you want to apply Billing Units
  3. Click Apply for Resource button
  4. Fill the form and click Send

3. Apply access for a Service

- Only an Academic Project can apply access to Service i.e. not a Personal Project
- Principal Investigator of an Academic Project can apply for access to Taito, Sisu, cPouta and IDA storage Services in SUI
  - https://sui.csc.fi/group/sui/resources-and-applications
- In SUI's menu: eService – Resources and Applications
  - A screenshot on the next slide

Resource and application tool in SUI

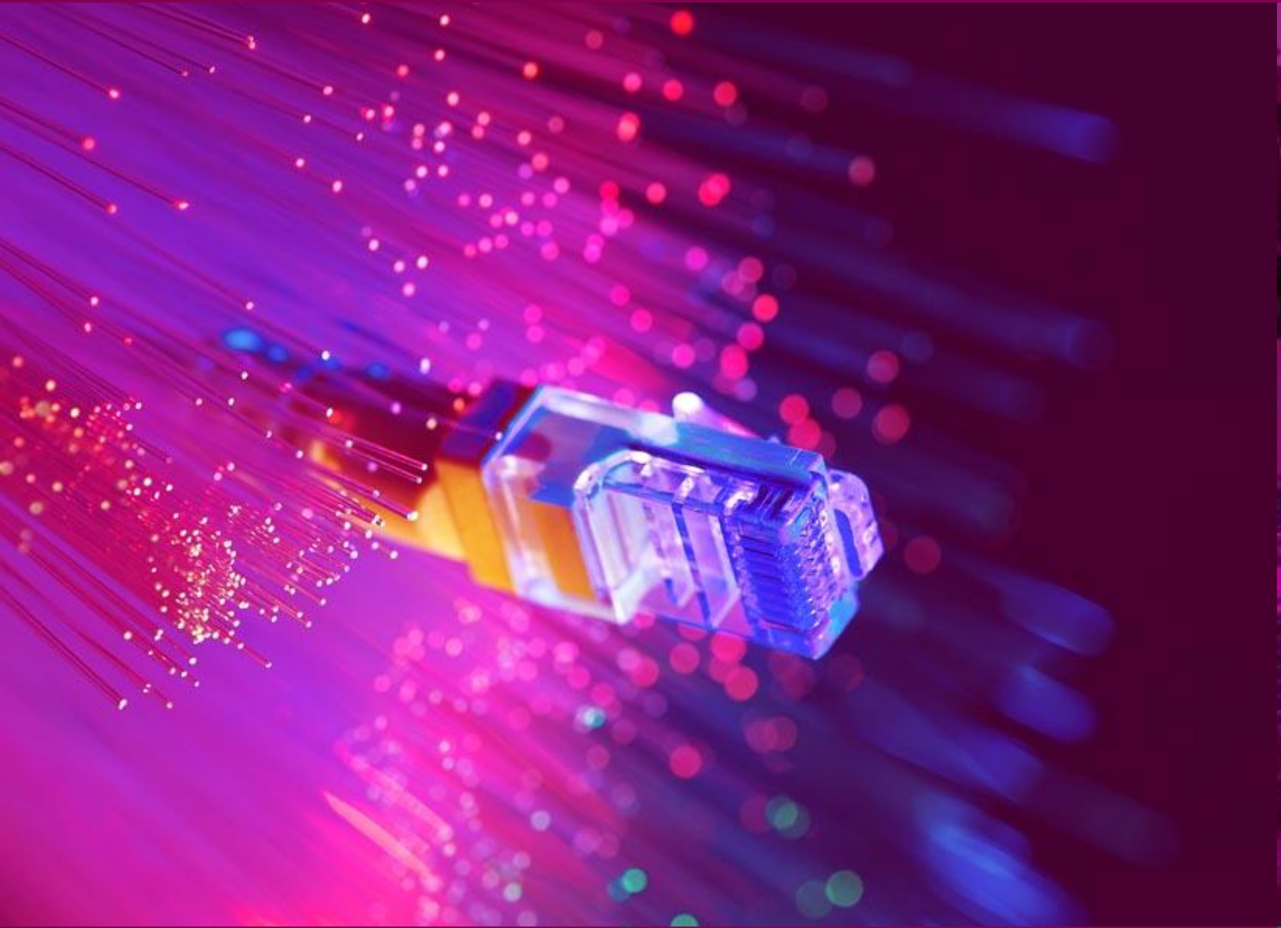


Resources and Applications

Resources	Current Rights
All Resources	
Computing	
Taito supercluster	✓
Sisu supercomputer	✓
cPouta cloud service	✓
Project	
Academic CSC Project	✓
Storage	
IDA Storage Service	✓

The Application form is found below when you select the service

Scientist's User Interface (SUI)

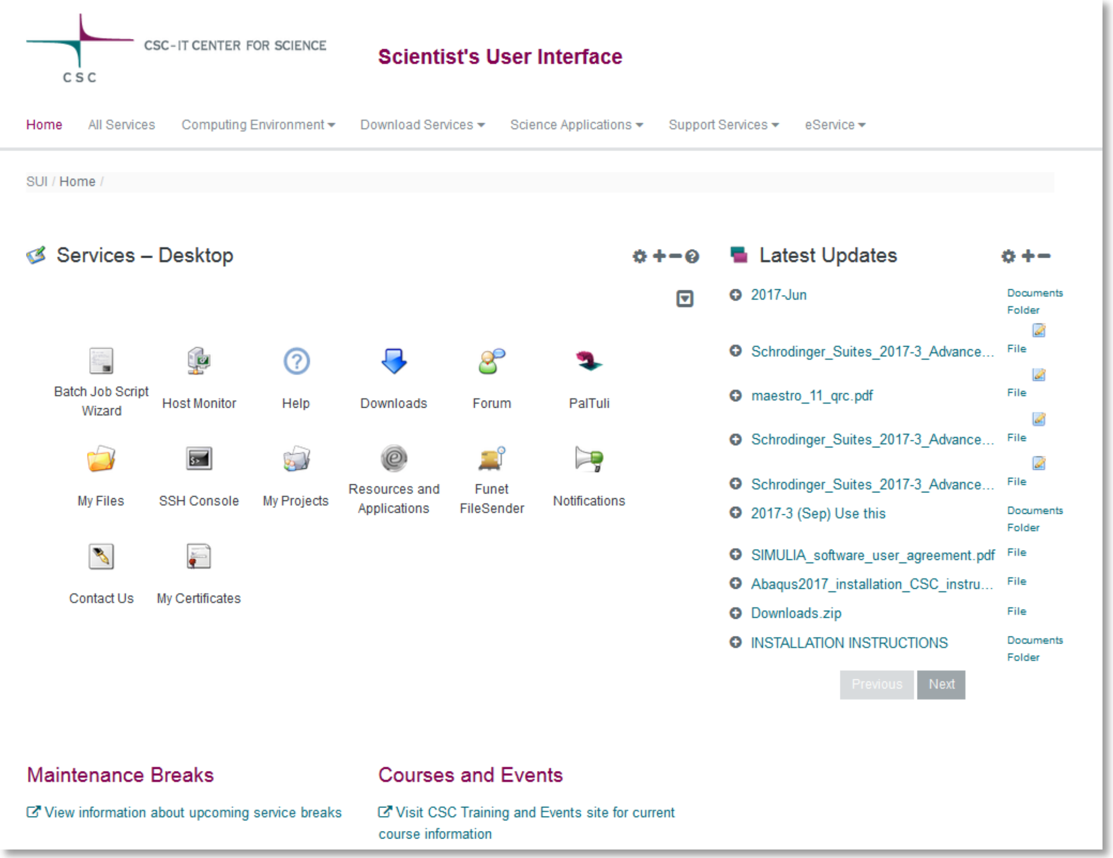


SUI – CSC Customer Portal

Web portal for all CSC users – sui.csc.fi

- **Sign up as customer**
- **Reset your password**
- **Manage your account**
- Apply for an Academic project
- Apply for computing services

- Access your data
- **Download material**
- Watch videos
- Submit jobs
- Monitor hosts and jobs
- Personalize your use
- Message board
- + more



2017-Jun	Documents Folder
Schrodinger_Suites_2017-3_Advance...	File
maestro_11_qrc.pdf	File
Schrodinger_Suites_2017-3_Advance...	File
Schrodinger_Suites_2017-3_Advance...	File
2017-3 (Sep) Use this	Documents Folder
SIMULIA_software_user_agreement.pdf	File
Abaqus2017_installation_CSC_instru...	File
Downloads.zip	File
INSTALLATION INSTRUCTIONS	Documents Folder

Maintenance Breaks

View information about upcoming service breaks

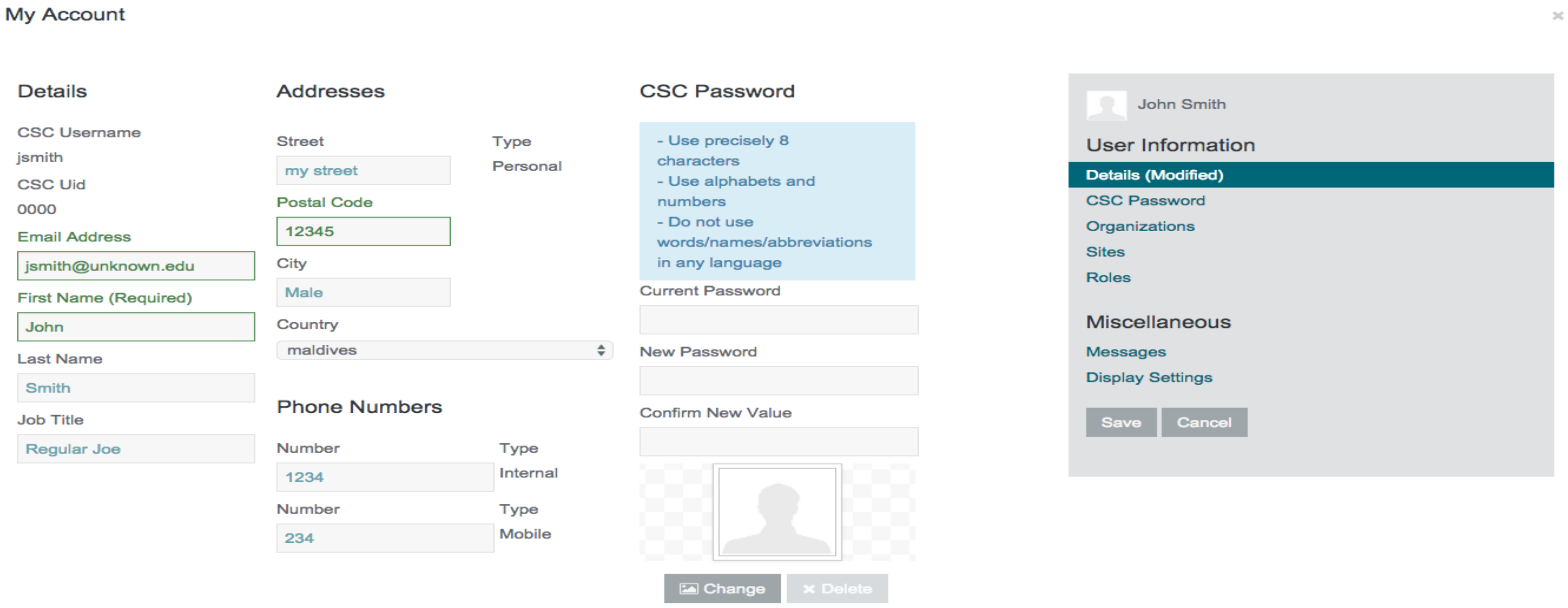
Courses and Events

Visit CSC Training and Events site for current course information

Scientist’s User Interface (SUI)

My Account

- **Maintain** your account information
- **Change password** for CSC environment
- **Define** your personal settings



My Account

Details

CSC Username
jsmith
CSC Uid
0000
Email Address
jsmith@unknown.edu
First Name (Required)
John
Last Name
Smith
Job Title
Regular Joe

Addresses

Street: my street
Type: Personal
Postal Code: 12345
City: Male
Country: maldives

Phone Numbers

Number	Type
1234	Internal
234	Mobile

CSC Password

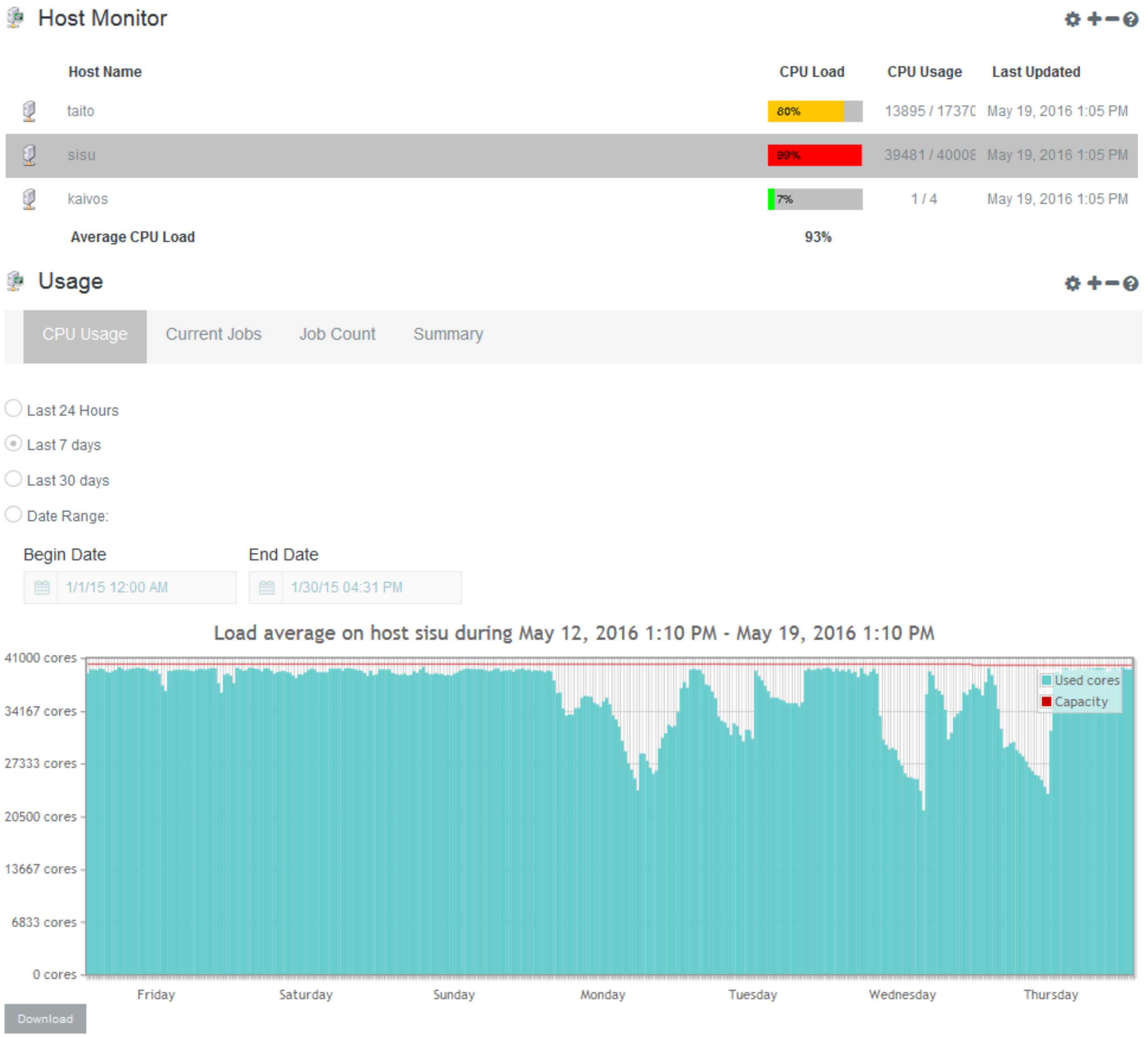
- Use precisely 8 characters
- Use alphabets and numbers
- Do not use words/names/abbreviations in any language

Current Password
New Password
Confirm New Value

Scientist’s User Interface (SUI)

Host Monitor

- **View statuses and details** of CSC’s computing servers and batch systems
- **Visualize history** of CPU usage and job count
- **Monitor jobs** in all hosts in single view
- **Control** your own jobs



Host Monitor

	Host Name	CPU Load	CPU Usage	Last Updated
	taito	80%	13895 / 1737	May 19, 2016 1:05 PM
	sisu	99%	39481 / 4000	May 19, 2016 1:05 PM
	kaivos	7%	1 / 4	May 19, 2016 1:05 PM
	Average CPU Load	93%		

Usage

CPU Usage | Current Jobs | Job Count | Summary

○ Last 24 Hours
● Last 7 days
○ Last 30 days
○ Date Range:

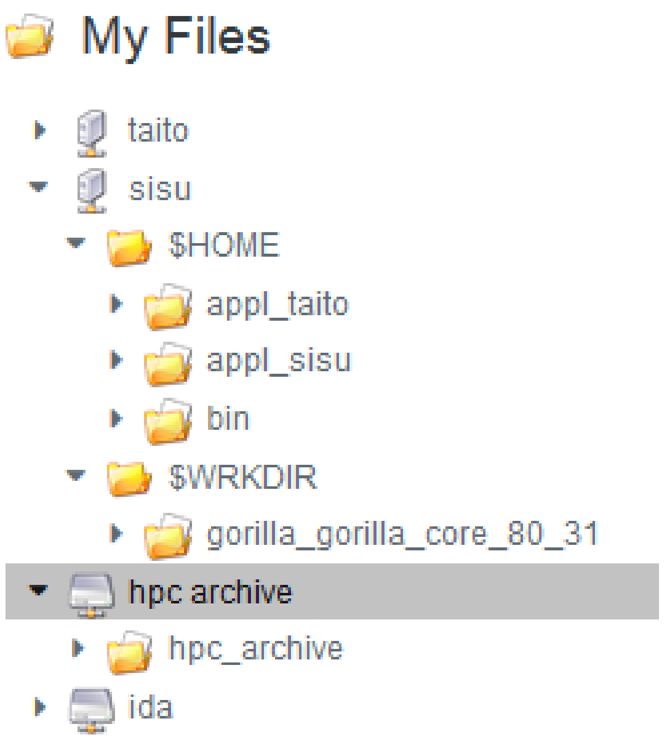
Begin Date: 1/1/15 12:00 AM
End Date: 1/30/15 04:31 PM

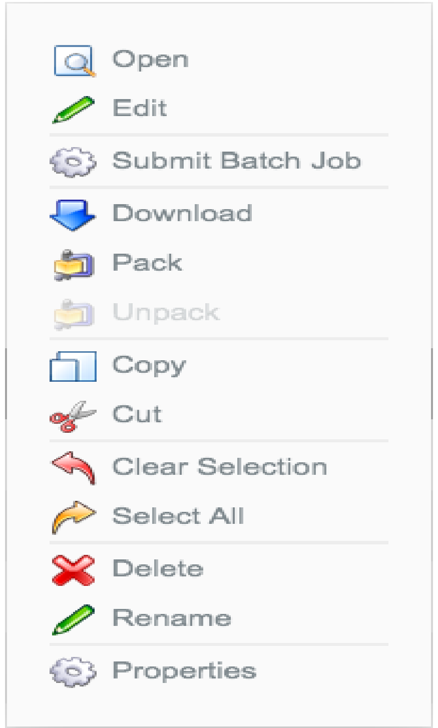
Load average on host sisu during May 12, 2016 1:10 PM - May 19, 2016 1:10 PM

Scientist’s User Interface (SUI)

My Files

- Access your data in CSC’s storage services in single view (computing servers, IDA and HPC Arcl
- Transfer files
- Search your data
- Submit jobs
- Typical folder and file operations are supported

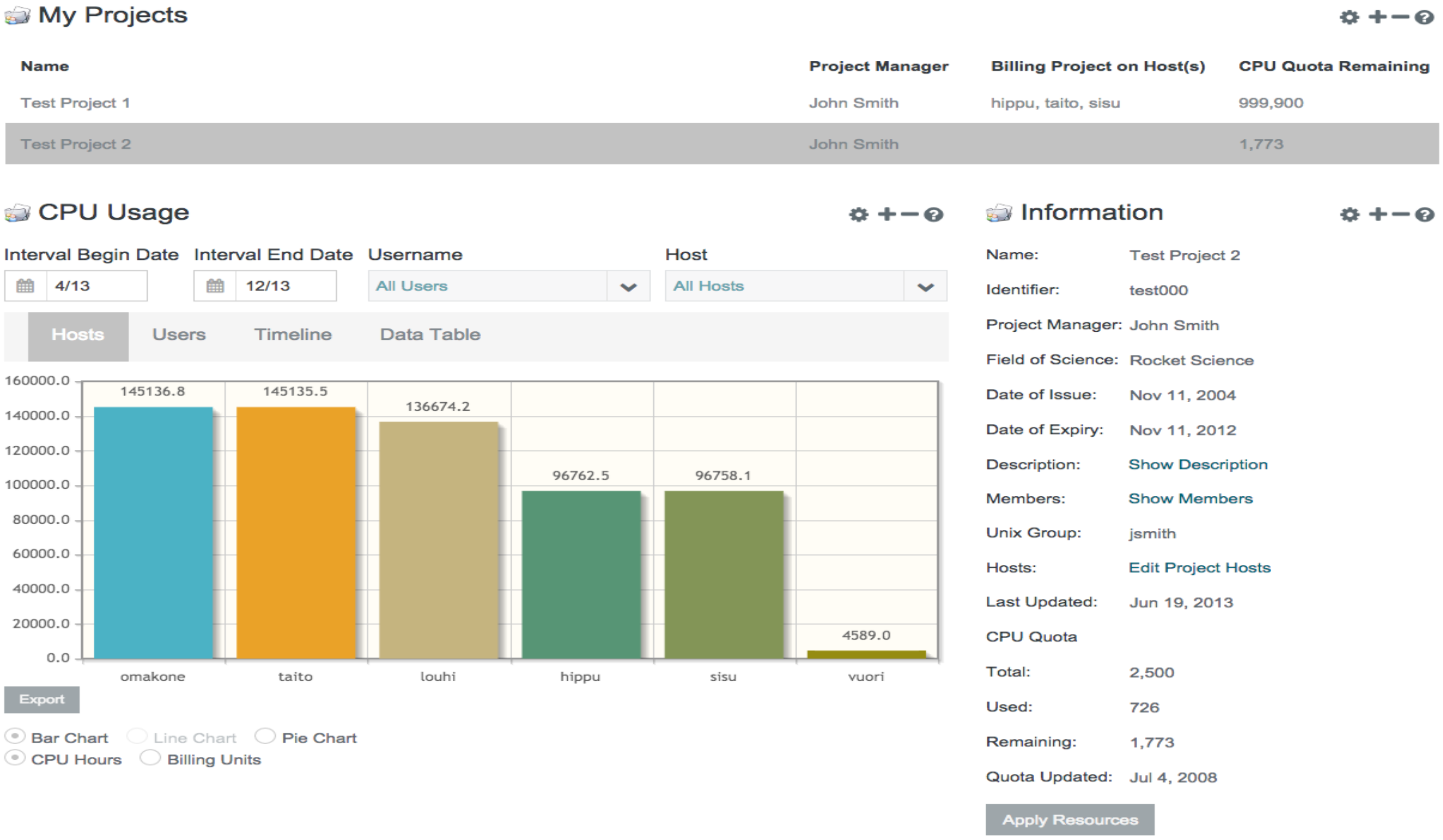




Scientist’s User Interface (SUI)

My Projects

- **View information and resource usage** of your CSC projects
- **Edit hosts** for projects
- **Apply resources** for your CSC customer project
- Resource usage currently not working due system changes



My Projects

Name	Project Manager	Billing Project on Host(s)	CPU Quota Remaining
Test Project 1	John Smith	hippu, taito, sisu	999,900
Test Project 2	John Smith		1,773

CPU Usage

Interval Begin Date	Interval End Date	Username	Host
4/13	12/13	All Users	All Hosts

○ Bar Chart ○ Line Chart ○ Pie Chart
○ CPU Hours ○ Billing Units

Information

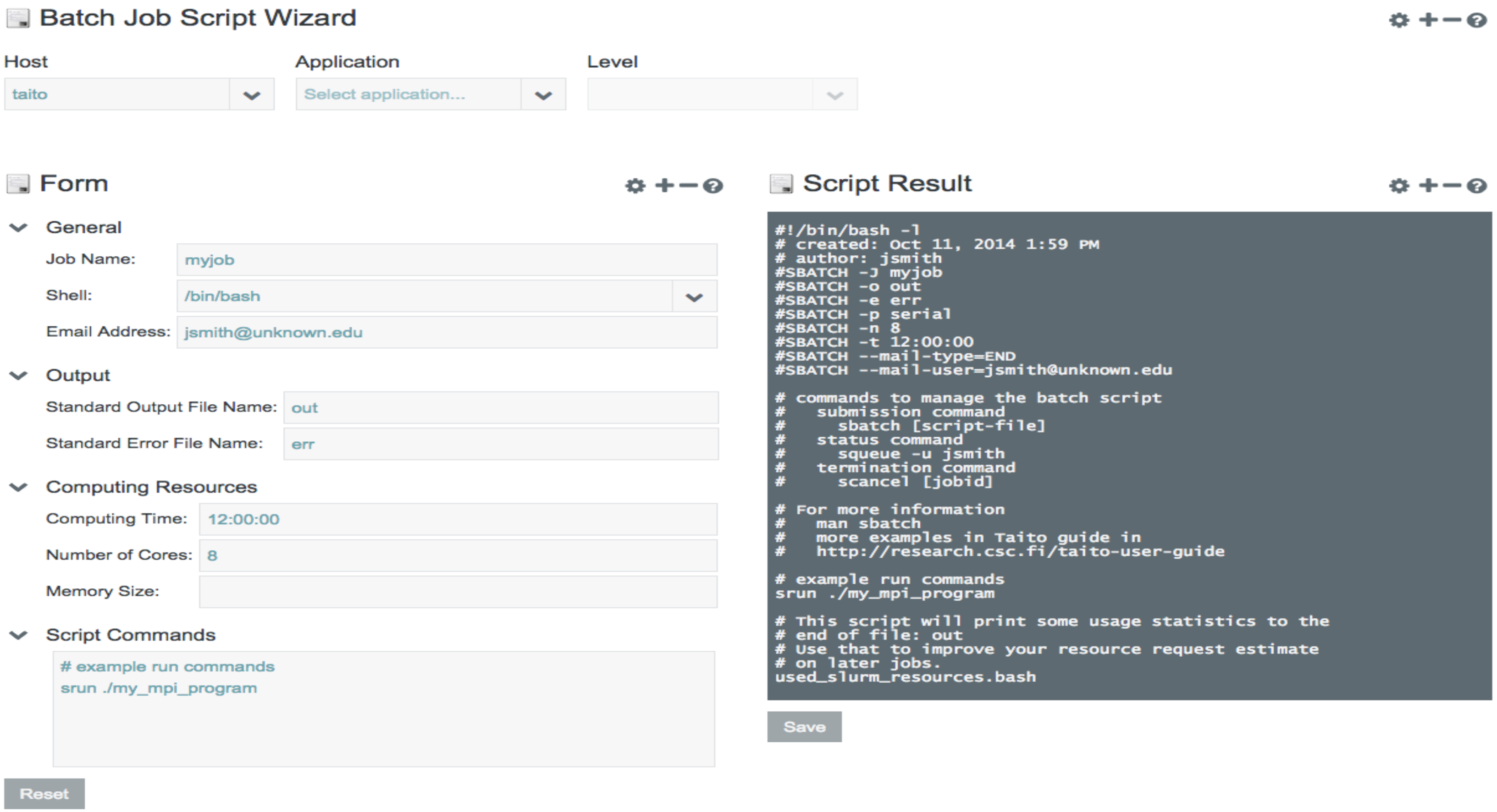
Name:	Test Project 2
Identifier:	test000
Project Manager:	John Smith
Field of Science:	Rocket Science
Date of Issue:	Nov 11, 2004
Date of Expiry:	Nov 11, 2012
Description:	Show Description
Members:	Show Members
Unix Group:	jsmith
Hosts:	Edit Project Hosts
Last Updated:	Jun 19, 2013
CPU Quota	
Total:	2,500
Used:	726
Remaining:	1,773
Quota Updated:	Jul 4, 2008

Apply Resources

Scientist’s User Interface (SUI)

Batch Job Script Wizard

- **Create job scripts** with easy to use forms
- **Save scripts** locally or in CSC **$HOME**
- Instructions of how to submit and monitor



Batch Job Script Wizard

Host	Application	Level
taito	Select application...	

Form

General

Job Name: myjob

Shell: /bin/bash

Email Address: jsmith@unknown.edu

Output

Standard Output File Name: out

Standard Error File Name: err

Computing Resources

Computing Time: 12:00:00

Number of Cores: 8

Memory Size:

Script Commands

```
# example run commands
srun ./my_mpi_program
```

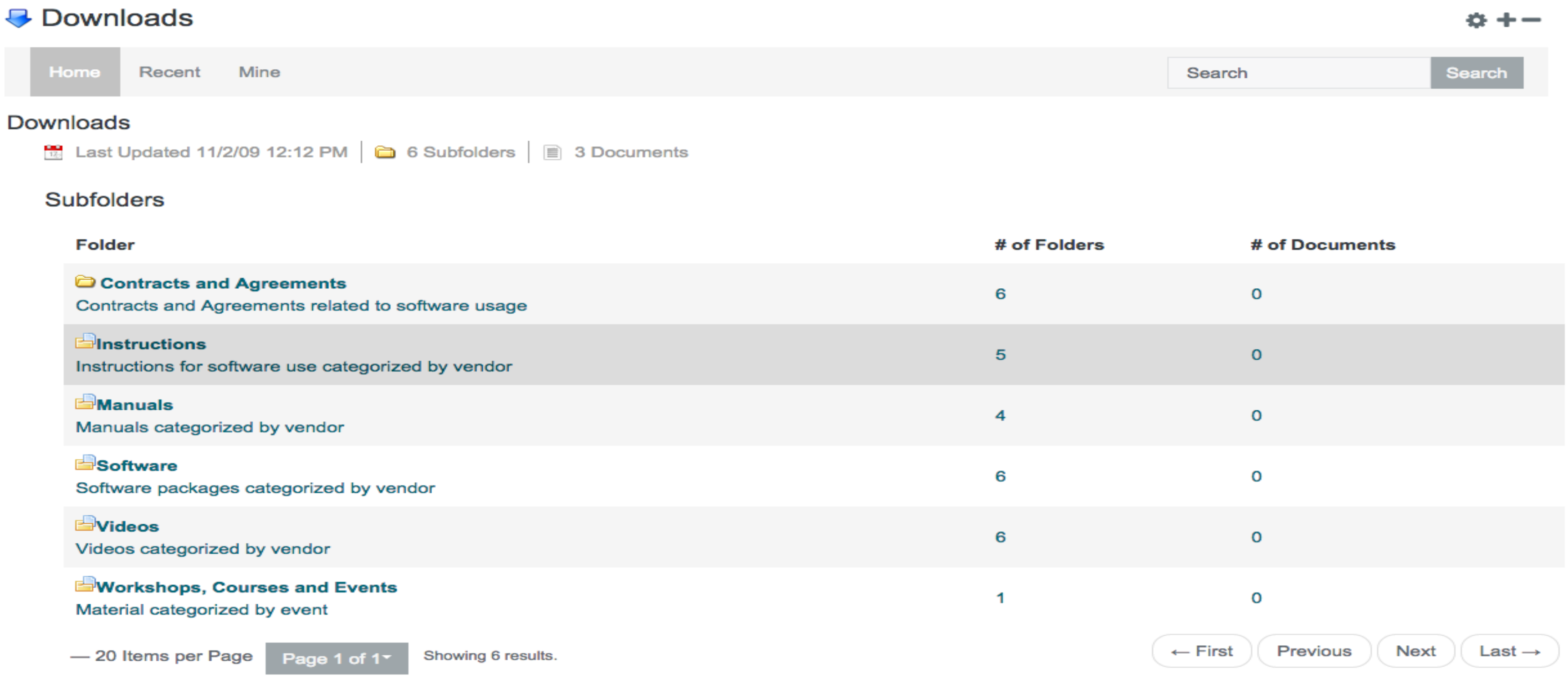
Script Result

```
#!/bin/bash -l
# created: Oct 11, 2014 1:59 PM
# author: jsmith
#SBATCH -J myjob
#SBATCH -o out
#SBATCH -e err
#SBATCH -p serial
#SBATCH -n 8
#SBATCH -t 12:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=jsmith@unknown.edu

# commands to manage the batch script
#   submission command
#     sbatch [script-file]
#   status command
#     squeue -u jsmith
#   termination command
#     scancel [jobid]

# For more information
#   man sbatch
#   more examples in Taito guide in
#   http://research.csc.fi/taito-user-guide

# example run commands
srun ./my_mpi_program

# This script will print some usage statistics to the
# end of file: out
# Use that to improve your resource request estimate
# on later jobs.
used_slurm_resources.bash
```

Scientist’s User Interface (SUI)

Downloads

- **Access material** provided to you by CSC
- Software installation packages, manuals, videos etc.



Downloads

Last Updated 11/2/09 12:12 PM | 6 Subfolders | 3 Documents

Subfolders

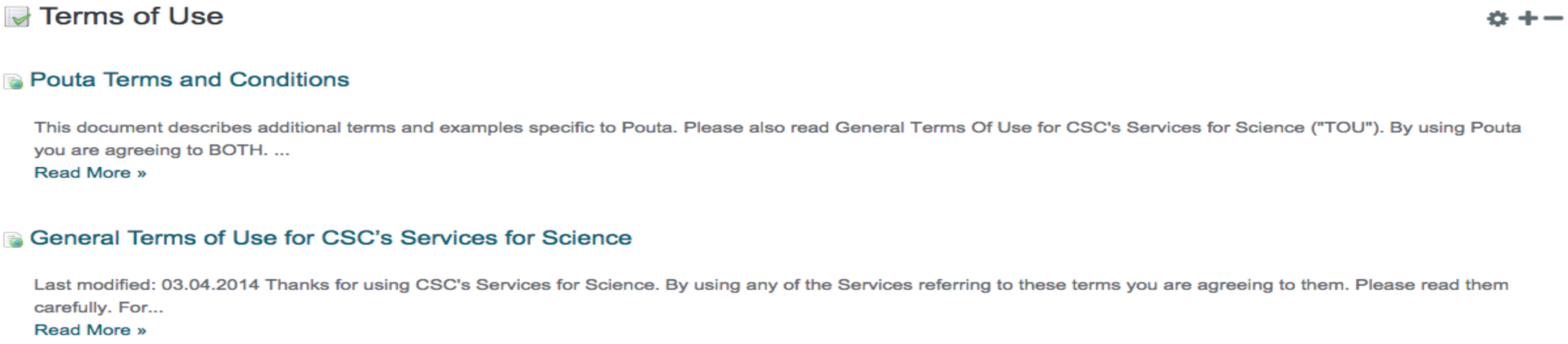
Folder	# of Folders	# of Documents
Contracts and Agreements Contracts and Agreements related to software usage	6	0
Instructions Instructions for software use categorized by vendor	5	0
Manuals Manuals categorized by vendor	4	0
Software Software packages categorized by vendor	6	0
Videos Videos categorized by vendor	6	0
Workshops, Courses and Events Material categorized by event	1	0

— 20 Items per Page | Page 1 of 1 | Showing 6 results.

Scientist's User Interface (SUI)

Terms of Use

- **Read** CSC's services' terms of use



Terms of Use

Pouta Terms and Conditions

This document describes additional terms and examples specific to Pouta. Please also read General Terms Of Use for CSC's Services for Science ("TOU"). By using Pouta you are agreeing to BOTH. ...
Read More »

General Terms of Use for CSC's Services for Science

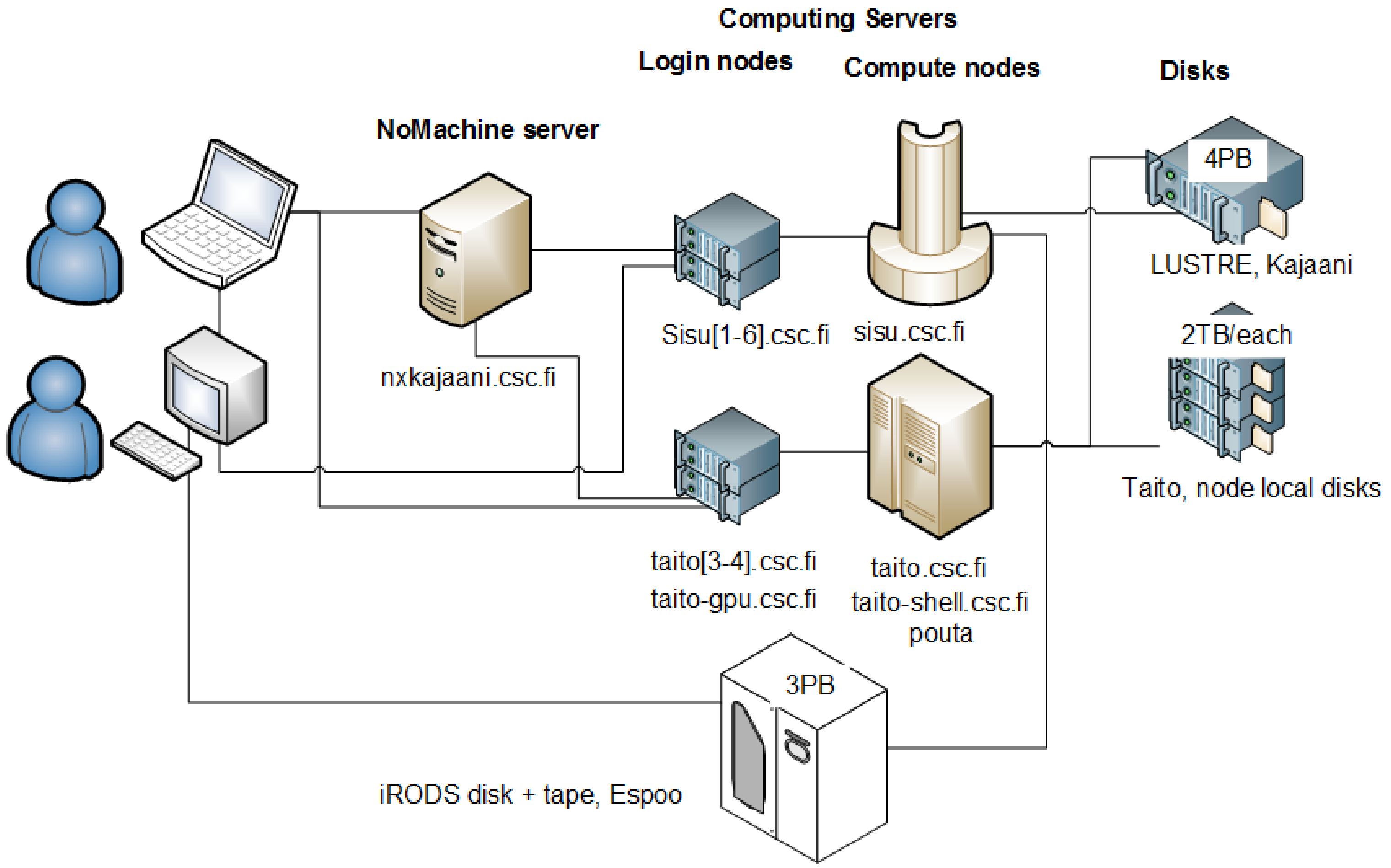
Last modified: 03.04.2014 Thanks for using CSC's Services for Science. By using any of the Services referring to these terms you are agreeing to them. Please read them carefully. For...
Read More »

Connecting to CSC

Learning targets

- Be aware of different ways of accessing CSC resources
- Logged in to Taito with ssh and NoMachine

The (almost) Complete Picture



Access via any of:

- Ssh
- NoMachine
- Browser (SUI, cloud, Avaa, …)
- Tunneling
- ARC (FGCI)
- HAKA
- iRODS

Direct ssh connection –Linux/Mac

- From UNIX/Linux/OSX command line
- Use –X (or –Y) to enable remote graphics*
- scp : copy file to remote machine

```
$ ssh –X yourid@taito.csc.fi
```

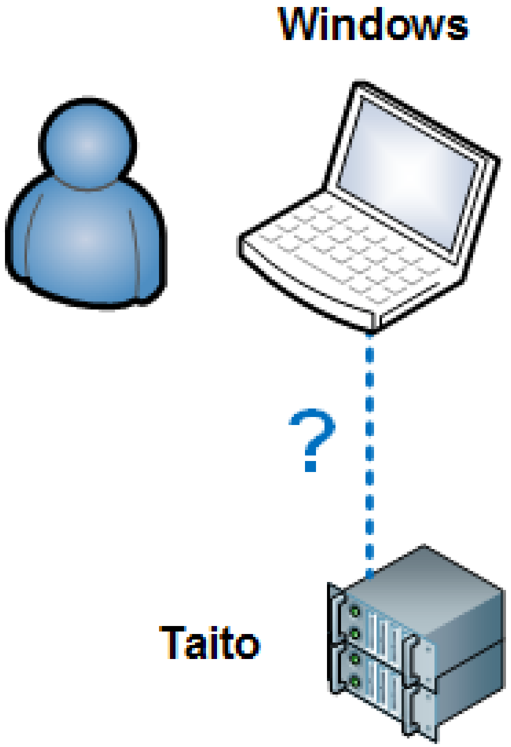
```
$ scp file yourid@taito.csc.fi:
```

```
login as: yourid
Last login: Tue Jul 09 13:14:15 2019 from cool.somewhere.fi
┌─ Welcome ───────────────────────────────────────────────────────────┐
│         CSC - Tieteen tietotekniikan keskus - IT Center for Science │
│                 HP Cluster Platform SL230s Gen8 TAITO               │
├─ Contact ───────────────────────────────────────────────────────────┤
...
```

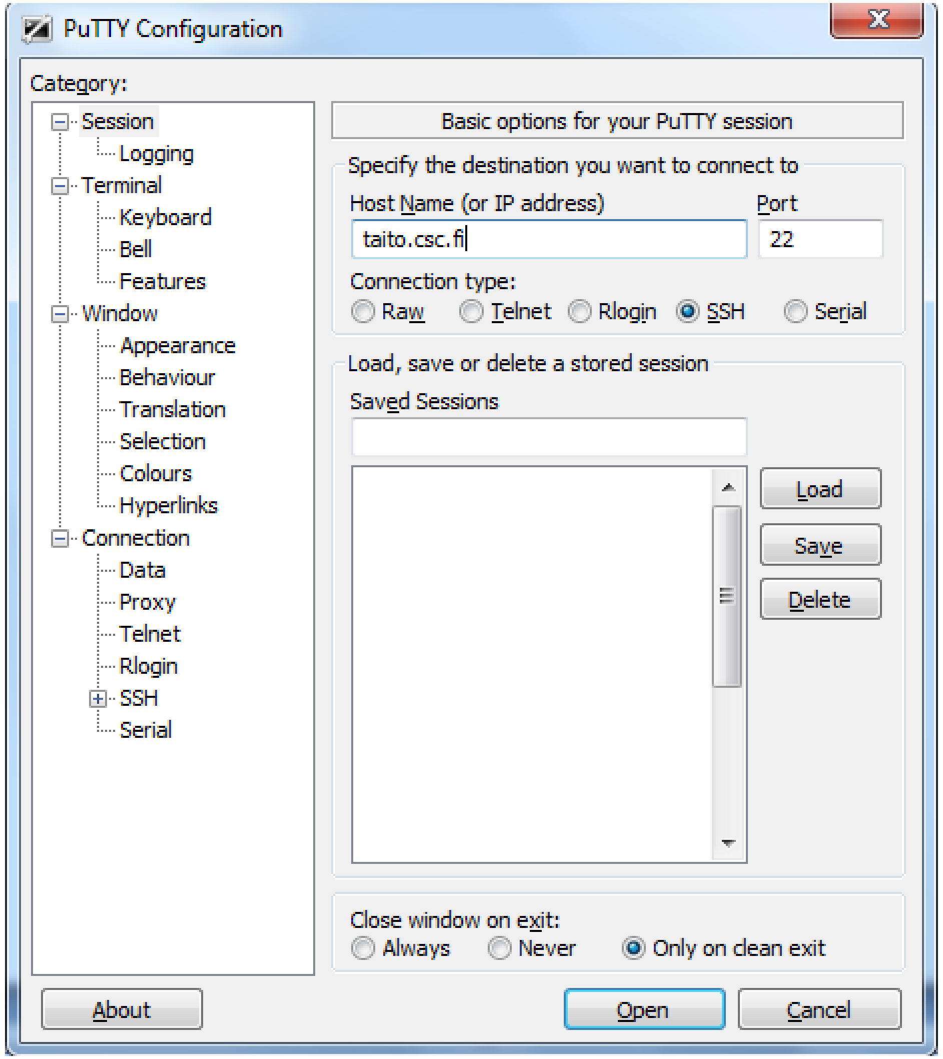
* In Windows you'd also need an X-windows emulator, but there is a better way

Access from Windows

- Putty for ssh connection
  - Can be installed without admin privileges
- NoMachine for GUI
  - Needs Admin privileges for installation and update
  - Recommended method (also for Linux/Mac)
- FileZilla/WinSCP for moving data
  - Efficient GUI
- Find about other access options and more information at: https://research.csc.fi/taito-connecting



Putty



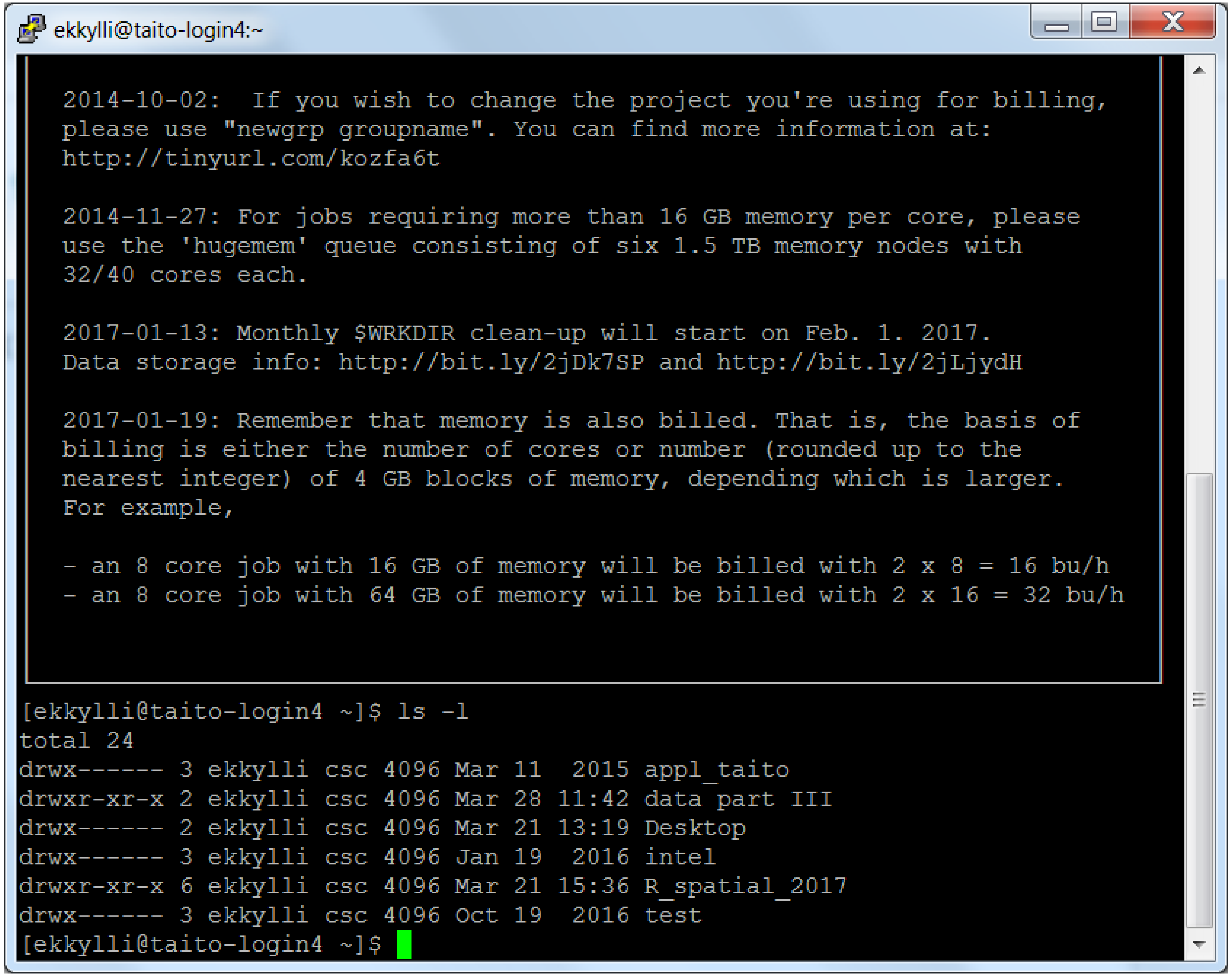
Specify the destination you want to connect to

Host Name (or IP address): taito.csc.fi

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

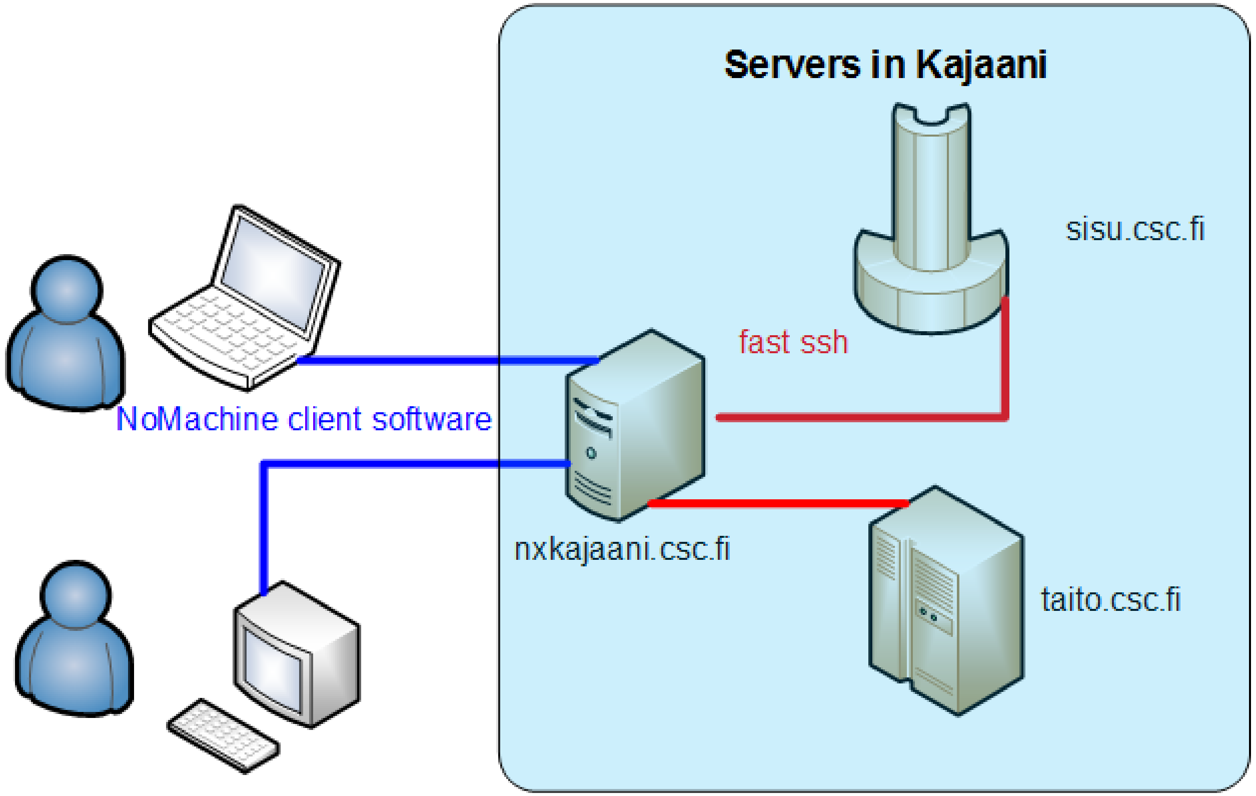
Close window on exit: ○ Always ○ Never ● Only on clean exit



```
  2014-10-02:  If you wish to change the project you're using for billing,
  please use "newgrp groupname". You can find more information at:
  http://tinyurl.com/kozfa6t

  2014-11-27: For jobs requiring more than 16 GB memory per core, please
  use the 'hugemem' queue consisting of six 1.5 TB memory nodes with
  32/40 cores each.

  2017-01-13: Monthly $WRKDIR clean-up will start on Feb. 1. 2017.
  Data storage info: http://bit.ly/2jDk7SP and http://bit.ly/2jLjydH

  2017-01-19: Remember that memory is also billed. That is, the basis of
  billing is either the number of cores or number (rounded up to the
  nearest integer) of 4 GB blocks of memory, depending which is larger.
  For example,

  - an 8 core job with 16 GB of memory will be billed with 2 x 8 = 16 bu/h
  - an 8 core job with 64 GB of memory will be billed with 2 x 16 = 32 bu/h

[ekkylli@taito-login4 ~]$ ls -l
total 24
drwx------ 3 ekkylli csc 4096 Mar 11  2015 appl_taito
drwxr-xr-x 2 ekkylli csc 4096 Mar 28 11:42 data part III
drwx------ 2 ekkylli csc 4096 Mar 21 13:19 Desktop
drwx------ 3 ekkylli csc 4096 Jan 19  2016 intel
drwxr-xr-x 6 ekkylli csc 4096 Mar 21 15:36 R_spatial_2017
drwx------ 3 ekkylli csc 4096 Oct 19  2016 test
[ekkylli@taito-login4 ~]$
```

NoMachine Remote Desktop

- Client connection between user and gateway
- Good performance even with slow network
- Ssh from gateway to server (fast if local)
- Persistent connection
- Suspendable
  - Continue later at another location
- Read the instructions…
  - ssh-key, keyboard layout, mac specific workarounds, …
- Choose an application or server to use (right click)



Ascii terminal

- Open a terminal on your workstation (right click on backround or select from menu), then in terminal:

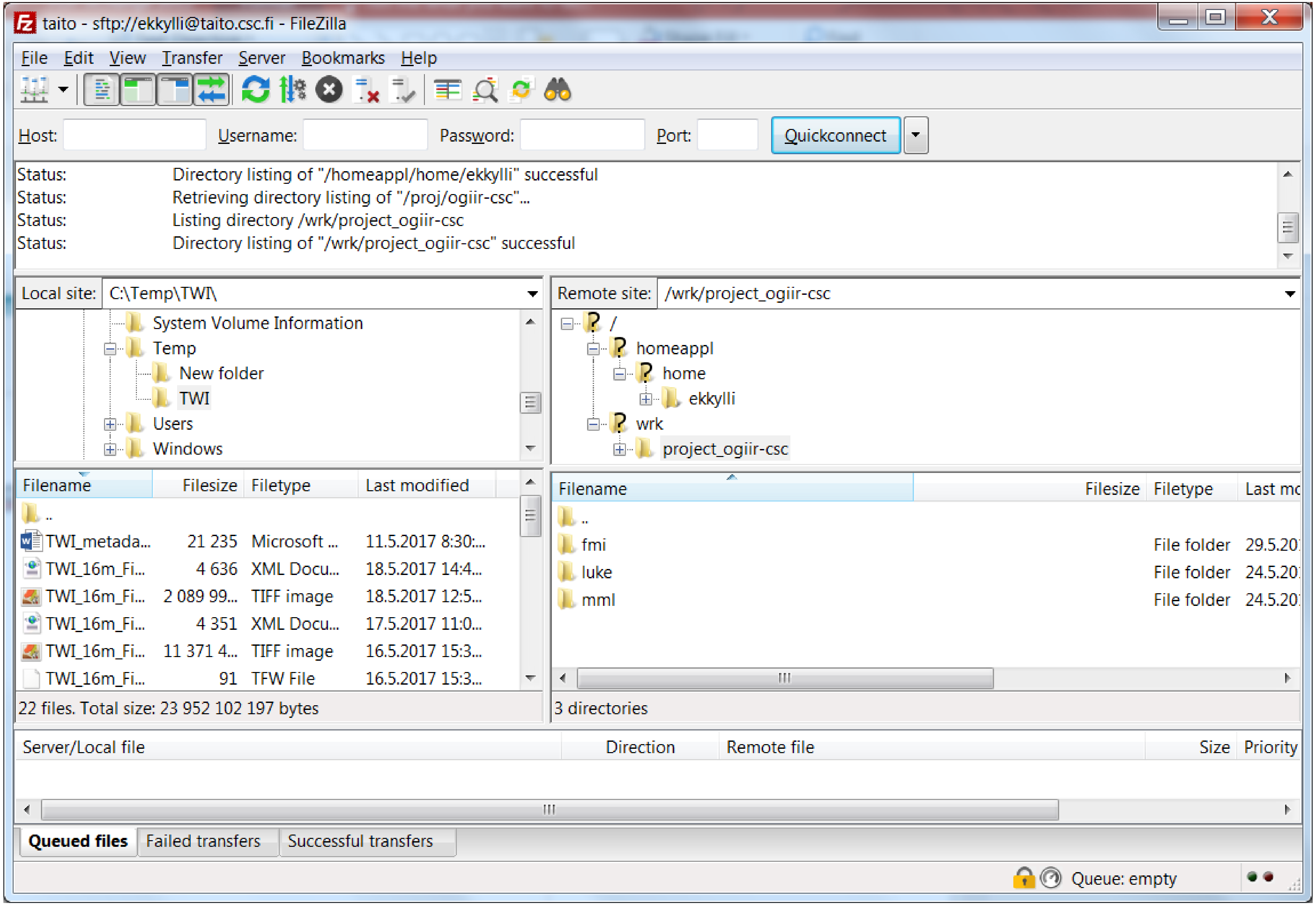
```
$ ssh user@taito.csc.fi
 (man in the middle?)
$ ls
$ hostname
$ gnuplot
$ plot sin(x)   [fails!]
```

NoMachine

- Open *NoMachine* client
- Select nxkajaani.csc.fi
- Insert your *username* and *password*
- (accept help screens)
- Right click on the background, choose taito from menu
- Give your CSC password

```
$ ls
$ hostname
$ ...
```

FileZilla



taito - sftp://ekkylli@taito.csc.fi - FileZilla

Host: Username: Password: Port:

```
Status:        Directory listing of "/homeappl/home/ekkylli" successful
Status:        Retrieving directory listing of "/proj/ogiir-csc"...
Status:        Listing directory /wrk/project_ogiir-csc
Status:        Directory listing of "/wrk/project_ogiir-csc" successful
```

Local site: C:\Temp\TWI\

```
System Volume Information
Temp
    New folder
    TWI
Users
Windows
```

Remote site: /wrk/project_ogiir-csc

```
/
    homeappl
        home
            ekkylli
    wrk
        project_ogiir-csc
```

Filename	Filesize	Filetype	Last modified
..			
TWI_metada...	21 235	Microsoft ...	11.5.2017 8:30:...
TWI_16m_Fi...	4 636	XML Docu...	18.5.2017 14:4...
TWI_16m_Fi...	2 089 99...	TIFF image	18.5.2017 12:5...
TWI_16m_Fi...	4 351	XML Docu...	17.5.2017 11:0...
TWI_16m_Fi...	11 371 4...	TIFF image	16.5.2017 15:3...
TWI_16m_Fi...	91	TFW File	16.5.2017 15:3...

22 files. Total size: 23 952 102 197 bytes

Filename	Filesize	Filetype	Last mo
..			
fmi		File folder	29.5.20
luke		File folder	24.5.20
mml		File folder	24.5.20

3 directories

Server/Local file	Direction	Remote file	Size	Priority

Access with scientific software

- Some software can be configured to use CSC servers directly, e.g.
  - TMolex, ADF, Maestro, Discovery Studio, Matlab
- The GUIs can be used to create and submit jobs directly to the Taito queueing system
  - Look at the instructions on the software pages

Finnish Grid and Cloud Infrastructure - FGCI

- Distributed computing capacity
- 9 universities + CSC
- Requires a certificate
- Lots of preinstalled software
- Access with ARC –client
- From your own computer or Taito

Oulu
Kuopio
Joensuu
Jyväskylä
Tampere
Lappeenranta
Turku
Espoo
Åbo
Helsinki

```
arcproxy
arcsub jobscript.xrsl
arcget gsiftp://usva.fgi.csc.fi:2811/jobs/12465133890987654
```

- FGCI guide

FGCI FINNISH GRID AND CLOUD INFRASTRUCTURE

Pouta Cloud service

Do I need…

Different operating system and software stack than CSC's systems?

To run web services?

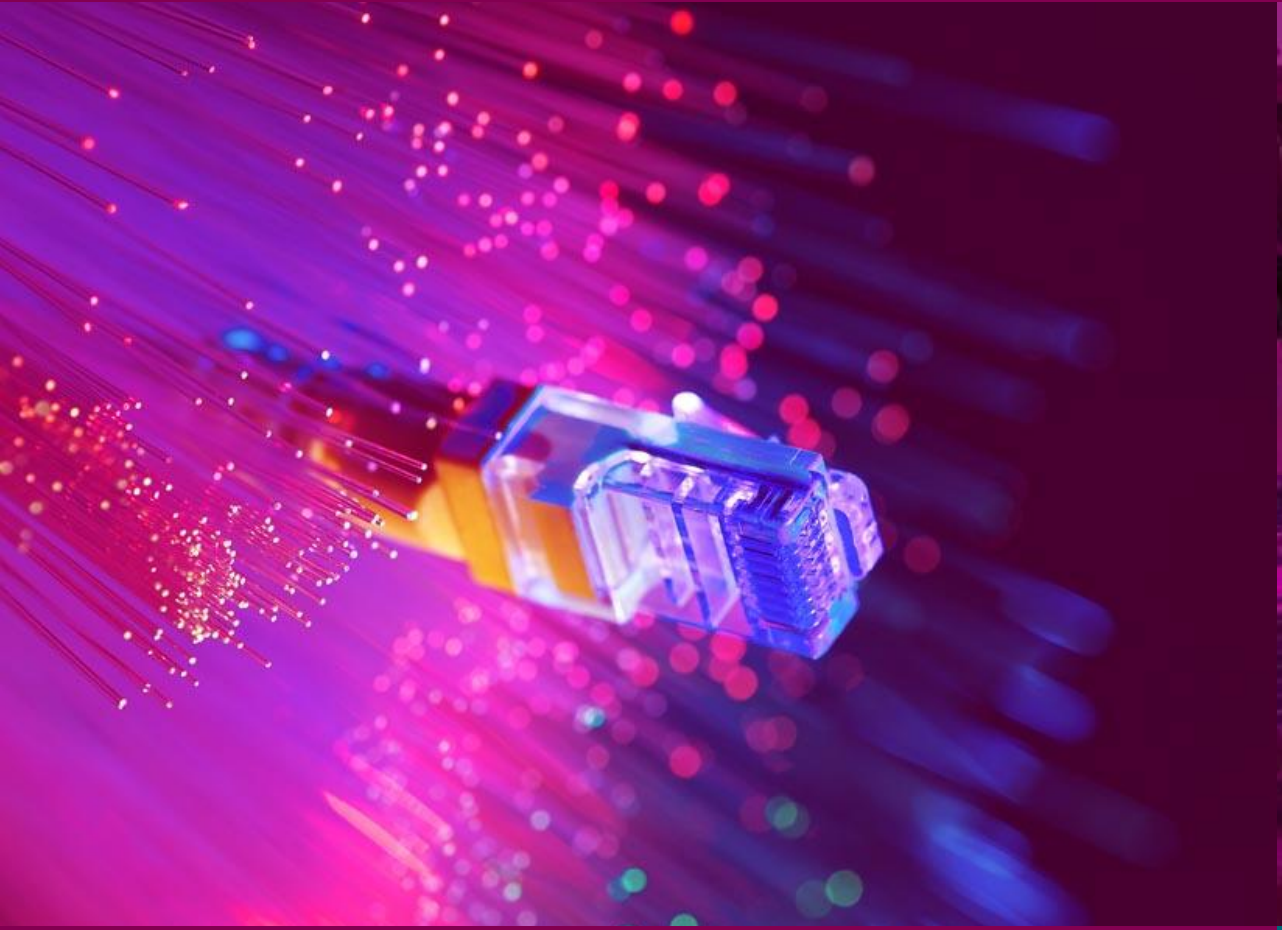
To extend my local computing resources?

→ http://research.csc.fi/cloud-computing

Summary: How to access resources at CSC

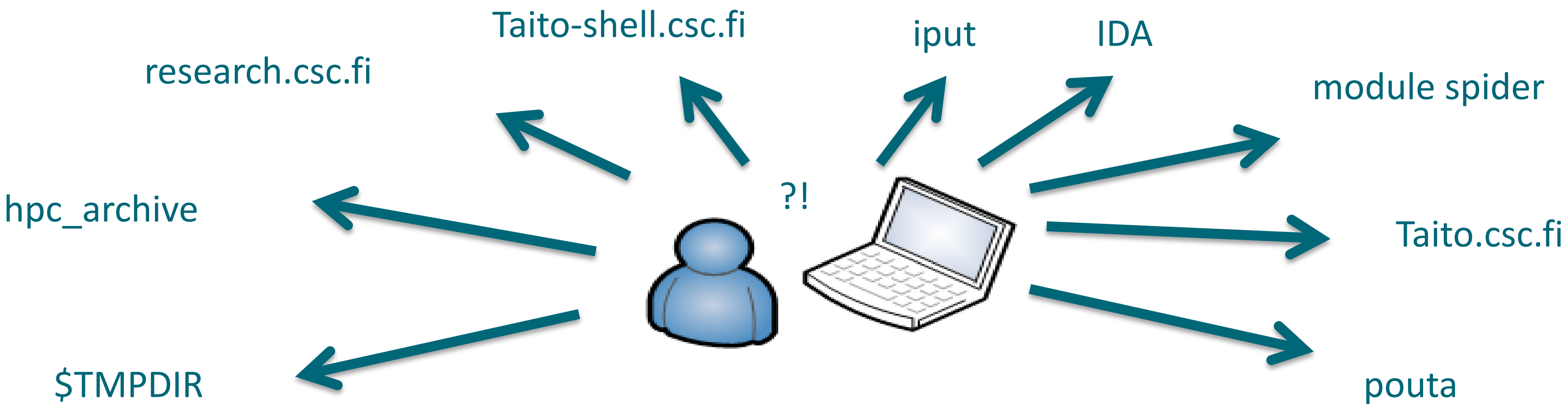
- Ssh terminal connection to CSC (Putty + X-term emulator for win)
- Installation at your own computer, license from CSC
  - Materials Studio, Discovery Studio, Ansys, ...
- GUI at your own computer, computation at CSC (ssh pipe)
  - Tmolex, ADFgui, Discovery Studio
- GUI at your own computer, input files to CSC by hand, jobs launched from command prompt
- Scientist's User Interface (www based) sui.csc.fi
  - File manager, certificates, terminal, software distribution, ...
- ARC (Nordugrid) middleware to run jobs in FGCI
- NoMachine Remote desktop (etätyöpöytä)
  - Client installed at your own computer, working with graphics at CSC
- Cloud services: pouta.csc.fi
  - Lots of freedom/flexibility and hence some administration and configuration work

CSC Computing Environment



Learning target

- Know how to choose right server (resource)
- Know where to put your files
- Know how to setup and use preinstalled software



On Clusters and Supercomputers (1/2)

- Shared Memory Parallel (SMP):

  All processors access (more or less) the same memory

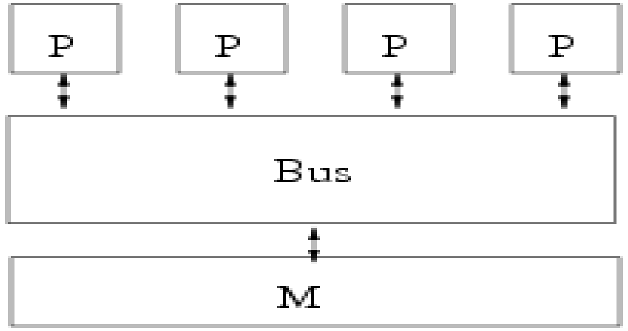
  Within node

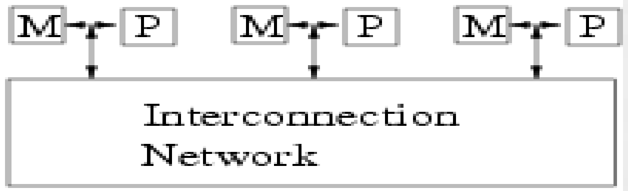
- Distributed Memory:

  Processes access their own memory

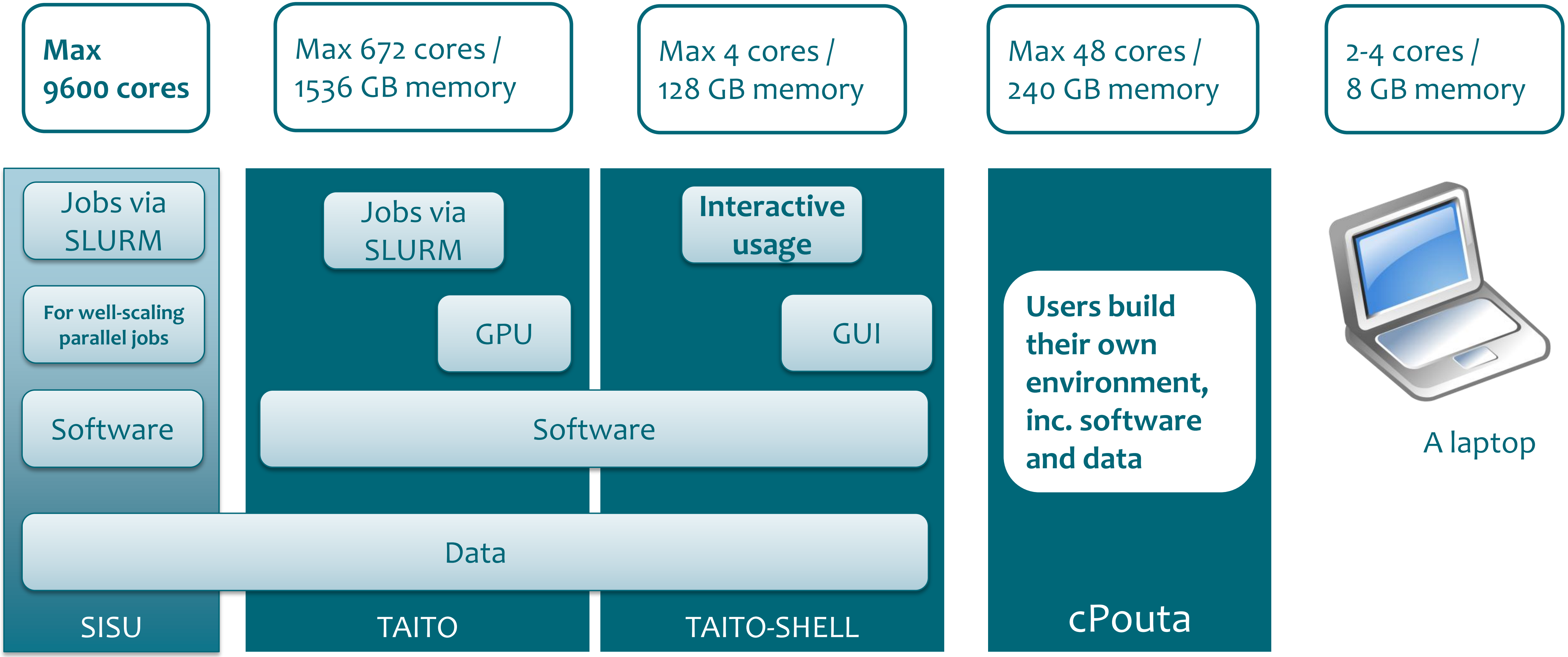
  Interconnection network for exchange

  Between nodes





CSC HPC resources



Max 9600 cores

Max 672 cores / 1536 GB memory

Max 4 cores / 128 GB memory

Max 48 cores / 240 GB memory

2-4 cores / 8 GB memory

Jobs via SLURM

For well-scaling parallel jobs

Software

SISU

Jobs via SLURM

GPU

TAITO

Interactive usage

GUI

TAITO-SHELL

Software

Data

Users build their own environment, inc. software and data

cPouta

A laptop

On Clusters and Supercomputers (2/2)

- A cluster is a connection of separate units (nodes) via a fast network

- All larger CSC platforms (Sisu, Taito, FGCI) are clusters in a general sense

Server use profiles

- Taito (HP)
- Serial and parallel upto 448/672 cores
- Huge memory jobs
- Lots of preinstalled software

- Taito-shell (HP)
- Interactive jobs
- Very long jobs
- Auto queue, shared resources

- Sisu (Cray XE40)
- Parallel from 72 up to thousands of cores
- Scaling tests 1008+

- cPouta (HP) Cloud
- Serial and parallel upto 16 cores

- FGCI (Dell/HP)
- Serial and parallel (16)

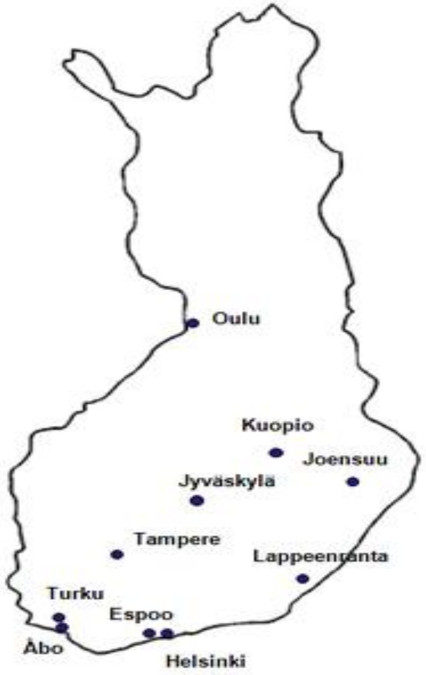
Main Computing capacity: Sisu,Taito FGCI

	Sisu (Phase 2)	Taito (Phase 2)	FGCI
Availability	2014-	2015-	2016-
CPU	Intel Haswell and Sandy Bridge, 2 x 12 and 2 x 8 cores, 2.6 GHz, Xeon E5-2690v3 and E5-2670		Intel Xeon, 2 x 6 cores, 2.7 GHZ, X5650 and 4x12 Intel Xeon CPU E7-4830v3 @2.1GHz
Interconnect	Aries	FDR IB	QDR IB
Cores	40512	9768+9216	7308+3600
RAM/node	64 GB	64/128/256/ 1536 GB	128/256/512 GB
Tflops	1688	515	218
GPU nodes	-	50	8
Disc space	4 PB	4 PB	1+ PB

FGCI – The Finnish Grid and Cloud Infrastructure

- Consortium of 9 Finnish Universities and CSC
- Infrastructure consists of 7368+3600 cores and 100 GPU cards (+ Taito)
- Accessed via ARC middleware
- Submit jobs from taito/own workstation
- Preinstalled software
- More information: FGCI guide

FGCI FINNISH GRID AND CLOUD INFRASTRUCTURE



Sample ARC job description file

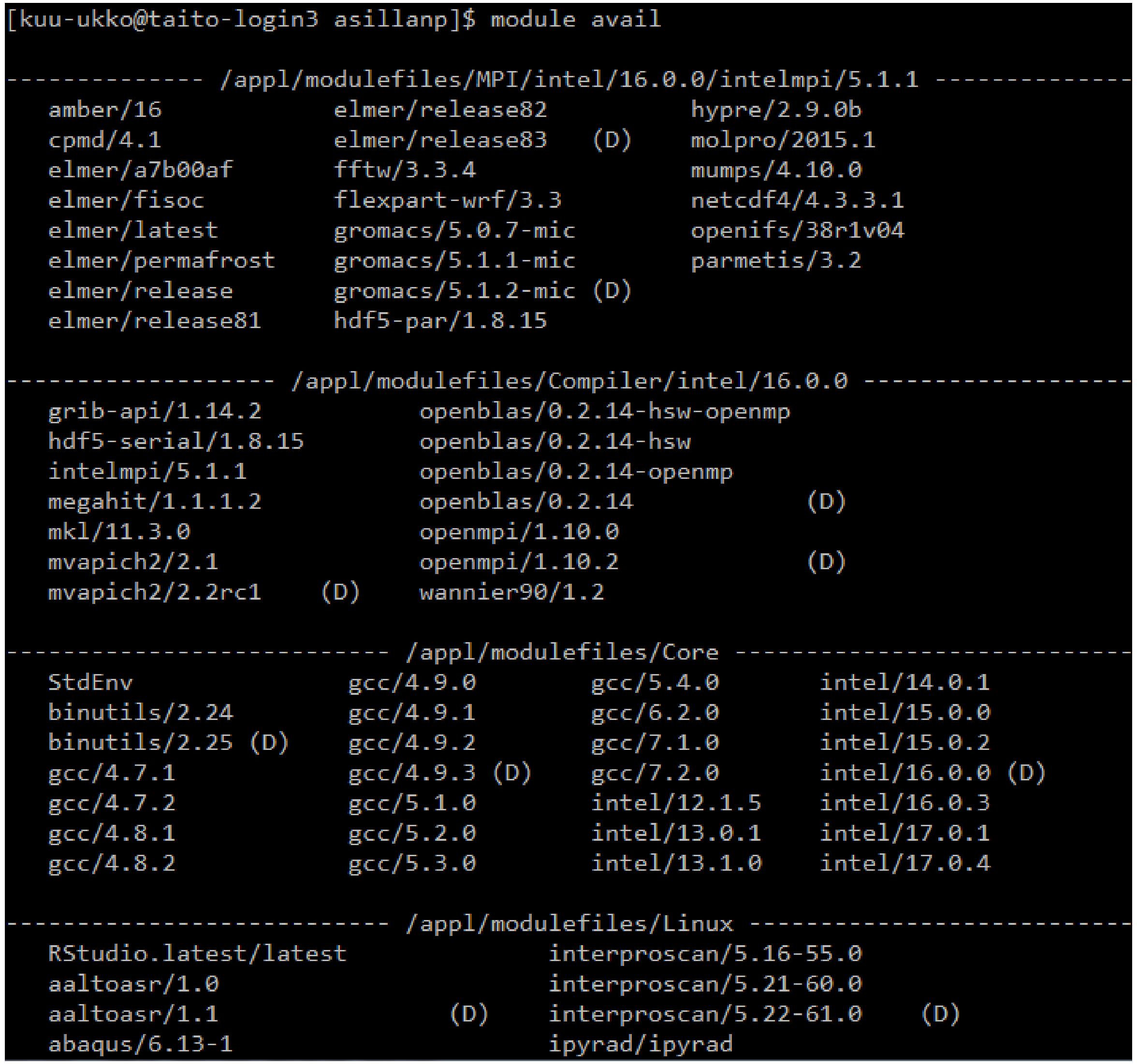
```
&
(executable=runbwa.sh)
(jobname=bwa_1)
(stdout=std.out)
(stderr=std.err)
(gmlog=gridlog_1)
(walltime=24h)
(memory=8000)
(disk=4000)
(runtimeenvironment>="APPS/BIO/BWA_0.6.1")
(inputfiles=
( "query.fastq" "query.fastq" )
( "genome.fa" "genome.fa" )
)
(outputfiles=
 ( "output.sam" "output.sam" )
)
```

IaaS cloud services

- https://research.csc.fi/cloud-computing
  - Infrastructure as a Service (IaaS) type of cloud
  - OpenStack cloud middleware for management
  - The Virtual Machines are admistrated by the user
  - **cPouta**
    - The cPouta service allows customers to run virtual machines connected to the Internet.
    - PI of a project can apply for access in SUI
    - Youtube videos on how to start a VM in cPouta
  - **ePouta**
    - The cloud service combines virtual computational resources with the customers' own resources using a dedicated light path or MPLS connection.
    - Designed for secure data handling

The module system

- Tool to set up your environment
  - Load libraries, adjust path, set environment variables
  - Needed on a server with hundreds of applications and several compilers etc.
- Slightly different on Taito vs. Sisu
- Used both in interactive and batch jobs



```
[kuu-ukko@taito-login3 asillanp]$ module avail

-------------- /appl/modulefiles/MPI/intel/16.0.0/intelmpi/5.1.1 --------------
   amber/16                elmer/release82          hypre/2.9.0b
   cpmd/4.1                elmer/release83    (D)   molpro/2015.1
   elmer/a7b00af           fftw/3.3.4               mumps/4.10.0
   elmer/fisoc             flexpart-wrf/3.3         netcdf4/4.3.3.1
   elmer/latest            gromacs/5.0.7-mic        openifs/38r1v04
   elmer/permafrost        gromacs/5.1.1-mic        parmetis/3.2
   elmer/release           gromacs/5.1.2-mic (D)
   elmer/release81         hdf5-par/1.8.15

------------------- /appl/modulefiles/Compiler/intel/16.0.0 -------------------
   grib-api/1.14.2              openblas/0.2.14-hsw-openmp
   hdf5-serial/1.8.15           openblas/0.2.14-hsw
   intelmpi/5.1.1               openblas/0.2.14-openmp
   megahit/1.1.1.2              openblas/0.2.14              (D)
   mkl/11.3.0                   openmpi/1.10.0
   mvapich2/2.1                 openmpi/1.10.2               (D)
   mvapich2/2.2rc1     (D)      wannier90/1.2

--------------------------- /appl/modulefiles/Core ----------------------------
   StdEnv                   gcc/4.9.0         gcc/5.4.0       intel/14.0.1
   binutils/2.24            gcc/4.9.1         gcc/6.2.0       intel/15.0.0
   binutils/2.25 (D)        gcc/4.9.2         gcc/7.1.0       intel/15.0.2
   gcc/4.7.1                gcc/4.9.3 (D)     gcc/7.2.0       intel/16.0.0 (D)
   gcc/4.7.2                gcc/5.1.0         intel/12.1.5    intel/16.0.3
   gcc/4.8.1                gcc/5.2.0         intel/13.0.1    intel/17.0.1
   gcc/4.8.2                gcc/5.3.0         intel/13.1.0    intel/17.0.4

--------------------------- /appl/modulefiles/Linux ---------------------------
   RStudio.latest/latest                interproscan/5.16-55.0
   aaltoasr/1.0                         interproscan/5.21-60.0
   aaltoasr/1.1                  (D)    interproscan/5.22-61.0    (D)
   abaqus/6.13-1                        ipyrad/ipyrad
```

Typical module commands

`module avail`	shows available modules (compatible modules in taito)
`module spider`	shows all available modules in taito
`module list`	shows currently loaded modules
`module load` ***<name>***	loads module <name> (default version)
`module load` ***<name/version>***	loads module <name/version>
`module switch` ***<name1> <name2>***	unloads module name1 and loads module name2
`module purge`	unloads all loaded modules

Taito has ”meta-modules” named *e.g.* gromacs-env, which will load all necessary modules needed to run gromacs.

Module example

- Show compatible modules on Taito

```
$ module avail
```

- Initialize R and RStudio statistics packages

```
$ module load r-env
$ module load rstudio
```

- Start RStudio using the command

```
$ rstudio
```

- It's better to run the GUI (and calculations) on a compute node (jobs that have used 1h of CPU on the login node will be killed automatically)
- For interactive work, use taito-shell.csc.fi

Simple plotting in R

```
> a=seq(0,10,by=0.1)
> plot(a,cos(a))
```

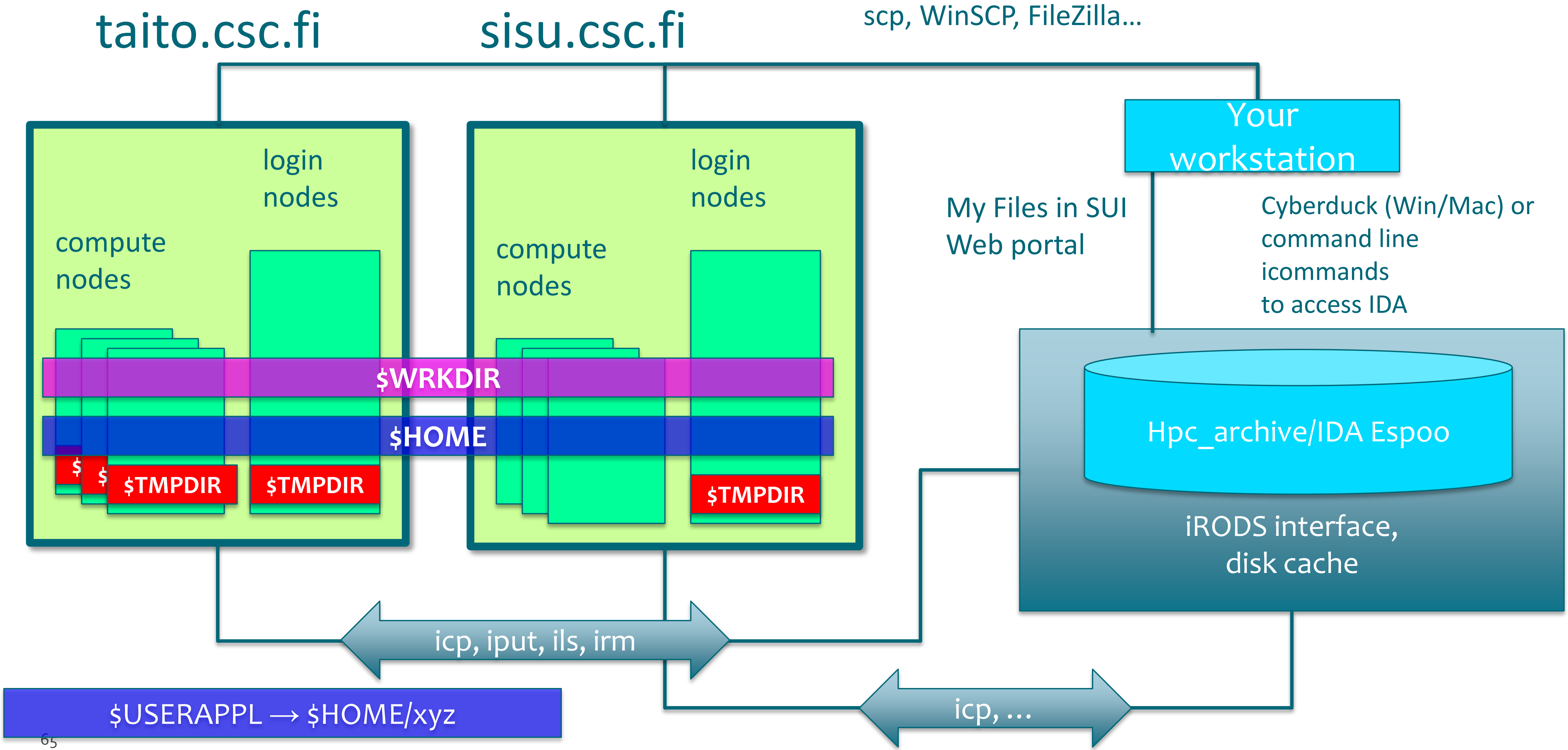
Directories at CSC Environment (1)

https://research.csc.fi/data-environment

Directory or storage area	Intended use	Default quota/user	Storage time	Backup
$HOME [1]	Initialization scripts, source codes, small data files. Not for running programs or research data.	50 GB	Permanent	Yes
$USERAPPL [1]	Users' own application software.	50 GB	Permanent	Yes
$WRKDIR [1]	Temporary data storage.	5 TB	90 days	No
$WRKDIR/DONOTREMOVE	Temporary data storage.	Incl. in above	Permanent	No
$TMPDIR [3]	Temporary users' files.	-	~2 days	No
Project [1]	Common storage for project members. A project can consist of one or more user accounts.	On request	Permanent	No
HPC Archive [2]	Long term storage.	2 TB	Permanent	Yes
IDA [2]	Storage and sharing of stable data.	On request	Permanent	No, multiple storage copies

[1]: Lustre parallel ([3]:local) file system in Kajaani [2]: iRODS storage system in Espoo

Directories at CSC Environment (2)



taito.csc.fi

sisu.csc.fi

scp, WinSCP, FileZilla…

Your workstation

login nodes

compute nodes

login nodes

compute nodes

$WRKDIR

$HOME

$TMPDIR

$TMPDIR

$TMPDIR

My Files in SUI Web portal

Cyberduck (Win/Mac) or command line icommands to access IDA

Hpc_archive/IDA Espoo

iRODS interface, disk cache

icp, iput, ils, irm

icp, …

$USERAPPL → $HOME/xyz

Storage: hard disks - 4 PB on DDN (Lustre), Sisu and Taito

- **`$USERAPPL`**: *put your own applications here*
  - /homeappl/home/username/app_taito
  - /homeappl/home/username/app_sisu
- **`/tmp`** (Taito, ~2 TB) to be used for *e.g. compiling codes on the login node or taito-shell*
- **`$TMPDIR`** on compute nodes: *for scratch files (accessed with **`$TMPDIR`** in batch script)*
- **`$HOME`** for configuration files and misc. smallish storage. If full, gives strange errors (X-graphics etc.)
- **`$WRKDIR`** for large data and during calculations. Avoid lots of small files. Files older than 90 days are deleted. No backup.
- **`$WRKDIR/DONOTREMOVE`** old files not deleted from here – don't **copy** files here, but **move** if you want to keep them (or hpc_archive)

Storage: disks and tape

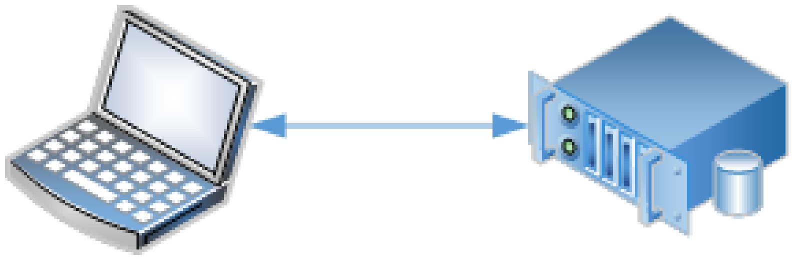
- IDA Storage Service
  - Common storage for project members
  - Storage for non-sensitive stable research data (*e.g.* provides persistent identifiers, automatic checksums)
  - Enables public sharing of data on the internet
  - Usage via SUI, command line or file transfer program
  - Quota available from universities, universities of applied sciences and Academy of Finland
  - Apply on the web http://openscience.fi/becoming-an-ida-user

- **hpc_archive Service**
  - Tape (+ disk cache)
  - Default long term storage
  - Access with i-commands from Sisu/Taito

hpc_archive/IDA interface at CSC

Some iRODS commands

`iput` ***file***	copy *file* to hpc_archive/IDA
`iget` ***file***	copy *file* from …/IDA
`ils`	list the current IDA directory
`icd` ***dir***	change the IDA directory
`irm` ***file***	remove *file* from IDA
`imv` ***file file***	move/rename *file* inside IDA
`imkdir` ***foo***	create a directory *foo* to IDA
`iinit`	Initialize your IDA account
`ipwd`	show current directory in IDA



IDA uses some different commands. See http://openscience.fi/ida-commands

Moving files, best practices

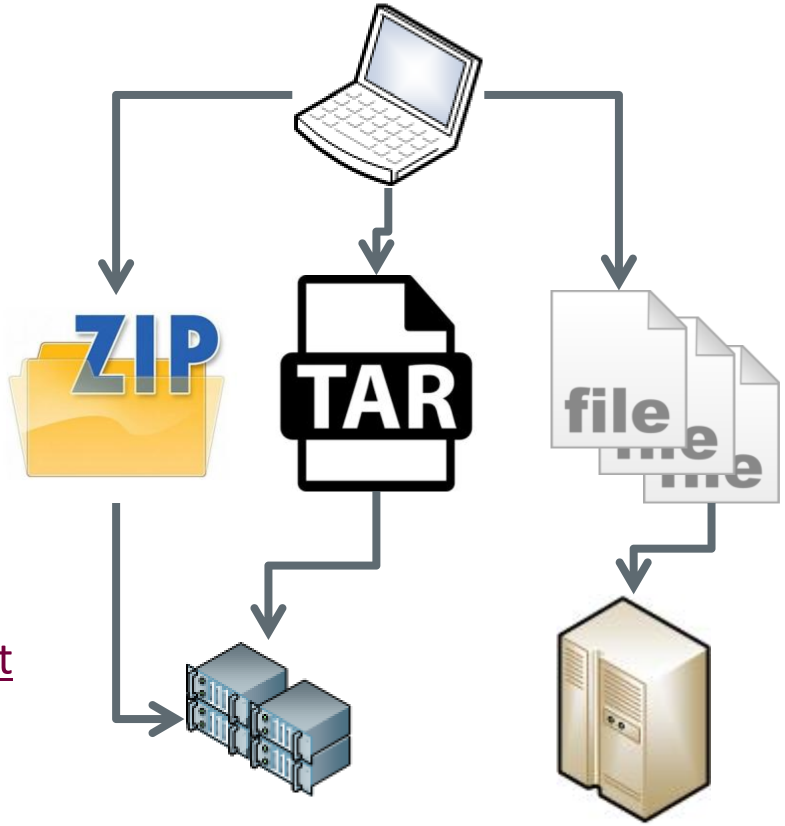
- rsync, not scp (when lots of/big files), *zip & tar first*

```
$ rsync -P username@taito-login3.csc.fi:/tmp/huge.tar.gz .
```

space!

- Funet FileSender (max 50 GB [1GB as an attachment? No!])
  - *https://filesender.funet.fi*
  - Files can be downloaded also with **`wget`**
- iRODS, batch-like process, staging
- IDA: http://openscience.fi/ida
- CSC can help to tune e.g. TCP/IP parameters
- FUNET backbone 100 Gbit/s
- Webinar Recording on Data Transfer

https://research.csc.fi/csc-guide-moving-data-between-csc-and-local-environment



Data transfer speed

	50 Mb / s	25 Mb / s	5 Mb / s
1 kb	0,00002 s	0,00001 s	0,0002 s
1 Mb	0,02 s	0,01 s	0,2
1 Gb	20 s	40 s	3,5 min
1 Tb	5,5 h	11 h	2d 7h

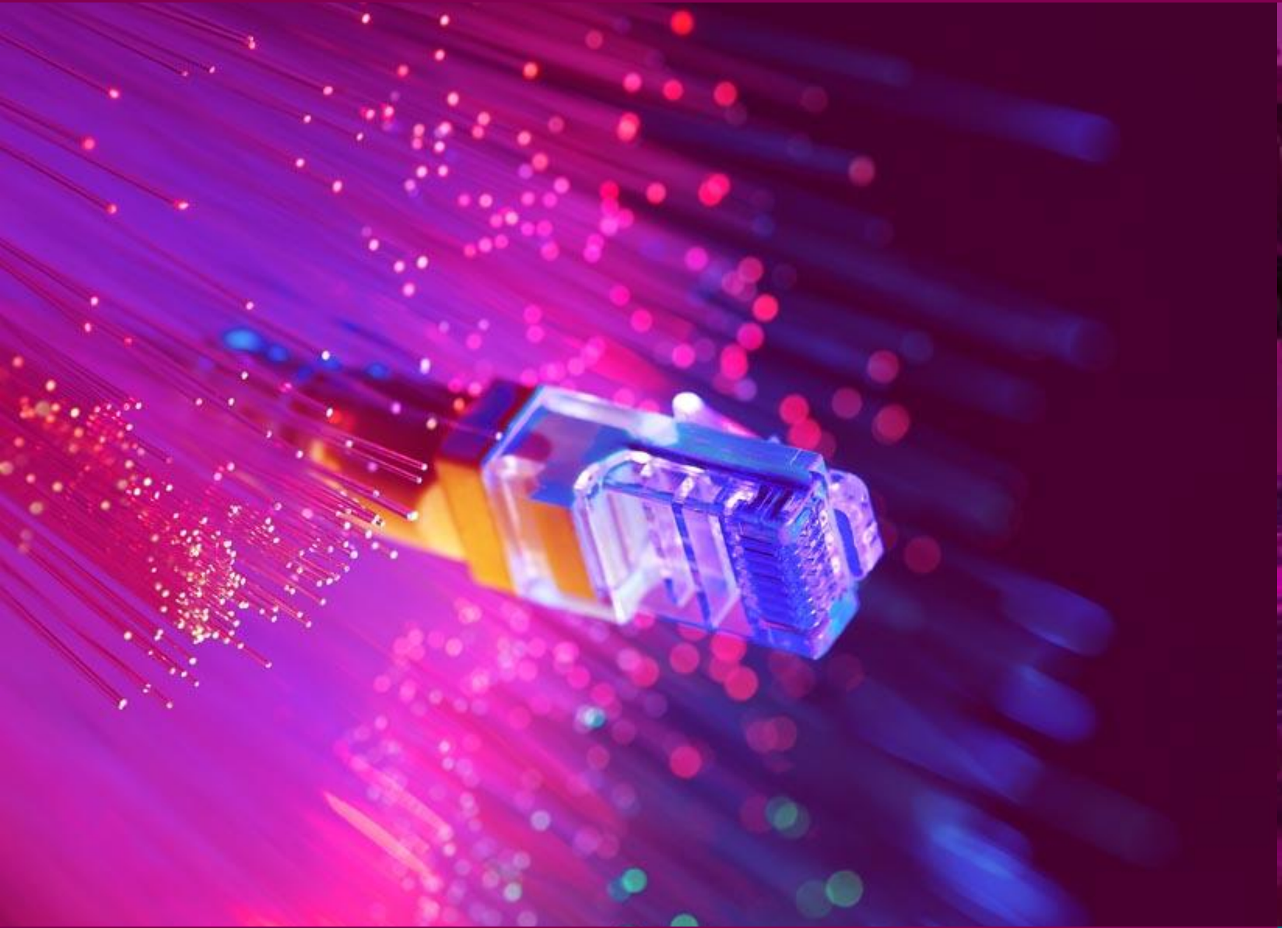
- 50 Mb/s often the realistic fast speed
- 25 Mb/s good normal speed
- 5 Mb/s realistic speed in mobile network

- For transfers taking hours or more, consider doing it in steps so that an interrupted transfer can be continued (rsync, FileZilla, avoid using just one gigantic file)
- Transfering lots (thousands) of small files (few kb) takes longer than fewer (few MB) but bigger files

Learning targets achieved?

- How to choose right server (resource)?
- How to setup and use preinstalled software/libraries/compilers?
- Where to put your files?

Running jobs at CSC

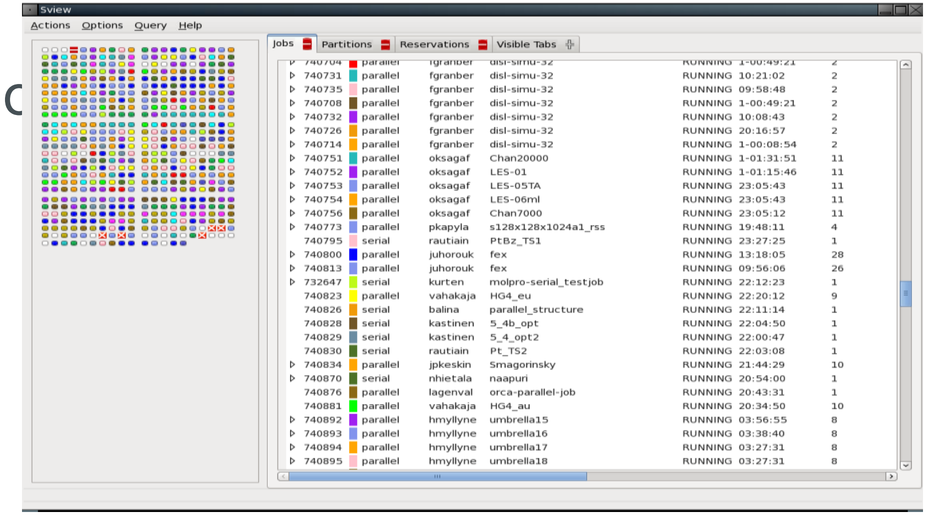


Batch jobs learning target

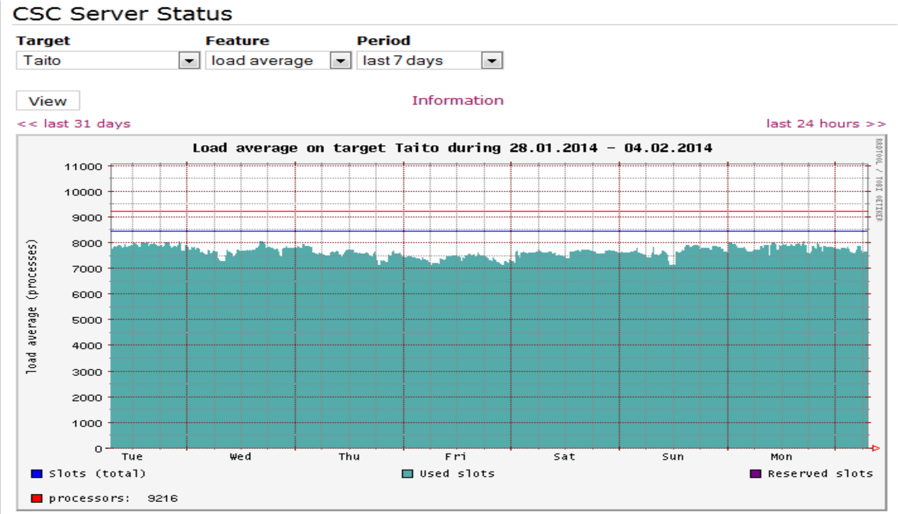
- Benefits of batch jobs for compute intensive jobs
  - Difference of login and compute node
  - Difference of interactive jobs (taito-shell) and batch jobs
- How to submit and monitor jobs
- Batch script contents *i.e.* resource requirements
- How to learn resource requirements of own jobs
- What is saldo [billing units]
- Be aware of batch script wizard in SUI
- Submit first job(s)
- Learn to read the the manual

What is a batch system?

- Optimizes resource usage by filling the server with jc
- Cores, memory, disk, length, …
- Jobs to run are chosen based on their priority
- Priority increases with queuing time
- Priority decreases with recently used resources
- Short jobs with little memory and cores queue the le
- CSC uses SLURM (Simple Linux Utility for Resource Management)



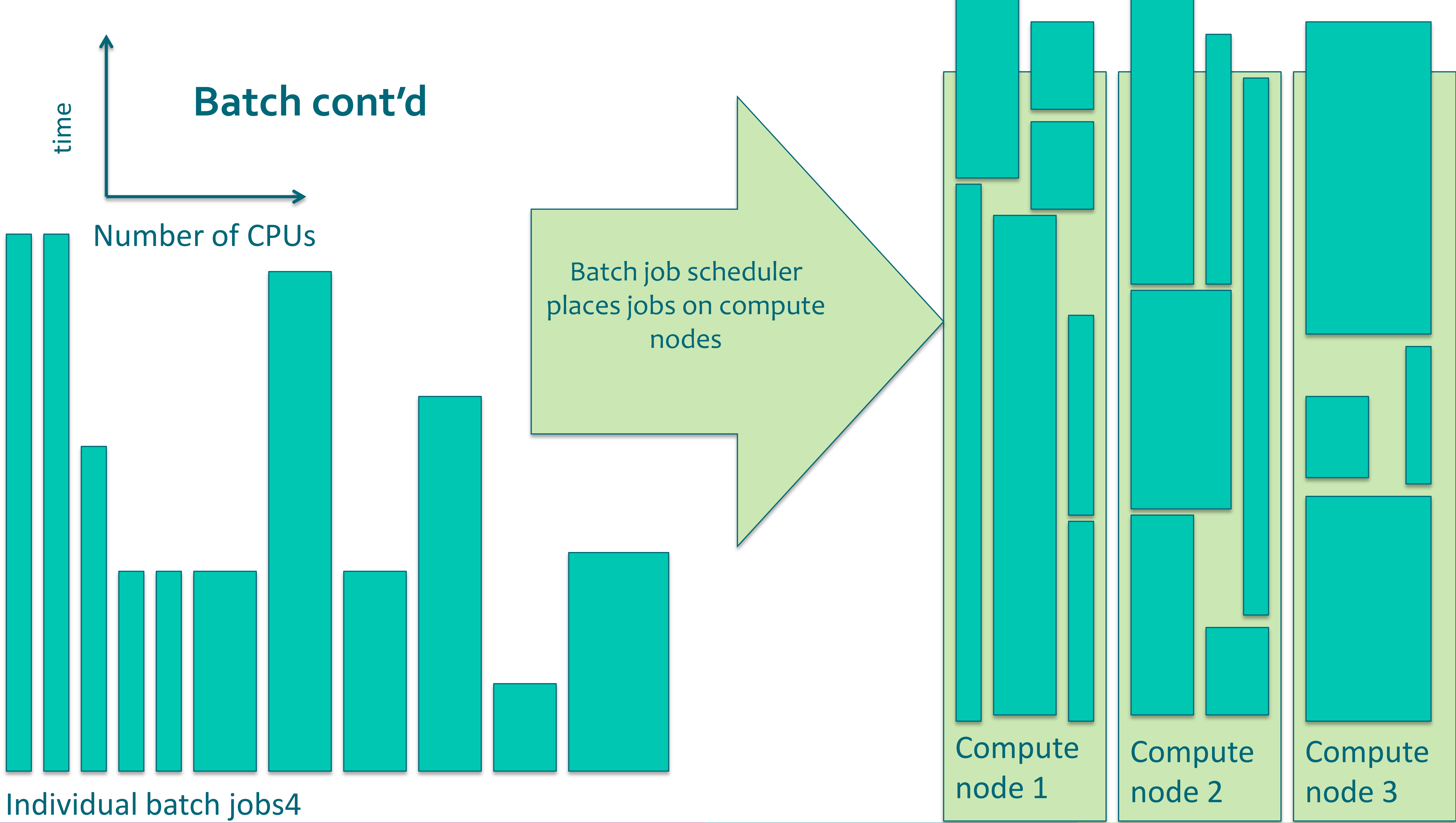
```
▷ 740704  parallel  fgranber  disl-simu-32             RUNNING  1-00:49:21   2
▷ 740731  parallel  fgranber  disl-simu-32             RUNNING  10:21:02     2
▷ 740735  parallel  fgranber  disl-simu-32             RUNNING  09:58:48     2
▷ 740708  parallel  fgranber  disl-simu-32             RUNNING  1-00:49:21   2
▷ 740732  parallel  fgranber  disl-simu-32             RUNNING  10:08:43     2
▷ 740726  parallel  fgranber  disl-simu-32             RUNNING  20:16:57     2
▷ 740714  parallel  fgranber  disl-simu-32             RUNNING  1-00:08:54   2
▷ 740751  parallel  oksagaf   Chan20000                RUNNING  1-01:31:51   11
▷ 740752  parallel  oksagaf   LES-01                   RUNNING  1-01:15:46   11
▷ 740753  parallel  oksagaf   LES-05TA                 RUNNING  23:05:43     11
▷ 740754  parallel  oksagaf   LES-06ml                 RUNNING  23:05:43     11
▷ 740756  parallel  oksagaf   Chan7000                 RUNNING  23:05:12     11
▷ 740773  parallel  pkapyla   s128x128x1024a1_rss      RUNNING  19:48:11     4
  740795  serial    rautiain  PtBz_TS1                 RUNNING  23:27:25     1
▷ 740800  parallel  juhorouk  fex                      RUNNING  13:18:05     28
▷ 740813  parallel  juhorouk  fex                      RUNNING  09:56:06     26
▷ 732647  serial    kurten    molpro-serial_testjob    RUNNING  22:12:23     1
  740823  parallel  vahakaja  HG4_eu                   RUNNING  22:20:12     9
  740826  serial    balina    parallel_structure       RUNNING  22:11:14     1
  740828  serial    kastinen  5_4b_opt                 RUNNING  22:04:50     1
  740829  serial    kastinen  5_4_opt2                 RUNNING  22:00:47     1
  740830  serial    rautiain  Pt_TS2                   RUNNING  22:03:08     1
▷ 740834  parallel  jpkeskin  Smagorinsky              RUNNING  21:44:29     10
▷ 740870  serial    nhietala  naapuri                  RUNNING  20:54:00     1
  740876  parallel  lagenval  orca-parallel-job        RUNNING  20:43:31     1
  740881  parallel  vahakaja  HG4_au                   RUNNING  20:34:50     10
▷ 740892  parallel  hmyllyne  umbrella15               RUNNING  03:56:55     8
▷ 740893  parallel  hmyllyne  umbrella16               RUNNING  03:38:40     8
▷ 740894  parallel  hmyllyne  umbrella17               RUNNING  03:27:31     8
▷ 740895  parallel  hmyllyne  umbrella18               RUNNING  03:27:31     8
```



CSC Server Status

Target: Taito
Feature: load average
Period: last 7 days

Batch cont’d



time

Number of CPUs

Batch job scheduler places jobs on compute nodes

Compute node 1

Compute node 2

Compute node 3

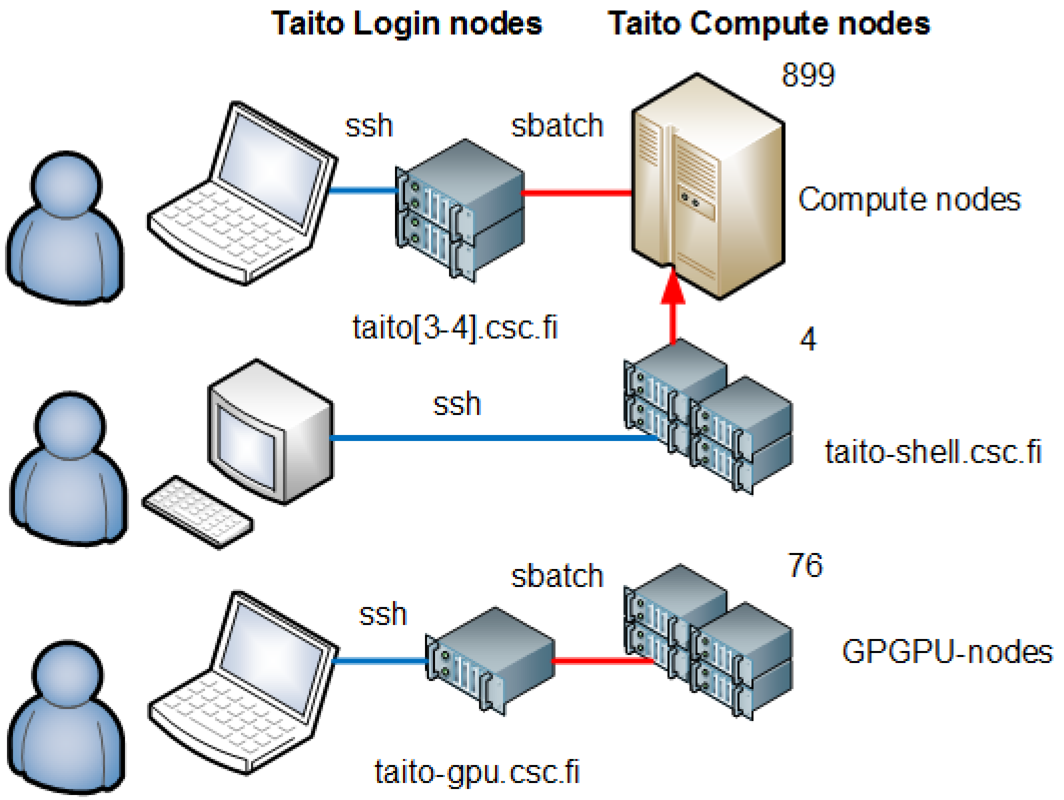
Individual batch jobs4

Compute nodes are used via the queuing system

```
$ sbatch job_script.sh
```

```
$ ./my_prog &
```

```
$ sbatch job_script.sh
```



Batch job overview

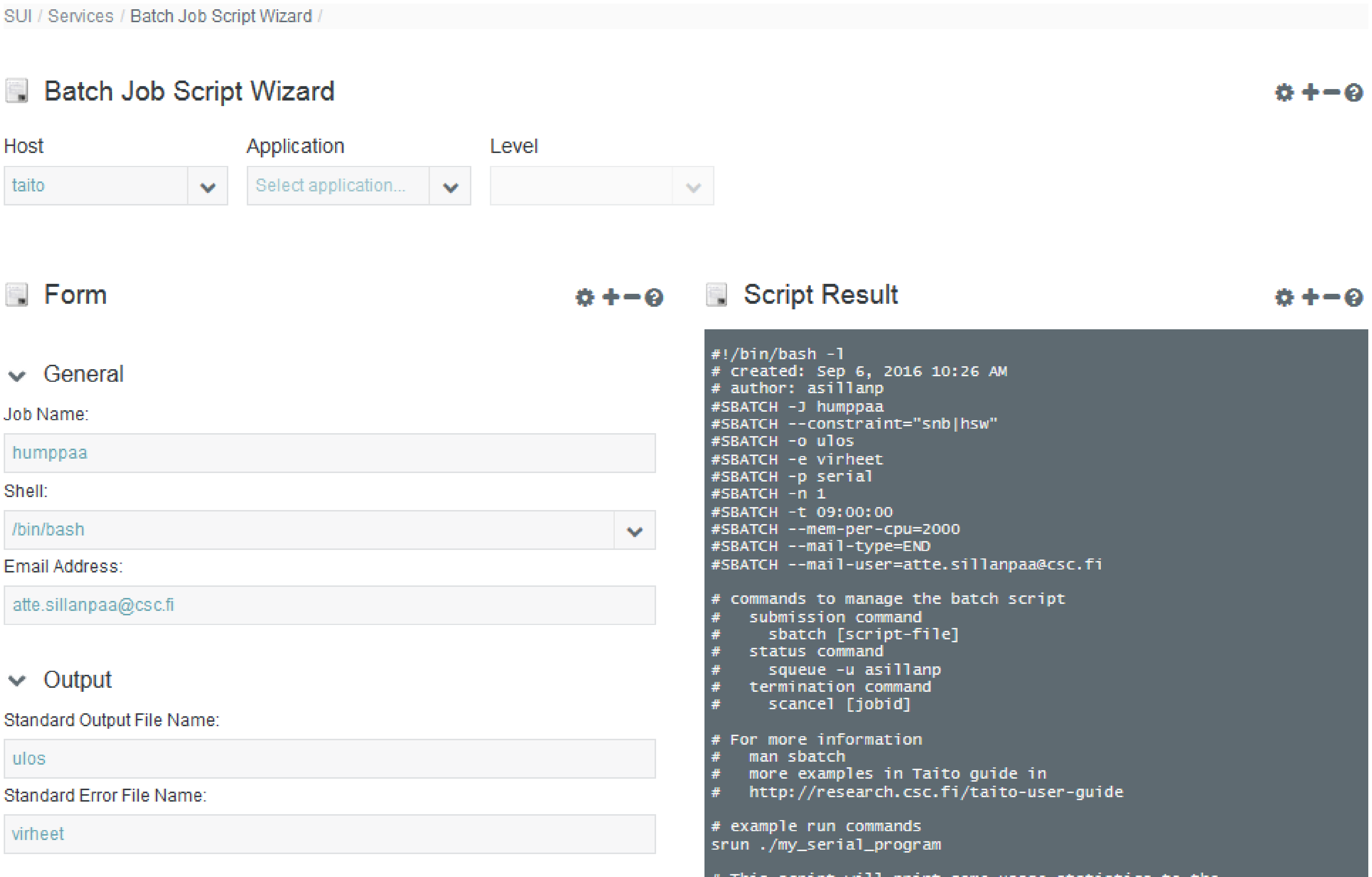
- Steps for running a batch job

1. Write a batch job script
   - This script will tell SLURM what resources are needed and then specifies your job
   - Script details depend on server, check CSC Guides or software page!
   - You can use the Batch Job Script Wizard in Scientist's User Interface:
   - https://sui.csc.fi/group/sui/batch-job-script-wizard

2. Make sure all the necessary files are in $WRKDIR
   - $HOME has limited space
   - **Login node** $TMPDIR is not available on compute nodes

3. Submit your job
   - `$ sbatch myscript`

Batch Job Script wizard in Scientist's User Interface



SUI / Services / Batch Job Script Wizard /

Batch Job Script Wizard

Host: taito

Application: Select application...

Level:

Form

General

Job Name: humppaa

Shell: /bin/bash

Email Address: atte.sillanpaa@csc.fi

Output

Standard Output File Name: ulos

Standard Error File Name: virheet

Script Result

```
#!/bin/bash -l
# created: Sep 6, 2016 10:26 AM
# author: asillanp
#SBATCH -J humppaa
#SBATCH --constraint="snb|hsw"
#SBATCH -o ulos
#SBATCH -e virheet
#SBATCH -p serial
#SBATCH -n 1
#SBATCH -t 09:00:00
#SBATCH --mem-per-cpu=2000
#SBATCH --mail-type=END
#SBATCH --mail-user=atte.sillanpaa@csc.fi

# commands to manage the batch script
#   submission command
#     sbatch [script-file]
#   status command
#     squeue -u asillanp
#   termination command
#     scancel [jobid]

# For more information
#   man sbatch
#   more examples in Taito guide in
#   http://research.csc.fi/taito-user-guide

# example run commands
srun ./my_serial_program

# This script will print some usage statistics to the
```

Batch jobs: what and why

- User has to specify necessary resources
  - Can be added to the batch job script or given as command line options for sbatch (or a combination of script and command line options)

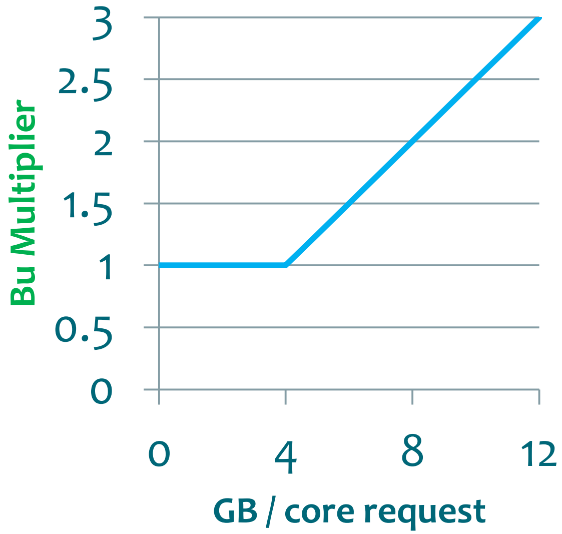
- Resources need to be adequate for the job
  - Too small memory reservation will cause the job to fail
  - When the time reservation ends, the job will be terminated whether finished or not

- But: Requested resources can affect the time the job spends in the queue
  - Especially memory reservation (and perhaps requested time)
  - Using more cores does not always make the job run faster - check!
  - Don’t request extra ”just in case” (time is less critical than memory wrt this)

- So: Realistic resource requests give best results
  - Not always easy to know beforehand
  - Usually best to try with smaller tasks first and check the used resources
  - You can check what was actually used with the **seff** command

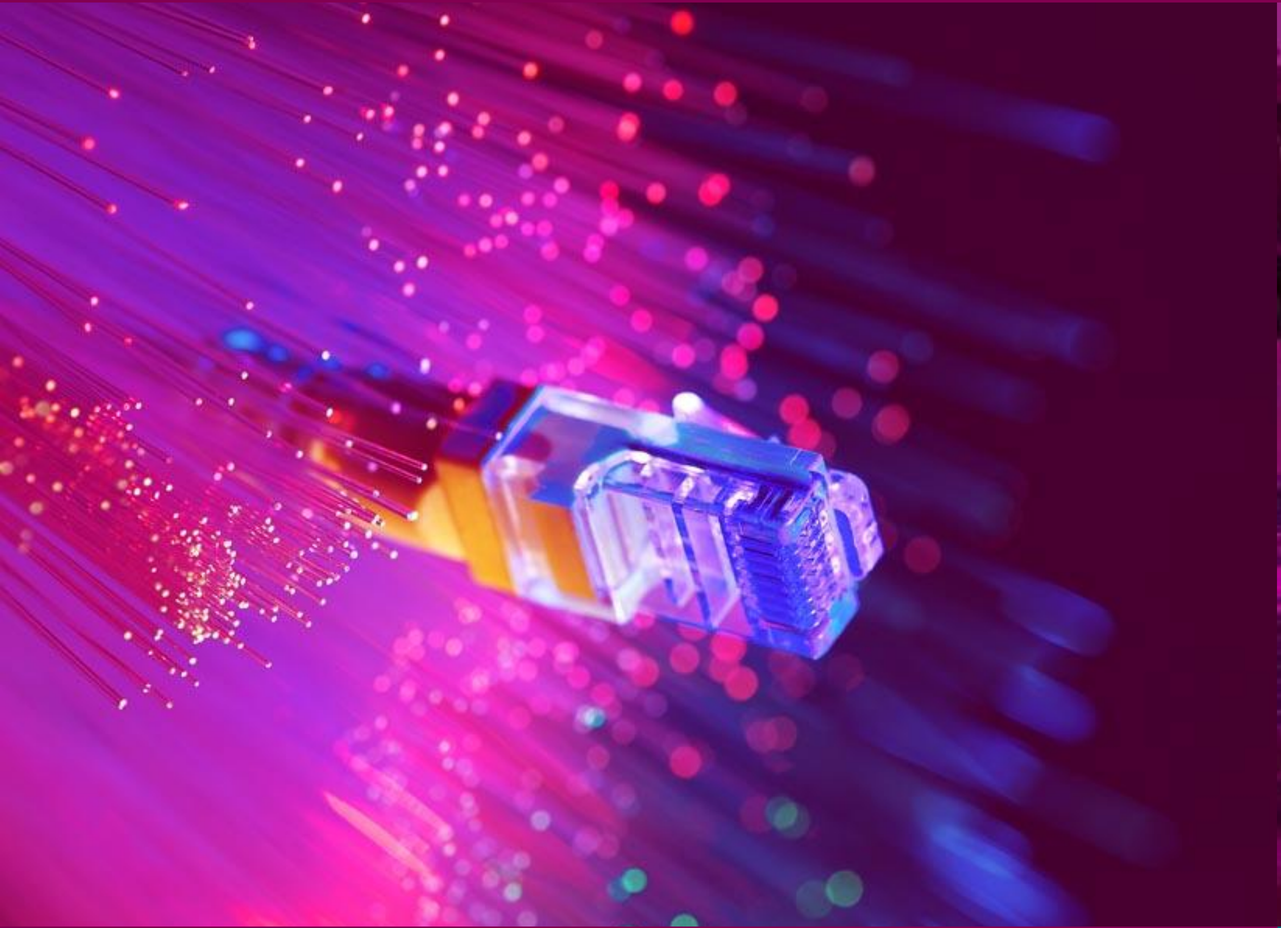
Saldo and billing units

- All jobs consume saldo
- https://research.csc.fi/saldo
- One core hour of computing equals 2 billing units [bu] times a multiplier
- Jobs requesting less than 4GB of memory per core have a multiplier of 1
- Jobs requesting 4GB or more per core have a multiplier X/4, where X is the requested memory per core:
  - 5GB/core = 5/4x = 1.25x
  - 12GB/core = 12/4x = 3x
  - …
- Requested but not used computing time is not billed
- If saldo runs out, no new jobs are possible
- New saldo can be requested from SUI
- GPU resources have an additional multiplier



- Serial job (1 core), 0.5 GB/core of memory, requested 24 hours, used 5 hours → billed: 1*5*2*1=10 bu
- (failed) parallel job: requested 24 cores, 2GB/memory per core, actually used 6 cores (18 cores idle) total run time 10 hours → billed 24*10*2*1=480 bu
- Parallel job 3 cores, 5 GB/core, 10 hours → billed: 3*10*2*5/4=75 bu

SLURM batch script contents



Example serial batch job script on Taito

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH –p serial
#SBATCH --constraint=snb

module load someprog
srun someprog -option1 -option2
```

```
#!/bin/bash -l
```

- Tells the computer this is a script that should be run using bash shell
- Everything starting with "**#SBATCH**" is passed on to the batch job system (Slurm)
- Everything (else) starting with "**#**" is considered a comment
- Everything else is executed as a command

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

`#SBATCH -J myjob`

- Sets the name of the job

- When listing jobs *e.g.* with **`squeue`**, only 8 first characters of job name are displayed.

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

```
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
```

- Option **–e** sets the name of the file where possible error messages (stderr) are written
- Option **–o** sets the name of the file where the standard output (stdout) is written
- When running the program interactively these would be written to the command promt
- What gets written to stderr and stderr depends on the program. If you are unfamiliar with the program, it’s always safest to capture both
- `%j` is replaced with the job id number in the actual file name

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

```
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
```

- Option **`--mail-type=END`** = send email when the job finishes
- Option **`--mail-user`** = your email address.
- If these are selected you get a email message when the job is done. This message also has a resource usage summary that can help in setting batch script parameters in the future.
- To see actually used resources try also: **`sacct –l –j <jobid>`** (more on this later)

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

`#SBATCH --mem-per-cpu=4000`

- The amount of memory reserved for the job in MB
  - 1000 MB = 1 GB
- Memory is reserved per-core basis even for shared memory (OpenMP) jobs
  - For those jobs it is better to ask memory *per job*:
  - **`--mem=1000`**
- Keep in mind the specifications for the nodes. Jobs with impossible requests are rejected (try **`squeue`** after submit)
- If you reserve too little memory the job will be killed (you will see a corresponding error in the output)
- If you reserve too much memory your job will spend much longer in queue and potentially waste resources (idle cores)

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

```
#SBATCH -t 02:00:00
```

TIP: If you're unsure of the syntax, use Batch job wizard in SUI

- Time reserved for the job in hh:mm:ss
- When the time runs out the job will be terminated!
- With longer reservations the job may queue longer
- Limit for normal serial jobs is 3d (72 h)
  - if you reserve longer time, choose "**`longrun`**" queue (limit 14d)
  - In the longrun queue you run at your own risk. If a batch job in that queue stops prematurely no compensation is given for lost cpu time
  - In longrun you likely queue for a longer time: shorter jobs and restarts are better (safer, more efficient)
- Default job length is 5 minutes → need to be set by yourself.

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

#SBATCH -n 1

- Number of cores to use. More than one means parallel.

- It’s also possible to control on how many **nodes** your job is distributed. Normally, this is not needed. By default use all cores in allocated nodes:
  - **`--ntasks-per-node=16 #(Sandy Bridge)`**
  - **`--ntasks-per-node=24 #(Haswell)`**

- Check documentation: http://research.csc.fi/software
  - There’s a lot of software that can only be run in serial

- OpenMP applications can only use cores in one node
  - For thread parallelization using 16 threads use (also) **`--cpus-per-task=16`**
  - This would start one task (**`-n 1`**) with 16 threads

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

#SBATCH -p serial

- The queue the job should be submitted to
- Queues are called ”partitions” in SLURM
- You can check the available queues with command

sinfo -l

```
[asillanp@taito-login4 ~]$ sinfo -l
Wed Jan 28 15:45:39 2015
PARTITION AVAIL  TIMELIMIT   JOB_SIZE ROOT    SHARE     GROUPS  NODES       STATE NODELIST
serial*      up 3-00:00:00          1    no       NO       all      1    draining c623
serial*      up 3-00:00:00          1    no       NO       all    101       mixed c[25,76-77,…
serial*      up 3-00:00:00          1    no       NO       all    593   allocated c[3-24,26-75,…
serial*      up 3-00:00:00          1    no       NO       all    226        idle c[211-213,…
parallel     up 3-00:00:00       1-28    no       NO       all      1    draining c623
parallel     up 3-00:00:00       1-28    no       NO       all    101       mixed c[25,76-77,…
parallel     up 3-00:00:00       1-28    no       NO       all    593   allocated c[3-24,26-75,…
parallel     up 3-00:00:00       1-28    no       NO       all    226        idle c[211-213,…
longrun      up 14-00:00:0          1    no       NO       all      1    draining c623
longrun      up 14-00:00:0          1    no       NO       all    101       mixed c[25,76-77,…
longrun      up 14-00:00:0          1    no       NO       all    587   allocated c[3-24,26-75,…
longrun      up 14-00:00:0          1    no       NO       all    226        idle c[211-213,…
test         up      30:00        1-2    no       NO       all      4        idle c[1-2,984-985]
hugemem      up 7-00:00:00          1    no       NO       all      2       mixed c[577-578]
```

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

`#SBATCH --constraint=snb`

- The job is run only in Sandy Bridge (snb) nodes
- The other option is Haswell node (hsw) or
  - **`#SBATCH --constraint=hsw`**
- Either that is free ”snb|hsw”
  - **`#SBATCH --constraint=”snb|hsw”`**

- Currently the default is to use *either* architecture in *serial* and *longrun* partitions
- Sandy Bridge in *test* and *parallel*
- A single job cannot use CPUs from both architectures, but SLURM will take care of this

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial
#SBATCH --constraint=snb

module load myprog
srun myprog -option1 -option2
```

```
module load myprog
srun myprog -option1 -option2
```

- Your commands
  - These define the actual job to performed: these commands are run on the compute node.
  - See application documentation for correct syntax
  - Some examples also from batch script wizard in SUI
- Remember to load modules if necessary
- By default the working directory is the directory where you submitted the job
  - If you include a **cd** command, make sure it points to correct directory
- Remember that input and output files should be in **$WRKDIR** (or in some case **$TMPDIR**)
- **$TMPDIR** contents are deleted after the job
- **srun** tells your program which cores to use. There are also exceptions…

```
#!/bin/bash -l
#SBATCH -J myjob
#SBATCH -e myjob_err_%j
#SBATCH -o myjob_output_%j
#SBATCH --mail-type=END
#SBATCH --mail-user=a.user@foo.net
#SBATCH --mem-per-cpu=4000
#SBATCH -t 02:00:00
#SBATCH -n 1
#SBATCH -p serial

module load myprog
srun myprog -option1 -option2
```

Most commonly used sbatch options

Slurm option	Description
--begin=*time*	defer job until HH:MM MM/DD/YY
-c, --cpus-per-task=*ncpus*	number of cpus required per task
-d, --dependency=*type:jobid*	defer job until condition on jobid is satisfied
-e, --error=*err*	file for batch script's standard error
--ntasks-per-node=*n*	number of tasks per node
-J, --job-name=*jobname*	name of job
--mail-type=*type*	notify on state change: BEGIN, END, FAIL or ALL
--mail-user=*user*	who to send email notification for job state changes
-n, --ntasks=*ntasks*	number of tasks to run
-N, --nodes=*N*	number of nodes on which to run
-o, --output=*out*	file for batch script's standard output
-t, --time=*minutes*	time limit in format hh:mm:ss
--mem-per-cpu=<number in MB>	maximum amount of real memory per allocated cpu (core) required by the job in megabytes
--mem=<number in MB>	maximum memory per node

SLURM:
Managing batch jobs in Taito

Submitting and cancelling jobs

- The script file is submitted with command

```
$ sbatch batch_job.file
```

- Job can be deleted with command

```
$ scancel <jobid>
```

Queues

- The job can be followed with command squeue:

```
$ squeue                      (shows all jobs in all queues)
$ squeue -p <partition>       (shows all jobs in single queue (partition))
$ squeue -u <username>        (shows all jobs for a single user)
$ squeue -j <jobid> -l        (status of a single job in long format)
```

- To estimate the start time of a job in queue

```
$ scontrol show job <jobid>
```

  row "StartTime=..." gives an *estimate* on the job start-up time, e.g.
  StartTime=2014-02-11T19:46:44 EndTime=Unknown

- **scontrol** will also show where your job is running
- If you add this to the end of your batch script, you'll get additional info to stdout about resource usage

```
seff $SLURM_JOBID
```

Examples of `seff` outputs

```
[erkki@taito]$ seff 52000797
Job ID: 52000797
Cluster: csc
User/Group: erkki/csc
State: COMPLETED (exit code 0)
Nodes: 4
Cores per node: 16
CPU Utilized: 00:37:22
CPU Efficiency: 87.58% of 00:42:40 core-
walltime
Memory Utilized: 7.53 GB (estimated
maximum)
Memory Efficiency: 3.21% of 234.38 GB
(58.59  GB/node)
```

Comments: only small part of memory used, could request less (now used the default 0.5GB/core), but for a parallel job like this, it’s better to request full nodes anyway.

```
[erkki@taito]$ seff 52000798_6
Job ID: 52000798
Array Job ID: 52000798_6
Cluster: csc
User/Group: erkki/csc
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 4
CPU Utilized: 00:24:09
CPU Efficiency: 98.17% of 00:24:36
core-walltime
Memory Utilized: 23.50 MB
Memory Efficiency: 1.15% of 2.00 GB
```

Comments: only small part of memory used, could request less (now used the default 0.5GB/core). Theoretically the job used only 23.5/4=6MB/core, but asking for e.g. 100MB/core (for safety) would likely make the job queue less.

Job logs

- Command **sacct** can be used to study *past* jobs
    - Useful when deciding proper resource requests

TIP: Check MaxRSS to see how much memory you need and avoid overbooking. See also command `seff JOBID`

`$ sacct`	Short format listing of jobs starting from midnight today
`$ sacct -l`	long format output
`$ sacct -j <jobid>`	information on single job
`$ sacct -S YYYY-MM-DD`	listing start date
`$ sacct -u <username>`	list only jobs submitted by username
`$ sacct -o`	list only named data fields, *e.g.*

```
$ sacct -o jobid,jobname,maxrss,reqmem,elapsed -j <jobid>
```

Available nodes/queues and limits

- You can check available resources per node in each queue:

```
 $ sjstat -c
-------------------------------------------------------------
Pool        Memory  Cpus  Total Usable   Free  Other Traits
-------------------------------------------------------------
serial*    258000Mb    24     10     10      5  hsw,haswell
serial*     64300Mb    16    502    502      9  snb,sandybridge
serial*    258000Mb    16     14     14      0  bigmem,snb,sandybridge
serial*    128600Mb    24    395    395      6  hsw,haswell
parallel   258000Mb    24     10     10      5  hsw,haswell
parallel    64300Mb    16    502    502      9  snb,sandybridge
parallel   258000Mb    16     14     14      0  bigmem,snb,sandybridge
parallel   128600Mb    24    395    395      6  hsw,haswell
longrun    258000Mb    16      8      8      0  bigmem,snb,sandybridge
longrun    258000Mb    24     10     10      5  hsw,haswell
longrun     64300Mb    16    502    502      9  snb,sandybridge
longrun    128600Mb    24    395    395      6  hsw,haswell
test        64300Mb    16      2      2      2  snb,sandybridge
test       128600Mb    24      2      2      2  hsw,haswell
hugemem   1551000Mb    32      2      2      0  bigmem,snb,sandybridge
hugemem   1551000Mb    40      4      4      1  bigmem,hsw,haswell,ssd
```

Most frequently used SLURM commands

Command	Description
`srun`	Run a parallel job.
`salloc`	Allocate resources for interactive use.
`sbatch`	Submit a job script to a queue.
`scancel`	Cancel jobs or job steps.
`sinfo`	View information about SLURM nodes and partitions.
`squeue`	View information about jobs located in the SLURM scheduling queue
`smap`	Graphically view information about SLURM jobs, partitions, and set configurations parameters
`sjstat`	Display statistics of jobs under control of SLURM (combines data from sinfo, squeue and scontrol)
`scontrol`	View SLURM configuration and state.
`sacct`	Displays accounting data for batch jobs.

Parallel jobs (1/2)

- Only applicable if your program supports parallel running
- Check application documentation for number of cores to use
  - Speed-up is often not linear (communication overhead)
  - Maximum number can be limited by the algorithms
  - Make sure (test it!) that using more cores speeds up your calculation
- Mainly two types: MPI jobs and shared memory (OpenMP) jobs
  - OpenMP jobs can be run only inside one node
    - All cores access same memory space
  - MPI jobs can span several nodes
    - Each core has its own memory space
  - In some cases you can use both: MPI between nodes and OpenMP within a node. Check the documentation of your program

Parallel jobs (2/2)

- Memory can be reserved either per core or per node
  - We recommend requesting memory per core
  - Don’t overallocate memory (check past jobs with **`seff JOBID`**)
  - If you reserve a complete node, you can also ask for all the memory
- Each server has different configuration so setting up parallel jobs in optimal way requires some thought
- See server guides for specifics: research.csc.fi/guides
  - Use Taito for large memory jobs
  - Sisu for massively parallel jobs
  - Check also the software specific pages for examples and detailed information: research.csc.fi/software

Array jobs (advanced usage)

- Best suited for running the same analysis for large number of files
- **`#SBATCH --array=1-100`**
- Defines to run 100 jobs, where a variable **`$SLURM_ARRAY_TASK_ID`** gets each number (**1,2,…100**) in turn as its value. This is then used to launch the actual job (e.g.
- **`$ srun myprog input_$SLURM_ARRAY_TASK_ID > output_$SLURM_ARRAY_TASK_ID)`**
- Thus this would run 100 jobs:

  ```
  srun myprog input_1 > output_1
  srun myprog input_2 > output_2
  …
  srun myprog input_100 > output_100
  ```

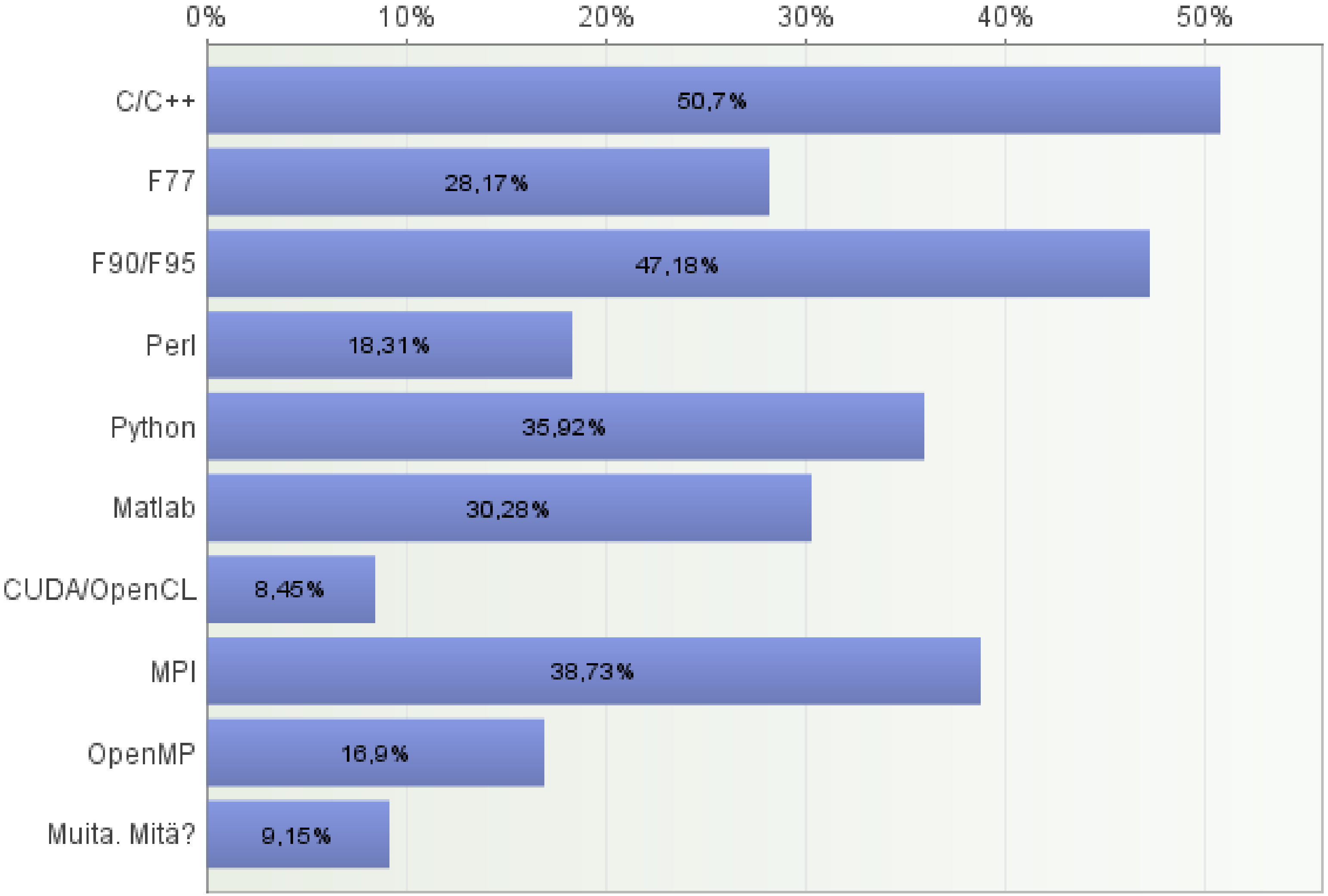
- For more information: research.csc.fi/taito-array-jobs

Compiling your program

What is a program?

- A program is a sequence of instructions understandable by a computer’s central processing unit (CPU) that indicates which operations the computer should perform
    - Ready-to-run programs are stored as executable files
    - An executable file is a file that has been converted from source code into machine code, by a specialized program called a compiler

Programming languages at supercomputers



gcc [source files] [-o prog]

- Compiles C source files into a program
- *-o* to give the name of the program, defaults to a.out
- -c to compile into .o -files

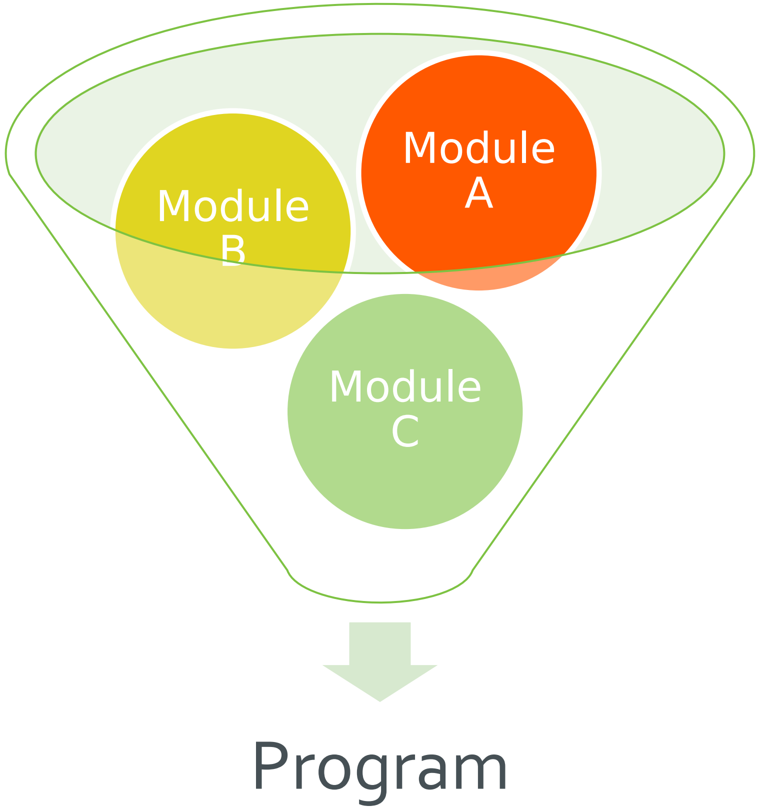
Compiling and installing programs

- For most programs, the three commands to compile and install in directory /home/user/programs are:

```
$ ./configure --prefix=/home/user/programs
$ make
$ make install
```

- `make` will be discussed in detail later today
- Destination for own programs in CSC computing environment: *$USERAPPL*

Why make?



- program separated into several files
- multiple inter-dependant modules
- compilation and linking becomes easily a nightmare
  - especially when developing the program!

Why make?

- when code has been modified, there are two approaches to compile the program:
  - re-compile everything → too slow
  - keep records and re-compile only what is needed → too much work

- `make` makes life easier by taking care of all the book keeping

Makefile

- defines:
  - work-flow(s) for producing target(s)
  - dependencies of each target
  - library paths, compiler flags etc.
- directives for conditional definitions etc.
- # starts a comment
- usually called `Makefile`
  - other choices: `makefile`, `GNUmakefile`

Basic syntax

RULE

name (usually filename)

list of files / rules

```
•target: dependencies
•    recipe
•    ...
```

commands to execute

example:

Note: use tabs instead of spaces to indent recipes!

```
•foo.o: foo.c bar.h   # module foo
•    cc -c foo.c

•clean:               # remove all
•    rm *.o
```

Basic syntax

- target
  - usually the file that is produced by the recipe
  - name of an action also commonly used
    - for example: clean, distclean
- dependencies
  - a list of (source) files needed by the recipe
  - may also be other targets
- recipe
  - a list of commands to execute to make target

Logic of make

- read general macro definitions etc.
- call the rule for target
    - check when dependencies were changed
    - if any of the dependencies have changed, the target is re-built according to the recipe
- dependencies may also be targets for other rules
    - in that case, make calls those rules

Simple example

```
hello: main.o sub1.o sub2.o sub3.o
   f90 -o hello main.o sub1.o sub2.o sub3.o
main.o: main.f90
   f90 -c main.f90
sub1.o: sub1.f90
   f90 -c sub1.f90
sub2.o: sub2.f90
   f90 -c sub2.f90
sub3.o: sub3.f90
   f90 -c sub3.f90
clean:
   rm hello main.o sub1.o sub2.o sub3.o
```

Which target?

- by default, the first target is called
  - 'hello' in the previous example
- target can be also specified when running `make`
  - `make target`
  - `make clean`
  - `make main.o`

Variables

- contain a string of text

  ```
  variable = value
  ```

- substituted in-place when referenced

  `$`(variable) → value

- sometimes also called macros
- shell variables are also available in the makefile
  - `$`(HOME), `$`(USER), ...

Two flavors of variables in GNU make

- recursive variables
  - defined as: `foo = bar`
  - expanded when referenced

- simple / constant variables
  - defined as: `foo := bar`
  - expanded when defined

```
foo = $(bar)
bar = $(ugh)
ugh = Huh?

$(foo) → Huh?

x := foo
y := $(x) bar
x = later

$(x) → later
$(y) → foo bar
```

Variables

- by convention variables are name in ALL-CAPS

- in the previous example we could have used a variable to store the names of all objects
  - `OBJ = main.o sub1.o sub2.o sub3.o`

Simple example revisited

```
OBJ = main.o sub1.o sub2.o sub3.o
hello: $(OBJ)
   f90 -o hello $(OBJ)
main.o: main.f90
   f90 -c main.f90
sub1.o: sub1.f90
   f90 -c sub1.f90
sub2.o: sub2.f90
   f90 -c sub2.f90
sub3.o: sub3.f90
   f90 -c sub3.f90
clean:
   rm hello $(OBJ)
```

Common variables

- some common variables
  - CC
  - CFLAGS
  - FC
  - FCFLAGS
  - LDFLAGS
  - OBJ
  - SRC

Special variables

- $@
  - name of the target

```
client: client.c
        $(CC) client.c -o $@
```

- $<
  - name of the first dependency

```
client: client.c
        $(CC) $< -o $@
```

Special variables

- $+
  - list of all dependencies
- $^
  - list of all dependencies (duplicates removed)
- $?
  - list of dependencies more recent than target

```
client: client.c
	$(CC) $+ -o $@
```

Special variables

- `$*`
  - common prefix shared by the target and the dependencies

```
client: client.c
        $(CC) -c -o $*.o $*.c
```

Special characters

- / continues a line
- # starts a comment
- @ executes a command quietly
  - by default, make echos all commands executed
  - this can be prevented by using @-sign at the beginning of the command

```
@echo ”quiet echo”

→ quiet echo
```

```
echo ”normal echo”

→    echo ”normal echo”
  normal echo
```

Special characters

- if there is an error executing a command, make stops
    - this can be prevented by using a – sign at the beginning of a command

```
clean:
        -rm hello
        -rm $(OBJ)
```

Implicit rules

- one can use special characters to define an implicit rule
- e.g. quite often target and dependencies share the name (different extensions)
  - define an implicit rule compiling an object file from a Fortran 90 source code file

```
%.o: %.f90
	$(F90) $(FFLAGS) -c -o $@ $<
```

Example revisited again

```
OBJ = main.o sub1.o sub2.o sub3.o

# implicit rule for compiling f90 files
%.o: %.f90
   f90 -c -o $@ $<

hello: $(OBJ)
   f90 -o hello $(OBJ)

clean:
   rm hello $(OBJ)
```

Built-in functions

- GNU make has also built-in functions
    - for a complete list see: www.gnu.org/software/make/manual/make.html#Functions

- strip, patsubst, sort, ...
- dir, suffix, basename, wildcard, ...

- general syntax
    - `$(function arguments)`

Command line options

- `-j` parallel execution
- `-n` dry-run

  shows the command, but does not execute them
- `-p` print defaults

  shows default rules and values for variables before execution
- `-s` silent-run

  do not print commands as they are executed

Command line options

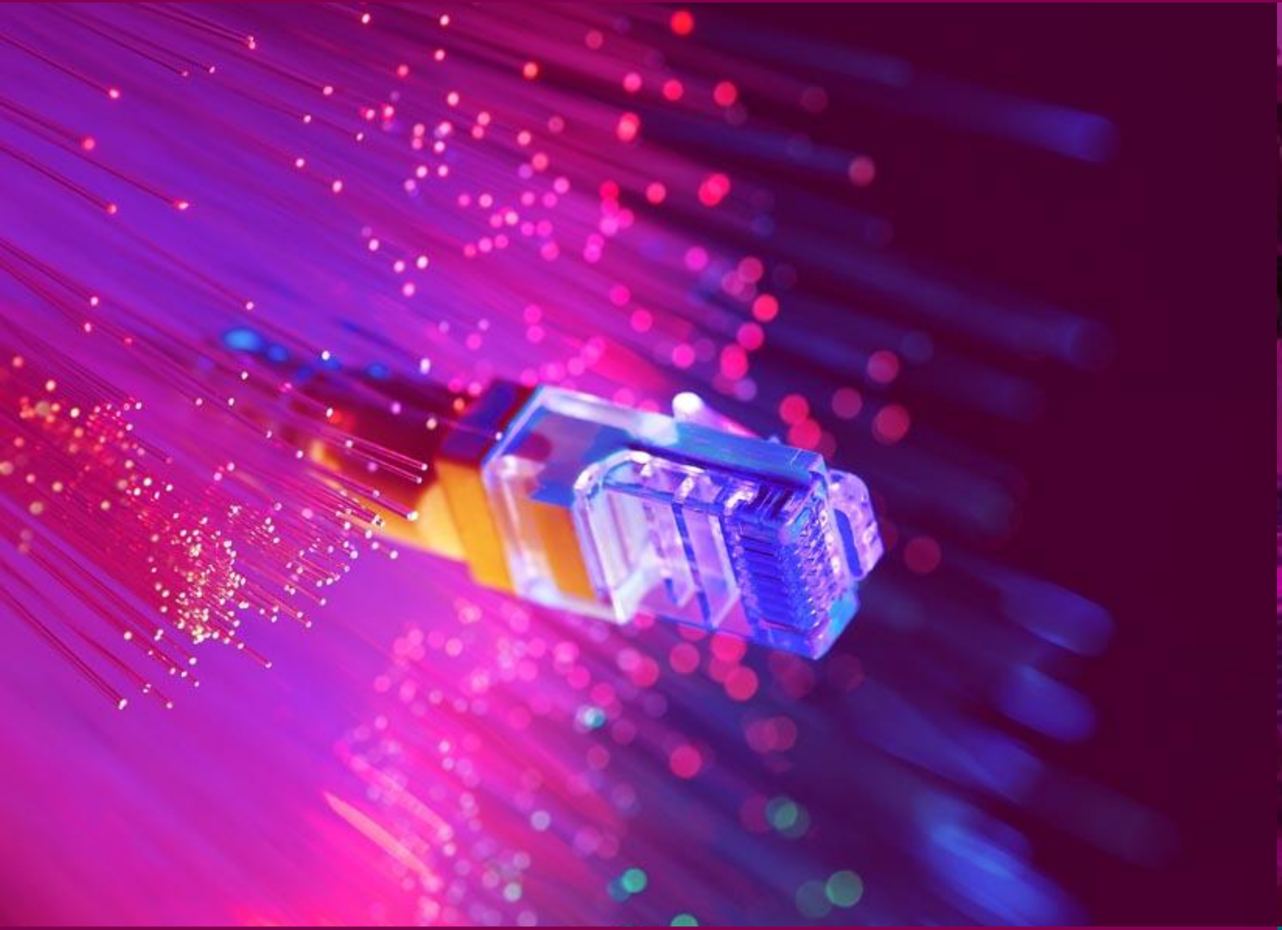
- variables can also be defined from the command line

```
make CC=gcc ”CFLAGS=-O3 –g” foobar
```

Complete example

```
SRC = main.f90 sub1.f90 sub2.f90 sub3.f90
OBJ = $(patsubst %.f90, %.o, $(SRC))
F90 = gfortran
FFLAGS =
DEST = bin

# implicit rule for compiling f90 files
%.o: %.f90
    $(F90) $(FFLAGS) -c -o $@ $<

hello: $(DEST)/hello

$(DEST)/hello: $(OBJ)
    $(F90) $(FFLAGS) -o $@ $(OBJ)

clean:
    -rm $(OBJ)
    -rm $(DEST)/hello

# extra dependencies
sub2.o: modules.o
```

Science services at CSC: a short introduction

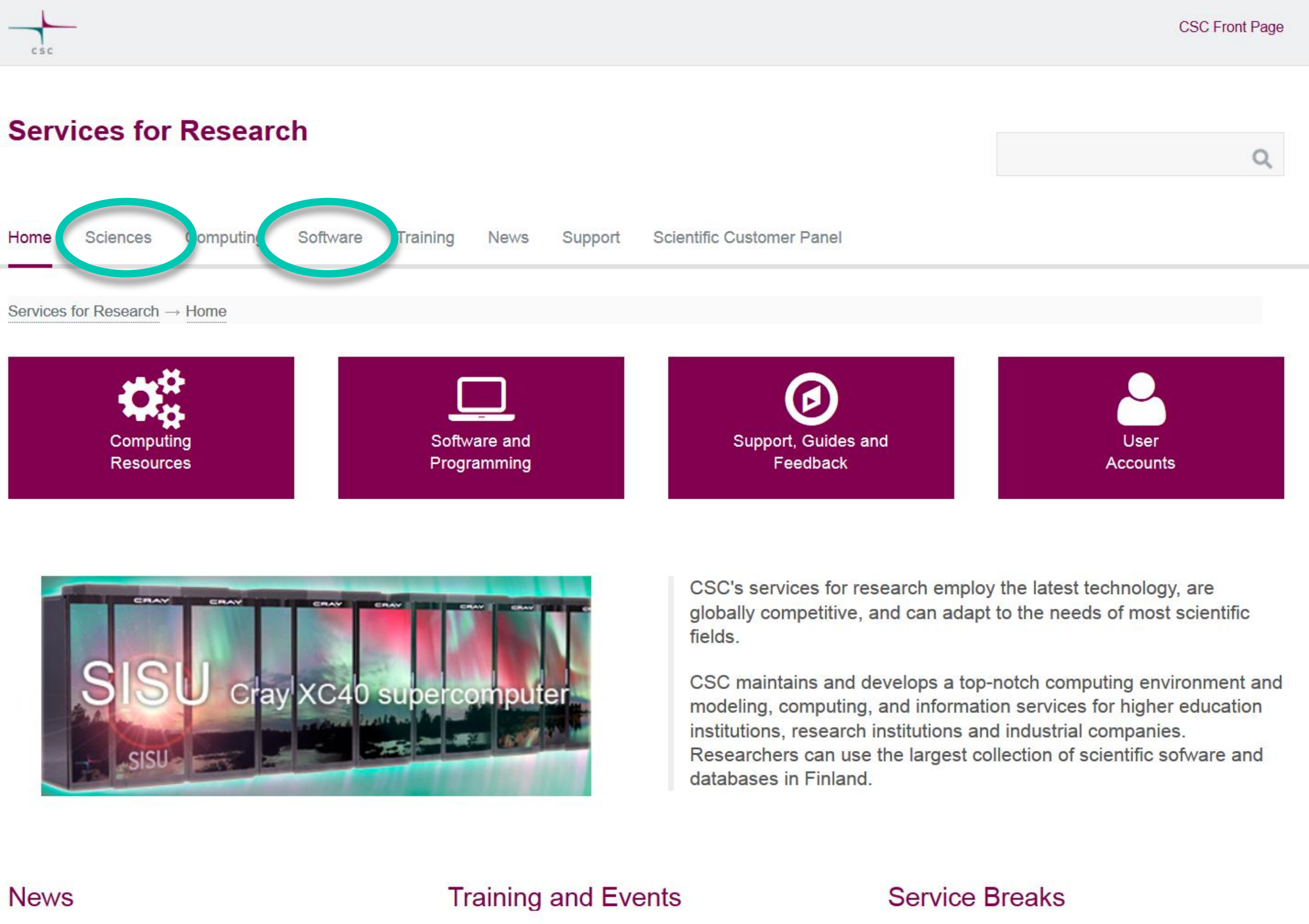


Software and databases at CSC

- Software selection at CSC: http://research.csc.fi/software

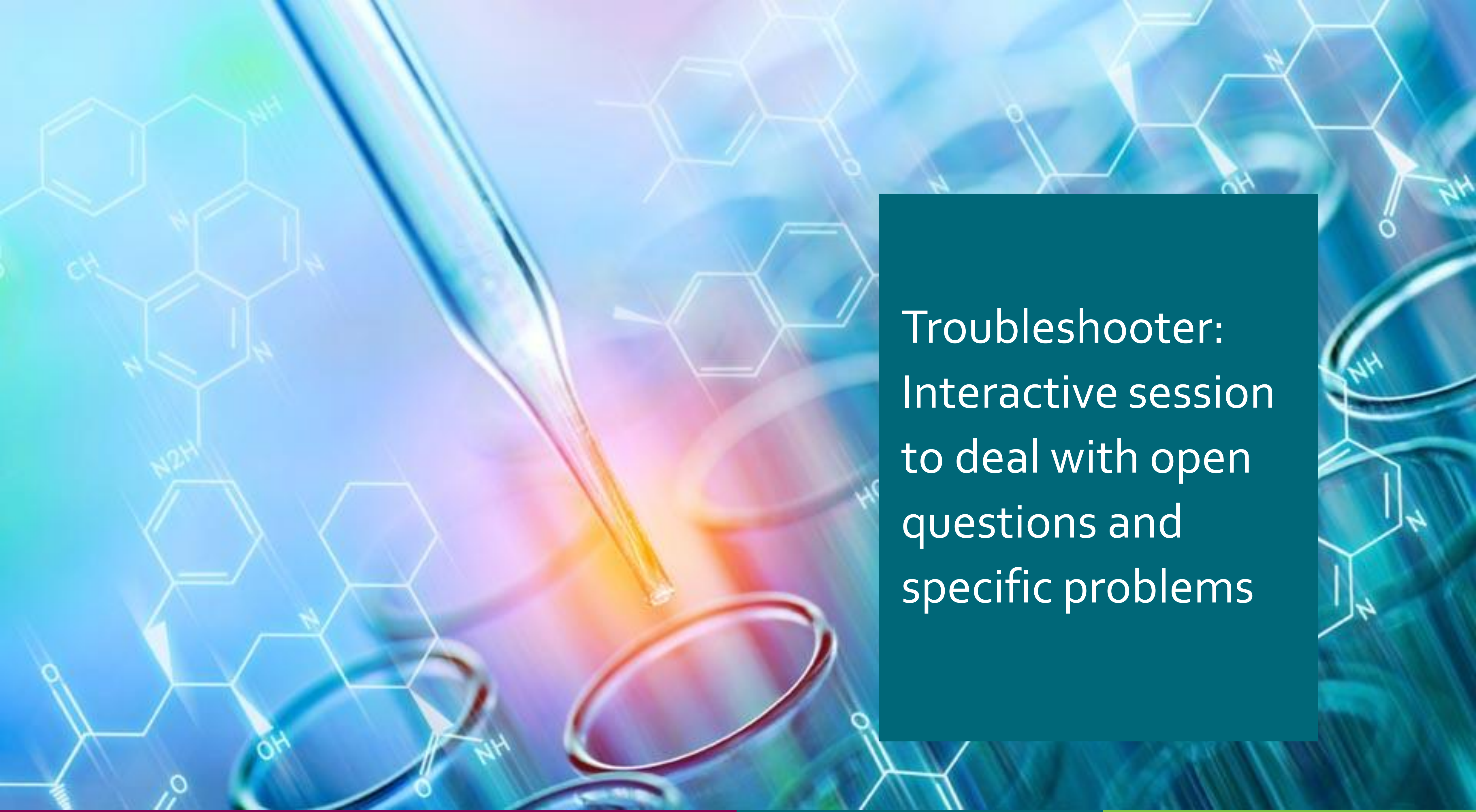
- Science discipline specific pages: http://research.csc.fi/biosciences http://research.csc.fi/chemistry

- Chipster data analysis environment: http://chipster.csc.fi



CSC's services for research employ the latest technology, are globally competitive, and can adapt to the needs of most scientific fields.

CSC maintains and develops a top-notch computing environment and modeling, computing, and information services for higher education institutions, research institutions and industrial companies. Researchers can use the largest collection of scientific sofware and databases in Finland.



Troubleshooter: Interactive session to deal with open questions and specific problems

CSC

CSC – IT Center for Science Ltd

+3589 457 2821 (service requests)
+3589 457 2001 (call center/ contacts)

servicedesk@csc.fi

www.csc.fi

https://www.facebook.com/CSCfi

https://twitter.com/CSCfi

https://www.youtube.com/c/CSCfi

https://www.linkedin.com/company/csc---it-center-for-science

Kuvat CSC:n arkisto ja Thinkstock