# MetaCentrum & CERIT-SC hands-on seminar

Tomáš Rebok

MetaCentrum, CESNET z.s.p.o.

CERIT-SC, Masaryk University

(rebok@ics.muni.cz)

# Overview

- **Brief MetaCentrum introduction**
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# MetaCentrum @ CESNET

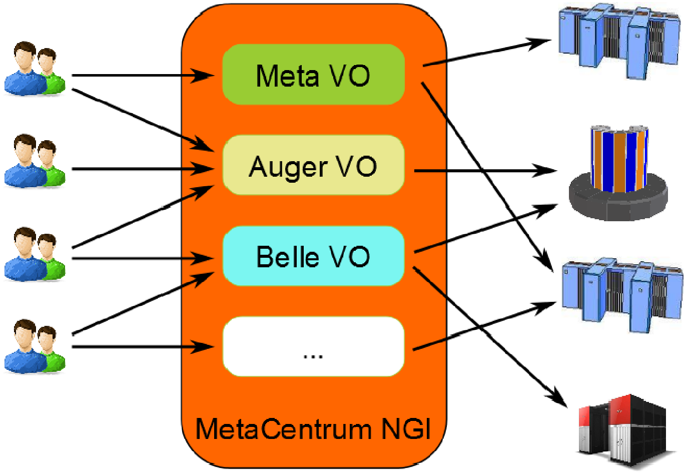
- CESNET department
- since 1996, responsible for coordinating and managing **grid activities in the Czech Republic** on behalf of the **Czech NGI**
  - comprises of **clusters, powerful servers** and **storages** provided by CESNET itself as well as cooperating institutions/universities
  - → an environment for collaboration in the area of computations and data processing/management
  - interconnected with European Grid Infrastructure (EGI)

# MetaCentrum NGI

http://www.metacentrum.cz

- NGI coordinator
- users are grouped into **virtual organizations (VOs)**
  - a group of users having “something in common”
    - e.g., cooperating on the same project
  - may have specific HW resources assigned, specific policies set, specific technologies in use, etc.

- **MetaCentrum NGI** may help with:
  - *establishment of a new HW centre*
  - *establishment of a new VO*
  - *integrating existing resources into grid infrastructure*
  - *joining a project with european infrastructures*



# MetaCentrum VO (Meta VO)

- intended for students/employees of Czech universities, Academy of Sciences, various research institutes, etc.

http://metavo.metacentrum.cz

- offers:
  - computing resources
  - storage capacities
  - application programs
- **free of charge** (after **registration**)
  - „payment“ in the form of publications with acknowledgement
  - → user priorities when the resources become fully utilized

- a part of CESNET’s e-infrastructure
  - data storage/repository, collaborative environment, …

# Meta VO – hardware

- resources of CESNET + involved organizations/institutions
  - ZČU, UK, MU, CERIT-SC, FZÚ AV ČR, JČU, MZLU, VUTBR, …
  - → CESNET performs the coordination
- computing resources: ca **5700 cores** (x86_64)
  - common HD nodes (2x4-8 cores) as well as SMP nodes (32-80 cores)
  - memory up to 512 GB per node
  - Infiniband for low-latency communication (MPI apps)
- **400 TB** for semi-permanent data
  - storage sites in Brno (3x) and Pilsen (1x), accessible from all clusters
  - prospectively being connected to CESNET’s **PB** storage
- availability of specialized equipment
  - e.g. NVIDIA CUDA cards in Pilsen, 35TB scratch for temporary data (Brno)

# Meta VO – hardware

- resources of CESNET + involved organizations/institutions
  - ZČU, UK, MU, CERIT-SC, FZÚ AV ČR, JČU, MZLU, VUTBR
- c
- 4
- a
  - no)

## What has changed in the last 6 months?

- **new clusters** – the number of cores has been increased by ca **1700**
- **installation of new clusters at JCU** in progress (further cores)
- we're finishing the purchase of **1 TB RAM machine**
- the **nodes equipped by GPU cards** are raising
- we're planning to purchase **2 new ~ 100 TB storage arrays** (Prague, Budejovice) for semi-permanent data
- an **establishment of a connection to Cesnet's PB data storage** in progress (currently used for backup only)
- …

# Meta VO – software

- similarly to HW, obtained in cooperation with involved organizations
- **~ 160 different applications** (see http://meta.cesnet.cz/wiki/Kategorie:Aplikace )
- **development tools**
  - GNU, Intel, PGI, debuggers and profiling tools (TotalView, Allinea)
- **mathematical software**
  - Matlab, Maple, gridMathematica
- **commercial/free software for chemistry**
  - Gaussian 09, Amber, Gamess, …
- **material simmulations**
  - Wien2k, Ansys Fluent, …
- **structural biology, bioinformatics**
  - a set of freely available modules
- we're looking for **new software proposals** (free/commercial)
  - possibility to buy/co-finance

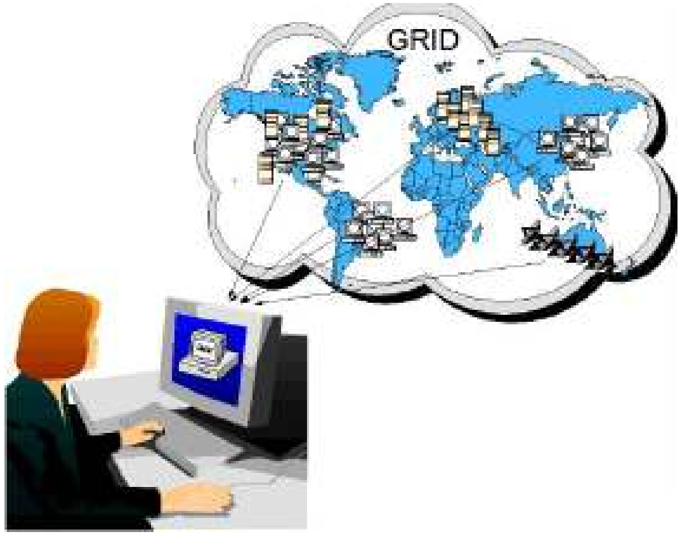
# Meta VO – software

- similarly to HW, obtained in cooperation with involved organizations
- **~ 160 different applications** (see http://meta.cesnet.cz/wiki/Kategorie:Aplikace )

## What has changed in the last 6 months?

- **Matlab (8.0), Matlab** basic licences **increased by 100 (350 in total)**
- **Matlab DCS/DCE increased by 128 (160 in total)**
- **TotalView 8.10** and **Allinea DDT 3.2** debuggers
- **Ansys CFD 14.0** (Fluent + CFX)**, Ansys HPC**
- **Maple 16**
- **PGI CDK 12.4**
- **Intel CDK 12** licences **increased by 2**
- **SciLab, CMAQ, Moses, Mosaik, Gromacs, QEspresso, …**

# Meta VO – computing environment



- *batch jobs*
  - descriptive job script
  - information about job's start/termination
- *interactive jobs*
  - **text** vs. **graphical** mode
- *cloud environment*
  - pilot installation with CERIT-SC
  - basic compatibility with Amazon EC2
  - users **do not run jobs**, but the whole **virtual machines**
    - possibility to tune the image (Windows, Linux) and start it on MetaVO nodes
    - suitable for applications, which do not comply with the grid approach

# Overview

- Brief MetaCentrum introduction
- **Brief CERIT-SC Centre introduction**

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# CERIT-SC Centre

- an important member/partner of the Czech national grid (∈MetaVO)
  1. provider of **HW resources**
     - ❑ SMP nodes (1600 cores, already installed)
     - ❑ HD nodes (580 cores, goal Q1/2013 >2500 cores)
     - ❑ storage capacity (~3,2 PB, goal Q1/2013 >3,5 PB)

**II. services beyond the scope of a “common” HW centre –**

**an environment for collaborative research**

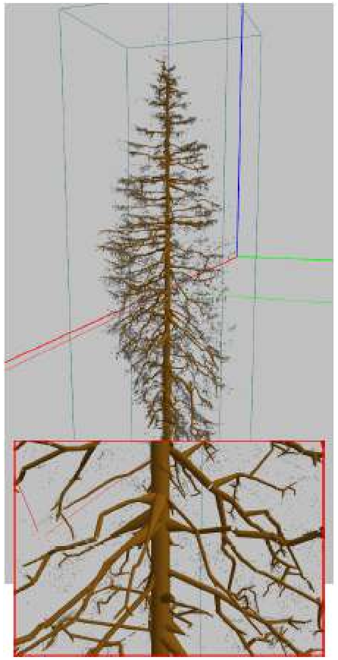
http://www.cerit-sc.cz

# CERIT-SC – main activities

- Infrastructure
  - interactive, convenient for experiments (highly flexible)
  - installed technology serves primarily for research and experiments
    - the latter purpose is for common computations and data storage/processing
  - minimal paperwork (no applications)
- Research and Development
  - **own research**, focused on principles/technologies of the maintained eInfrastructure and its optimization
  - **collaborative**, comprises of a design and optimization of algorithms, models, tools and environment based on the needs of our users/partners
    - → **a collaboration of** IT experts and users

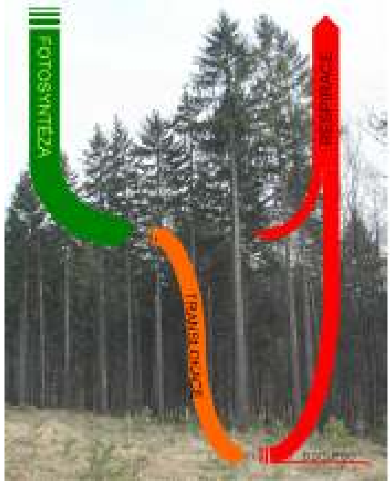
# Examples of collaboration with partners I.

- **Rekonstrukce stromu z jeho laserového skenu**
  - **partner:** *CzechGlobe* (prof. Marek, doc. Zemek, dr. Hanuš, dr. Kaplan)
  - **cíl projektu:** návrh algoritmu pro rekonstrukci stromu (smrků)
    - z mraku nasnímaných 3D bodů
      - strom nasnímán laserovým snímačem LIDAR
      - výstupem jsou souřadnice XYZ + intenzita odrazu
    - *očekávaný výstup:* struktura popisující strom (Blender)
  - **hlavní problémy:** překryvy (mezery v datech)



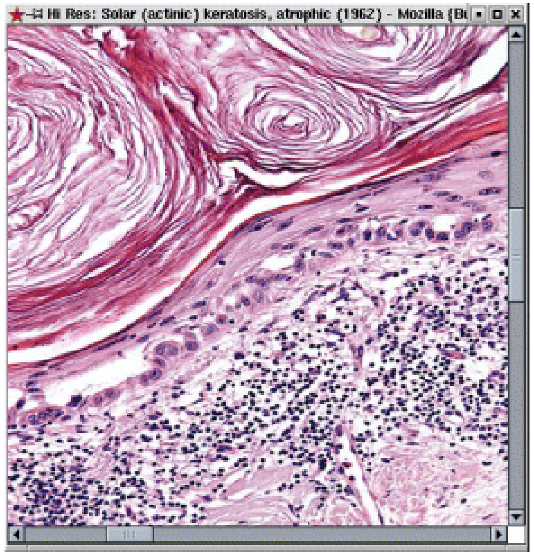
# Examples of collaboration with partners II.

- **Použití neuronových sítí pro doplňování chybějících dat eddy-kovariančních měření**
  - ❑ **partner:** *CzechGlobe* (prof. Marek, dr. Pavelka)
  - ❑ **cíl projektu:** nalezení nové, plně automatické metody pro doplňování chybějících měření
    - formou učení na historických datech
      - ❑ *doprovodné charakteristiky* – teplota, tlak, vlhkost, …
  - ❑ **hlavní problémy:**
    - nutnost brát v úvahu i historická data
    - les se vyvíjí (roste)



# Examples of collaboration with partners III.

- **Virtuální mikroskop, patologické atlasy**
  - **partner:** *LF MU* (doc. Feit)
  - **cíl projektu:** implementace virtuálního mikroskopu pro dermatologický atlas (webová aplikace)
    - zobrazuje skeny tkání
      - rozlišení až 170000x140000 pixelů
      - složeno z dlaždic (až 30000 ks)
    - umožňuje „doostřovat“ jako skutečný mikroskop
  - **hlavní problémy:**
    - optimalizace zpracování snímků, autentizace


Hi Res: Solar (actinic) keratosis, atrophic (1962) - Mozilla (Bı

# Examples of collaboration with partners IV.

- **ELIXIR (ESFRI projekt) – bio-informační infrastruktura**
  - **partner:** *ÚOCHB AV ČR* (dr. Vondrášek), *BIOMED AV ČR* (dr. Vohradský)
  - **cíl projektu:** vybudování českého uzlu evropské bio-informační infrastruktury
    - data z oblastí přírodovědy a medicíny, případně i z příslušně orientovaných průmyslových odvětví
    - potřeba s daty organizovaně zacházet
      - sdružování, správa a přístup k datům
      - nutná celoevropská a celosvětová koordinace
  - CERIT-SC je zakládající člen konsorcia ELIXIR_CZ

# Examples of collaboration with partners V.

- Biobanka klinických vzorků (BBMRI_CZ)
  - ***partner:*** *Masarykův onkologický ústav, Recamo*
- Modely šíření epileptického záchvatu a dalších dějů v mozku
  - ***partner:*** *LF MU, ÚPT AV, CEITEC*
- Fotometrický archív astronomických snímků
- Extrakce fotometrických údajů o objektech z astronomických snímků
- Automatické fitování kontinua echelletovských spekter
  - ***3x partner:*** *Ústav teoretické fyziky a astrofyziky PřF MU*
- Bioinformatická analýza dat z hmotnostního spektrometru
  - ***partner:*** *Ústav experimentální biologie PřF MU*
- Identifikace oblastí zasažených geometrickými distorzemi v leteckých skenech krajiny
- Synchronizace časových značek v leteckých snímcích krajiny
  - ***2x partner:*** *CzechGlobe*
- …

# CERIT-SC HW/SW equipment summary

**Hardware:**

- **SMP: 20 nodes** (*zewura* cluster)
  - **80 cores** and **512 GB** of memory per node
- **HD: 48 nodes** (*zegox* cluster)
  - **12 cores** and **90 GB** of memory per node
- the clusters' nodes interconnected by **Infiniband**
- own **storage volume** for user homes

**Software:**

- **exactly the same as available on the other MetaVO nodes**

# Overview

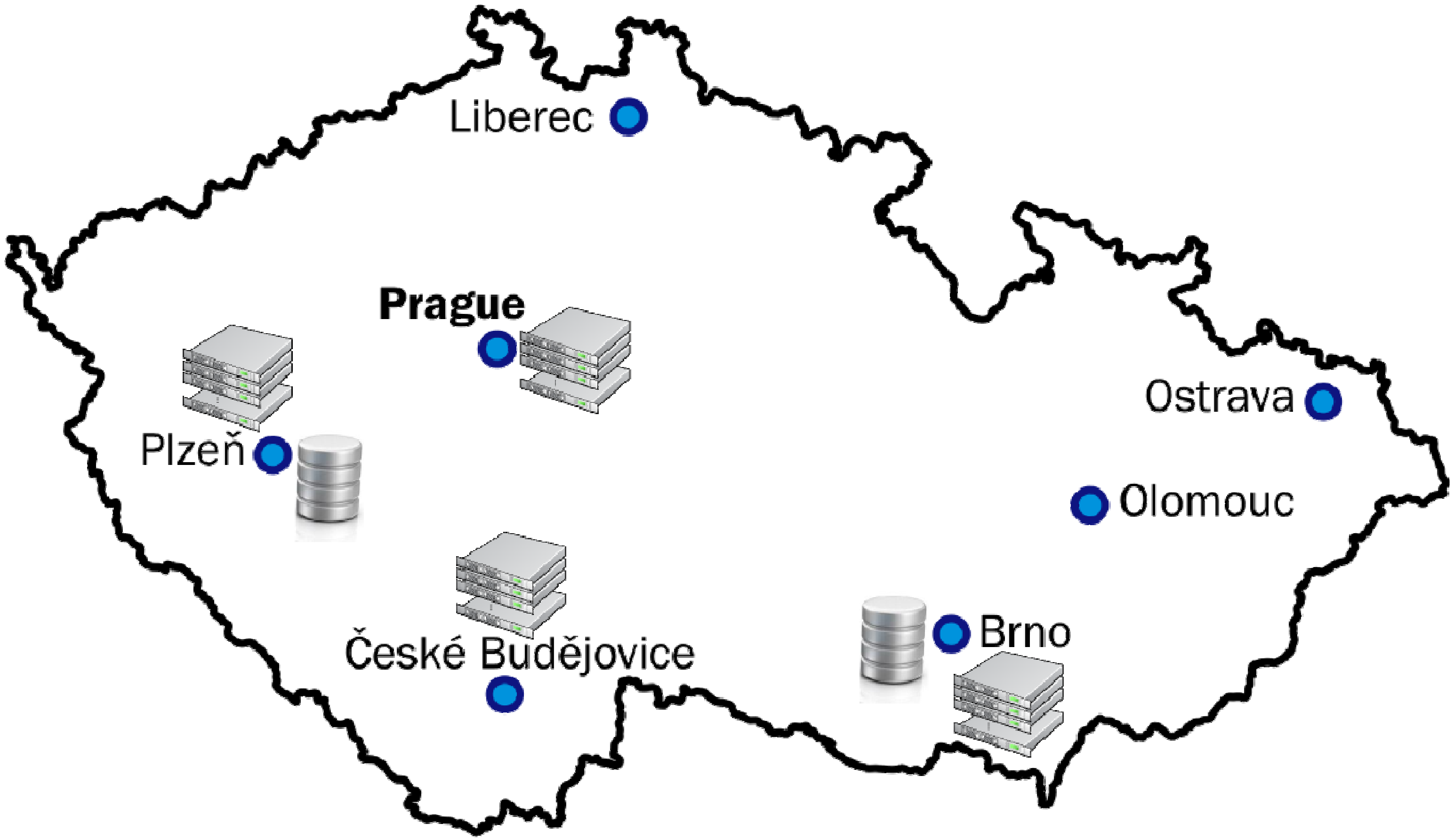
- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- **Grid infrastructure overview**
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

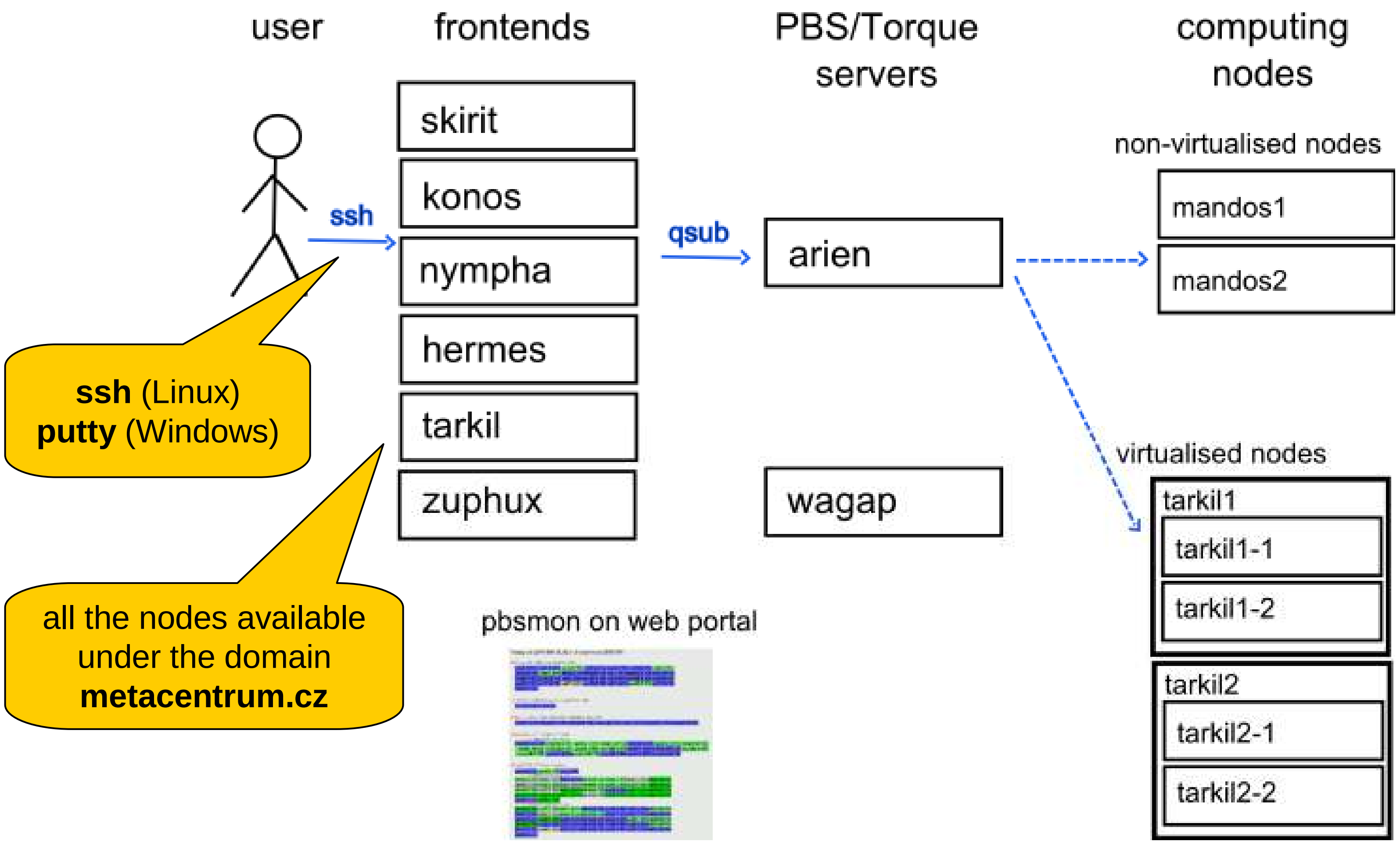
- CERIT-SC specifics

- Real-world examples

# Grid infrastructure overview I.



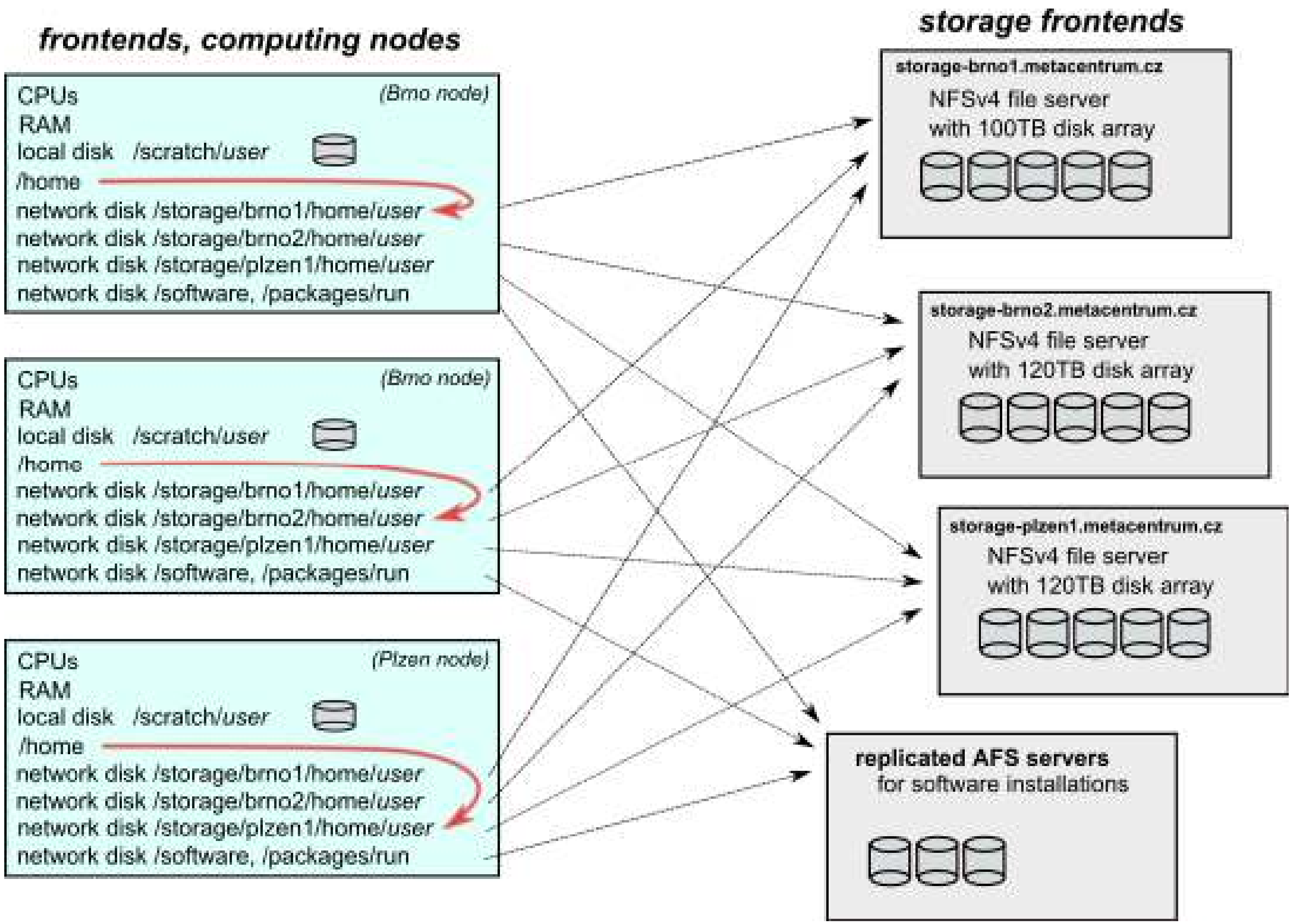
# Grid infrastructure overview II.



**ssh** (Linux)
**putty** (Windows)

all the nodes available under the domain **metacentrum.cz**

# Grid infrastructure overview III.



frontends, computing nodes

CPUs (Brno node)
RAM
local disk /scratch/*user*
/home
network disk /storage/brno1/home/*user*
network disk /storage/brno2/home/*user*
network disk /storage/plzen1/home/*user*
network disk /software, /packages/run

CPUs (Brno node)
RAM
local disk /scratch/*user*
/home
network disk /storage/brno1/home/*user*
network disk /storage/brno2/home/*user*
network disk /storage/plzen1/home/*user*
network disk /software, /packages/run

CPUs (Plzen node)
RAM
local disk /scratch/*user*
/home
network disk /storage/brno1/home/*user*
network disk /storage/brno2/home/*user*
network disk /storage/plzen1/home/*user*
network disk /software, /packages/run

storage frontends

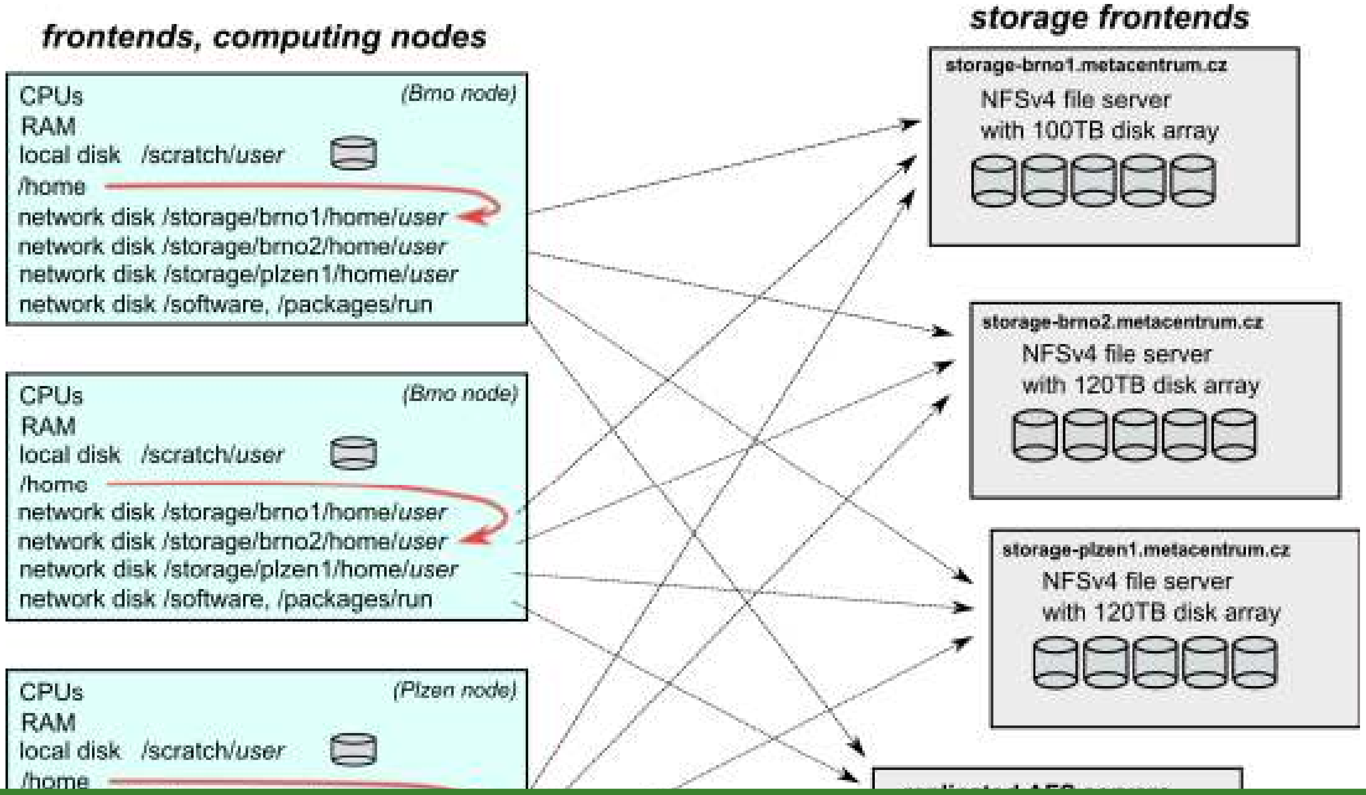
storage-brno1.metacentrum.cz
NFSv4 file server
with 100TB disk array

storage-brno2.metacentrum.cz
NFSv4 file server
with 120TB disk array

storage-plzen1.metacentrum.cz
NFSv4 file server
with 120TB disk array

replicated AFS servers
for software installations

# Grid infrastructure overview III.



frontends, computing nodes

CPUs (Brno node)
RAM
local disk /scratch/*user*
/home
network disk /storage/brno1/home/*user*
network disk /storage/brno2/home/*user*
network disk /storage/plzen1/home/*user*
network disk /software, /packages/run

CPUs (Brno node)
RAM
local disk /scratch/*user*
/home
network disk /storage/brno1/home/*user*
network disk /storage/brno2/home/*user*
network disk /storage/plzen1/home/*user*
network disk /software, /packages/run

CPUs (Plzen node)
RAM
local disk /scratch/*user*
/home

storage frontends

storage-brno1.metacentrum.cz
NFSv4 file server
with 100TB disk array

storage-brno2.metacentrum.cz
NFSv4 file server
with 120TB disk array

storage-plzen1.metacentrum.cz
NFSv4 file server
with 120TB disk array

## Planned improvements:

- the /storage/XXX/home/$USER **as default login directory**

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- **How to … specify requested resources**
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# How to … specify requested resources I.

- before running a job, one needs to have an idea **what resources** the job requires
  - and how many of them
- means for example:
  - number of **nodes**
  - number of **cores per node**
  - an **upper estimation** of job's **runtime**
  - amount of **free memory**
  - amount of **scratch space** for temporal data
  - number of requested **software licenses**
  - etc.
- the resource requirements are then **provided to** the **qsub utility** (when submitting a job)

- **details about resources' specification:** http://meta.cesnet.cz/wiki/Plánovací_systém_-_detailní_popis#Specifikace_požadavků_na_výpočetní_zdroje

# How to … specify requested resources II.

**Graphical way:**

- *qsub assembler:* http://metavo.metacentrum.cz/cs/state/personal

- allows to:
  - graphically specify the requested resources
  - check, whether such resources are available
  - generate command line options for qsub
  - check the usage of MetaVO resources

**Textual way:**

- **more powerful** and (once being experienced user) **more convenient**
- see the following slides/examples →

# How to … specify requested resources III.

**Node(s) specification:**

- *general format:* `-l` **nodes**`=...`

*Examples:*

- 2 nodes:
  - `-l nodes=2`
- 5 nodes:
  - `-l nodes=5`

- by default, allocates just a single core on each node
  - → should be used together with **processors per node (PPN)** specification
- if “`-l nodes=...`” is not provided, just a single node with a single core is allocated

# How to … specify requested resources IV.

**Processors per node (PPN) specification:**

- *general format:* `-l nodes=...:`**`ppn`**`=...`

- 2 nodes, both of them having 3 processors:
  - `-l nodes=2:ppn=3`
- 5 nodes, each of them with 2 processors:
  - `-l nodes=5:ppn=2`

More complex specifications are also supported:

- 3 nodes: one of them with just a single processor, the other two with four processors per node:
  - `-l nodes=1:ppn=1+2:ppn=4`
- 4 nodes: one with a single processor, one with two processors, and two with four processors:
  - `-l nodes=1:ppn=1+1:ppn=2+2:ppn=4`

# How to … specify requested resources V.

**Other useful nodespec features:**

- nodes just from a **single (specified) cluster** (suitable e.g. for MPI jobs):
  - *general format:* `-l nodes=…:cl_<cluster_name>`
  - e.g., `-l nodes=3:ppn=1:cl_skirit`
- nodes located in a **specific location** (suitable when accessing storage in the location)
  - *general format:* `-l nodes=…:<brno|plzen|...>`
  - e.g., `-l nodes=1:ppn=4:brno`
- asking for a **specific node(s)**:
  - *general format:* `-l nodes=…:<node_name>`
  - e.g., `-l nodes=1:ppn=4:manwe3.ics.muni.cz`
- **exclusive node assignment**:
  - *general format:* `-l nodes=…#excl`
  - e.g., `-l nodes=1#excl`
- **negative** specification:
  - *general format:* `-l nodes=…:^<feature>`
  - e.g., `-l nodes=1:ppn=4:^cl_manwe`
- ...

A list of nodes' features can be found here: http://metavo.metacentrum.cz/pbsmon2/props

# How to … specify requested resources VI.

**Specifying memory resources** (default = 400mb)**:**

- *general format:* `-l mem=...<suffix>`
  - e.g., `-l mem=300mb`
  - e.g., `-l mem=2gb`

**Specifying job's maximum runtime** (default = normal)**:**

- it is necessary to assign a job into a queue, providing an upper limit on job's runtime:
  - **short** = 2 hours, **normal** (default) = 24 hours, **long** = 1 month
- *general format:* `-q <queue_name>`
  - e.g., `-q short`
  - e.g., `-q long`

# How to … specify requested resources VII.

**Specifying requested scratch space:**

- useful, when the application performs I/O intensive operations
  - the scratches are **local to the nodes** (smaller) and/or **shared for the nodes** of a specific cluster over Infiniband (bigger) -- currently “mandos” cluster only
    - thus being as fast as possible
- *scratch space* ***(amount in Kbytes)***: `-l scratch=<amount>`
  - `e.g., -l scratch=500000` (asking for 500MB)
- there is a **private scratch directory for particular job**
  - `/scratch/$USER/`**`job_$PBS_JOBID`** directory for job’s scratch
- there is a **`SCRATCHDIR` environment variable** available in the system
  - points to the assigned scratch space/location

# How to … specify requested resources VII.

**Specifying requested scratch space:**

- useful, when the application performs I/O intensive operations

a

## Planned improvements:

**SCRATCH:**

- possibility to set the requested scratch space in human-readable format (KB, MB, GB, …)
- additional property to indicate a specific scratch type requested
  - `-l scratch_type=[local|shared|ssd|first]`
- reservations/quotas

**General:**

- a possibility to choose the nodes/cores with a particular (specified) performance (or higher)

# How to … specify requested resources VIII.

## Specifying requested software licenses:

- necessary when an application requires a SW licence
  - the job becomes started once the requested licences are available
  - the information about a licence necessity is **provided within the application description** (see later)
- *general format:* `-l <lic_name>=<amount>`
  - e.g., `-l matlab=2`
  - e.g., `-l gridmath8=20`

**…**

## (advanced) Dependencies on another jobs

- allows to create a workflow
  - e.g., to start a job once another one successfully finishes, breaks, etc.
- see qsub's "**`–W`**" option (`man qsub`)

# How to … specify requested resources IX.

**Questions and Answers:**

- *Why is it necessary to specify the resources in a proper number/amount?*
  - because when a job consumes more resources than announced, it will be **killed** by us (you’ll be informed)
    - otherwise it may influence other processes running on the node
- *Why is it necessary not to ask for excessive number/amount of resources?*
  - the jobs having smaller resource requirements are started (i.e., get the time slot) **faster**

- *Any other questions?*

# How to … specify requested resources X.

**Examples:**

- *Ask for a single node with 4 CPUs, 1gb of memory.*
  - `qsub –l nodes=1:ppn=4 –l mem=1gb`
- *Ask for a single node (1 CPU) – the job will run approx. 3 days and will consume up to 10gb of memory.*
  - *???*
- *Ask for 2 nodes (1 CPU per node) not being located in Brno.*
  - *???*
- *Ask for two nodes – a single one with 1 CPU, the other two having 5 CPUs and being from the manwe cluster.*
  - *???*
- *…*

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- **How to … run an interactive job**
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# How to … run an interactive job I.

**Interactive jobs:**

- result in getting a prompt on a single **(master) node**
  - one may perform interactive computations
  - the other nodes, if requested, remain allocated and accessible (see later)

- How to **ask for** an **interactive job**?
  - add the option “`-I`” to the qsub command
  - e.g., `qsub –I –l nodes=1:ppn=4:cl_mandos`

- **Example** (valid for this demo session):
  - `qsub –I –q MetaSeminar –l nodes=1`

# How to … run an interactive job II.

**Textual mode:** simple

**Graphical mode:**

- *(easier, preferred)* **tunnelling a display through ssh** (Windows/Linux)**:**
  - connect to the frontend node having SSH forwarding/tunneling enabled:
    - Linux: `ssh –X skirit.metacentrum.cz`
    - Windows:
      - install an XServer (e.g., Xming)
      - set Putty appropriately to enable X11 forwarding when connecting to the frontend node
        - Connection → SSH → X11 → Enable X11 forwarding
  - ask for an interactive job, **adding "`-X`" option** to the qsub command
    - e.g., `qsub –I `**`–X`**` –l nodes=... ...`

- **exporting a display** from the master node to a Linux box:
  - `export DISPLAY=mycomputer.mydomain.cz:0.0`
  - on a Linux box, run "`xhost +`" to allow all the remote clients to connect
    - be sure that your display manager allows remote connections

# How to … run an interactive job III.

**Questions and Answers:**

- *How to **get an information** about the **other nodes allocated** (if requested)?*
  - `master_node$ cat $PBS_NODEFILE`
  - works for batch jobs as well
- *How to **use the other nodes allocated**? (holds for batch jobs as well)*
  - MPI jobs use them automatically
  - otherwise, use the **pbsdsh** utility (see `”man pbsdsh”` for details) to run a remote command
  - if the pbsdsh does not work for you, use the **ssh** to run the remote command
- *Any other questions?*

# How to … run an interactive job III.

**Questions and Answers:**

- *How to **get an information** about the **other nodes allocated** (if*

- )

  un a

-

## Hint:

- there are several useful environment variables one may use
  - `$ set | egrep “PBS|TORQUE”`
- e.g.:
  - PBS_JOBID … job’s identificator
  - PBS_NUM_NODES, PBS_NUM_PPN … allocated number of nodes/processors
  - PBS_O_WORKDIR … submit directory (alert: /home path!)
  - …

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- **How to … use application modules**
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# How to … use application modules I.

**Application modules:**

- the **modullar subsystem** provides a user interface to modifications of user environment, which are necessary for running the requested applications
- allows to “add” an application to a user environment

- **getting a list** of available application modules:
  - `$ module avail`
  - http://meta.cesnet.cz/wiki/Kategorie:Aplikace
    - provides the documentation about modules’ usage
    - besides others, includes:
      - information whether it is necessary to ask the scheduler for an available licence
      - information whether it is necessary to express consent with their licence agreement

# How to … use application modules II.

## Application modules:

- **loading** an application into the environment:
  - `$ module add <modulename>`
  - e.g., `module add maple`
- **listing** the already loaded modules:
  - `$ module list`
- **unloading** an application from the environment:
  - `$ module del <modulename>`
  - e.g., `module del openmpi`

- ***Note:*** *An application may require to express consent with its licence agreement before it may be used (see the application's description). To provide the aggreement, visit the following webpage:* http://metavo.metacentrum.cz/cs/myaccount/eula

- for more information about application modules, see http://meta.cesnet.cz/wiki/Aplikační_moduly

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- **How to … run a batch job**
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# How to … run a batch job I.

**Batch jobs:**

- perform the computation as described in their **startup script**
  - the submission results in getting **a job identifier**, which further serves for getting more information about the job (see later)

- How to **submit** a **batch job**?
  - add the reference to the startup script to the qsub command
  - e.g., `qsub –l nodes=3:ppn=4:cl_mandos` **`<myscript.sh>`**

- **Example** (valid for this demo session):
  - `qsub` **`–q MetaSeminar`** `–l nodes=1` **`myscript.sh`**
  - results in getting something like “`12345.arien.ics.muni.cz`”

# How to run a batch job I

Ba

for

### Hint:

- create the file `myscript.sh` with the following content:
  - `$ vim myscript.sh`

```
#!/bin/bash

# my first batch job
uname –a
```

- see the standard output file (`myscript.sh.o<JOBID>`)
  - `$ cat myscript.sh.o<JOBID>`

- **Example** (valid for this demo session):
  - `qsub –q MetaSeminar –l nodes=1 myscript.sh`
  - results in getting something like “`12345.arien.ics.muni.cz`”

# How to … run a batch job II.

**Startup script preparation/skelet: (non IO-intensive computations)**

```
#!/bin/bash

DATADIR="/storage/brno1/home/$USER/"   # shared via NFSv4
cd $DATADIR

# ... initialize & load modules, perform the computation ...
```

- **further details** – see http://meta.cesnet.cz/wiki/Plánovací_systém_-_detailní_popis#Příklady_použití

# How to … run a batch job III.

**Startup script preparation/skelet: (IO-intensive computations or long-term jobs)**

```
#!/bin/bash

DATADIR="/storage/brno1/home/$USER/"

# prepare the input data
cp $DATADIR/input.txt $SCRATCHDIR || exit 1

# go to the working directory and perform the computation
cd $SCRATCHDIR

# ... initialize & load modules, perform the computation ...

# move out the output data
cp $SCRATCHDIR/output.txt $DATADIR

if [ $? -ne 0 ]; then
    echo Copy output data failed. Copy them manualy from `hostname` 1>&2
    exit 2
fi

# clean the scratch temporal directory
rm -rf $SCRATCHDIR
```

# How to … run a batch job IV.

**Using the application modules within the batch script:**

- to use the **application modules** from a **batch script**, add the following line into the script (before loading the module):

```
. /packages/run/modules-2.0/init/sh
...
module add maple
```

**Getting the job’s standard output and standard error output:**

- once finished, there appear **two files** in the directory, which the job has been started from:
  - `<`*`job_name`*`>.o<`*`jobID`*`>` ... standard output
  - `<`*`job_name`*`>.e<`*`jobID`*`>` ... standard error output

  - the `<job_name>` can be modified via the `“-N”` qsub option

# How to … run a batch job V.

**Job attributes specification:**

in the case of batch jobs, the requested resources and further job information (*job attributes* in short) may be specified either on the command line (see “`man qsub`”) or directly within the script:

- by adding the “`#PBS`” directives (see “`man qsub`”):

```
#PBS -N Job_name
#PBS -l nodes=2:ppn=1
#PBS –l mem=320kb
#PBS -m abe
#
< … commands … >
```

- the submission may be then simply performed by:
  - `$ qsub myscript.sh`

# How to … run a batch job V.

**Questions and Answers:**

- *Should I prefer batch or interactive jobs?*
  - definitely the **batch ones** – they use the computing resources **more effectively**
  - use the interactive ones just for testing your startup script, GUI apps, or data preparation

- *Any other questions?*

# How to … run a batch job VI.

**Example:**

- Create and submit a batch script, which performs a simple Maple computation, described in a file:

```
plotsetup(gif,
   plotoutput=`/storage/brno1/home/<username>/myplot.gif`,
   plotoptions=`height=1024,width=768`);
plot3d( x*y, x=-1..1, y=-1..1, axes = BOXED, style =
   PATCH);
```

  - ❑ process the file using Maple (from a batch script):
    - hint: `$ maple <filename>`

# How to … run a batch job VI.

**Example:**

- Create and submit a batch script, which performs a simple Maple computation, described in a file:

```
plotsetup(gif,
   plotoutput=`/storage/brno1/home/<username>/myplot.gif`,
   plotoptions=`height=1024,width=768`);
plot3d( x*y, x=-1..1, y=-1..1, axes = BOXED, style =
   PATCH);
```

- ❑ process the file using Maple (from a batch script):
  - hint: `$ maple <filename>`

## Hint:

- see the solution at `/storage/brno2/home/jeronimo/MetaSeminar/20121114-UEB/maple`

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- **How to … determine a job state**
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# How to … determine a job state I.

## Job identifiers

- every job (no matter whether interactive or batch) is **uniquely identified** by its identifier (JOBID)
  - e.g., `12345.arien.ics.muni.cz`
- to obtain any information about a job, the **knowledge of its identifier is necessary**
  - how to list all the recent jobs?
    - graphical way – PBSMON: http://metavo.metacentrum.cz/pbsmon2/jobs/allJobs
    - `frontend$ qstat` (run on any frontend)
  - how to list all the recent jobs of a specific user?
    - graphical way – PBSMON: http://metavo.metacentrum.cz/pbsmon2/jobs/my
    - `frontend$ qstat –u <username>` (again, any frontend)

# How to … determine a job state II.

## How to determine a job state?

- graphical way – see PBSMON
  - ❑ list all your jobs and click on the particular job's identifier
  - ❑ http://metavo.metacentrum.cz/pbsmon2/jobs/my
- textual way – `qstat` command (see `man qstat`)
  - ❑ brief information about a job: `$ qstat JOBID`
    - informs about: job's state (*Q=queued*, *R=running*, *E=exiting*, *C=completed*, …), job's runtime, …
  - ❑ complex information about a job: `$ qstat –f JOBID`
    - shows all the available information about a job
    - useful properties:
      - ❑ *exec_host* -- the nodes, where the job did really run
      - ❑ *resources_used, start/completion time, exit status, …*

# How to … determine a job state III.

**Hell, when my jobs will really start?**

- nobody can tell you ☺
  - the **God/scheduler decides** (based on the other job’s finish)
  - we’re working on an estimation method to inform you about its probable startup

- check the **queues’ fulfilment**: http://metavo.metacentrum.cz/cs/state/jobsQueued
  - the higher fairshare (queue’s AND job’s) is, the earlier the job will be started
- **stay informed** about job’s startup / finish / abort (via email)
  - by default, just an information about job’s abortation is sent
  - → when submitting a job, add “`-m abe`” option to the `qsub` command to be informed about all the job’s states
    - or “`#PBS –m abe`” directive to the startup script

# How to … determine a job state IV.

## Monitoring running job’s stdout, stderr, working/temporal files

1. via ssh, log in directly to the execution node(s)
   - how to get the job’s execution node(s)?
- to examine the working/temporal files, navigate directly to them
  - logging to the execution node(s) is necessary -- even though the files are on a shared storage, their content propagation takes some time
- to examine the stdout/stderr of a running job:
  - navigate to the `/var/spool/torque/spool/` directory and examine the files:
    - `$PBS_JOBID.OU` for standard output (stdout – e.g., “`1234.arien.ics.muni.cz.OU`”)
    - `$PBS_JOBID.ER` for standard error output (stderr – e.g., “`1234.arien.ics.muni.cz.ER`”)

## Job’s forcible termination

- `$ qdel JOBID`    (the job may be terminated in any previous state)
- during termination, the job turns to *E (exiting)* and finally to *C (completed)* state

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- **How to … run a parallel/distributed computation**
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# How to … run a parallel/distributed computation I.

**Parallel jobs (OpenMP):**

- if your application is able to use multiple threads via a shared memory, **ask for a single node with multiple processors**

  ```
  $ qsub –l nodes=1:ppn=...
  ```

- **make sure**, that before running your application, the **OMP_NUM_THREADS** environment variable **is appropriately set**
  - otherwise, your application will use all the cores available on the node
    - → and influence other jobs…
  - usually, setting it to **PPN** is OK

  ```
  $ export OMP_NUM_THREADS=$PBS_NUM_PPN
  ```

# How to … run a parallel/distributed computation II.

**Distributed jobs (MPI):**

- if your application consists of multiple processes communicating via a message passing interface, **ask for a set of nodes** (with arbitrary number of processors)

  ```
  $ qsub –l nodes=...:ppn=...
  ```

- **make sure**, that before running your application, the appropriate **openmpi/mpich2** module is loaded into the environment

  ```
  $ module add openmpi
  ```

  - then, you can use the `mpirun/mpiexec` routines

    ```
    $ mpirun myMPIapp
    ```

    - it's **not necessary** to provide these routines neither with the number of nodes to use ("`-np`" option) nor with the nodes itself ("`--hostfile`" option)
      - the computing nodes become **automatically detected** by the openmpi/mpich2

# How to … run a parallel/distributed computation III.

## Distributed jobs (MPI): accelerating their speed I.

- to accelerate the speed of MPI computations, ask just for the nodes interconnected by a **low-latency Infiniband interconnection**
  - all the nodes of a cluster are interconnected by Infiniband
  - there are several clusters having an Infiniband interconnection
    - mandos, minos, skirit, tarkil, nympha, zewura (CERIT-SC)

- *submission example:*

```
$ qsub –l nodes=4:ppn=2:cl_mandos myMPIscript.sh
```

- *starting the MPI computation making use of an Infiniband:*
  - in a common way: `$ mpirun myMPIapp`
    - the Infiniband will be automatically detected

# How to … run a parallel/distributed computation III.

**Distributed jobs (MPI): accelerating their speed I.**

- to accelerate the speed of MPI computations, ask just for the nodes interconnected by a **low-latency Infiniband interconnection**
  - all the nodes of a cluster are interconnected by Infiniband
  - there are several clusters having an Infiniband interconnection
    - mandos, minos, skirit, tarkil, nympha, zewura (CERIT-SC)

- *submission example:*

```
$ qsub –l nodes=4:ppn=2:cl_mandos myMPIscript.sh
```

**Planned improvements:**

- an intelligent “`infiniband`” attribute
  - just the nodes interconnected with a shared IB switch will be chosen

# How to … run a parallel/distributed computation IV.

## Distributed jobs (MPI): accelerating their speed II.

- to test the functionality of an Infiniband interconnection:
  - create a simple program `hello.c` as described here:

    http://www.slac.stanford.edu/comp/unix/farm/mpi.html
  - compile with “`mpicc`”

    ```
    $ module add openmpi
    $ mpicc hello.c –o hello
    ```
  - run the binary (within a job) with the following command:

    ```
    $ mpirun --mca btl ^tcp hello
    ```

# How to … run a parallel/distributed computation IV.

## Distributed jobs (MPI): accelerating their speed II.

- to test the functionality of an Infiniband interconnection:
  - create a simple program `hello.c` as described here:

    http://www.slac.stanford.edu/comp/unix/farm/mpi.html
  - compile with “`mpicc`”

    ```
    $ module add openmpi
    $ mpicc hello.c –o hello
    ```
  - run the binary (within a job) with the following command:

    ```
    $ mpirun --mca btl ^tcp hello
    ```

### Hint:

- see the solution at `/storage/brno2/home/jeronimo/MetaSeminar/20121114-UEB/IB_hello`

# How to … run a parallel/distributed computation V.

**Questions and Answers:**

- *Is it possible to simultaneously use both OpenMP and MPI?*
  - Yes, it is. But be sure, how many processors your job is using
    - appropriately set the “-np” option (MPI) and the OMP_NUM_THREADS variable (OpenMP)

- Any other questions?

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- **Another mini-HowTos …**
- What to do if something goes wrong?

- CERIT-SC specifics

- Real-world examples

# Another mini-HowTos … I.

- **how to transfer large amount of data to MetaVO nodes?**
  - copying through the frontends/computing nodes may not be efficient
  - → connect directly to the storage frontends (via **SCP** or **SFTP**)
    - `$ sftp storage-brno1.metacentrum.cz`
    - `$ scp <files> storage-plzen1.metacentrum.cz:<dir>`
    - etc.
    - use FTP only together with the Kerberos authentication
      - otherwise insecure
- **how to access the data arrays?**
  - **easier:** use the SFTP/SCP protocols (suitable applications)
  - **OR mount the storage arrays directly to your computer**
    - http://meta.cesnet.cz/wiki/P%C5%99ipojen%C3%AD_datov%C3%BDch_%C3%BAlo%C5%BEi%C5%A1%C5%A5_k_vlastn%C3%AD_pracovn%C3%AD_stanici_p%C5%99es_NFSv4

# Another mini-HowTos … II.

- **how to secure private data?**
  - by default, all the data are readable by everyone
  - → use common Linux/Unix mechanisms/tools to make the data private
    - `r`,`w`,`x` rights for *user*, *group*, *other*
    - e.g., `chmod go= <filename>`
      - see `man chmod`
      - use “`–R`” option for recursive traversal (applicable to directories)
- **how to restore accidentally erased data**
  - the storage arrays (⇒ including homes) are regularly backed-up
    - several times a week
  - → write an email to meta@cesnet.cz specifying what to restore

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- **What to do if something goes wrong?**

- CERIT-SC specifics

- Real-world examples

# What to do if something goes wrong?

1. check the MetaVO/CERIT-SC documentation, application module documentation
   - whether you use the things correctly
2. check, whether there haven’t been any infastructure updates performed
   - visit the webpage https://meta.cesnet.cz/wiki/Provozní_změny
     - one may stay informed via an RSS feed
3. write an email to meta@cesnet.cz, resp. support@cerit-sc.cz
   - your email will create a ticket in our Request Tracking system
     - identified by a unique number → one can easily monitor the problem solving process
   - please, include **as good problem description as possible**
     - problematic job’s JOBID, startup script, problem symptoms, etc.

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- **CERIT-SC specifics**

- Real-world examples

# CERIT-SC specifics

**In comparison with the MetaVO infrastructure, the CERIT-SC infrastructure has several specifics:**

- own **frontend** (`zuphux.cerit-sc.cz`)
- own **scheduling server** (`wagap.cerit-sc.cz`)
- **no queues** for jobs’ maximum runtime specification
  - the maximum runtime is specified via a qsub’s `walltime` parameter

# CERIT-SC: job submission

**From CERIT-SC frontend** (zuphux.cerit-sc.cz):

- “common way” (just the `walltime` specification is necessary – see later)

**From MetaCentrum frontends:**

- necessary to specify the CERIT-SC’s scheduling server:
- `skirit$ qsub –q @wagap.cerit-sc.cz –l ...`
- `skirit$ qstat –q @wagap.cerit-sc.cz`
- `skirit$ qstat –f 12345.wagap.cerit-sc.cz`
- `...`

***Note:*** *It is also possible to submit MetaVO jobs from the CERIT-SC frontend:*

- `zuphux$ qsub –q short@arien.ics.muni.cz –l ...`
- `zuphux$ qstat –q @arien.ics.muni.cz`
- `zuphux$ qstat –f 12345.arien.ics.muni.cz`
- `...`

- details: http://www.cerit-sc.cz/cs/docs/access/

# CERIT-SC: maximum job’s runtime specification

- **no queues**
- specified using the qsub’s **`walltime` parameter** (default value **24 hours**)
  - *general format:*

```
-l walltime=[[hours:]minutes:]seconds[.miliseconds]
```

- *examples:*
  - `$ qsub -l walltime=30 myjob.sh` - a request to submit the *myjob.sh* script, specifying it's maximum run-time in the length of 30 seconds (submitted via the CERIT-SC frontend)
  - `$ qsub -l walltime=10:00 myjob.sh` - a request to submit the *myjob.sh* script, specifying it's maximum run-time in the length of 10 minutes (submitted via the CERIT-SC frontend)
  - `$ qsub -q @wagap.cerit-sc.cz -l walltime=100:15:00 myjob.sh` - a request to submit the *myjob.sh* script, specifying it's maximum run-time in the length of 100 hours and 15 minutes (submitted via a MetaCentrum frontend)

# Overview

- Brief MetaCentrum introduction
- Brief CERIT-SC Centre introduction

- Grid infrastructure overview
- How to … specify requested resources
- How to … run an interactive job
- How to … use application modules
- How to … run a batch job
- How to … determine a job state
- How to … run a parallel/distributed computation
- Another mini-HowTos …
- What to do if something goes wrong?

- CERIT-SC specifics

- **Real-world examples**

# Real-world examples

***Vaše podněty:***

- Maple
- Gromacs
- Scilab
- skládání sekvencí
- paralelní Matlab

- demo sources:

```
/storage/brno2/home/jeronimo/MetaSeminar/20121114-UEB
```

**command:**

```
cp –r /storage/brno2/home/jeronimo/MetaSeminar/20121114-UEB $HOME
```

# Thank You for attending!

**rebok@ics.muni.cz**

www.cesnet.cz www.metacentrum.cz www.cerit-sc.cz