CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Managing GPUs by Slurm

Massimo Benini
HPC Advisory Council Switzerland Conference
March 31 - April 3, 2014
Lugano

Agenda

- **General Slurm introduction**
- **Slurm@CSCS**
- **Generic Resource Scheduling for GPUs**
- **Resource Utilization Reporting (RUR)**

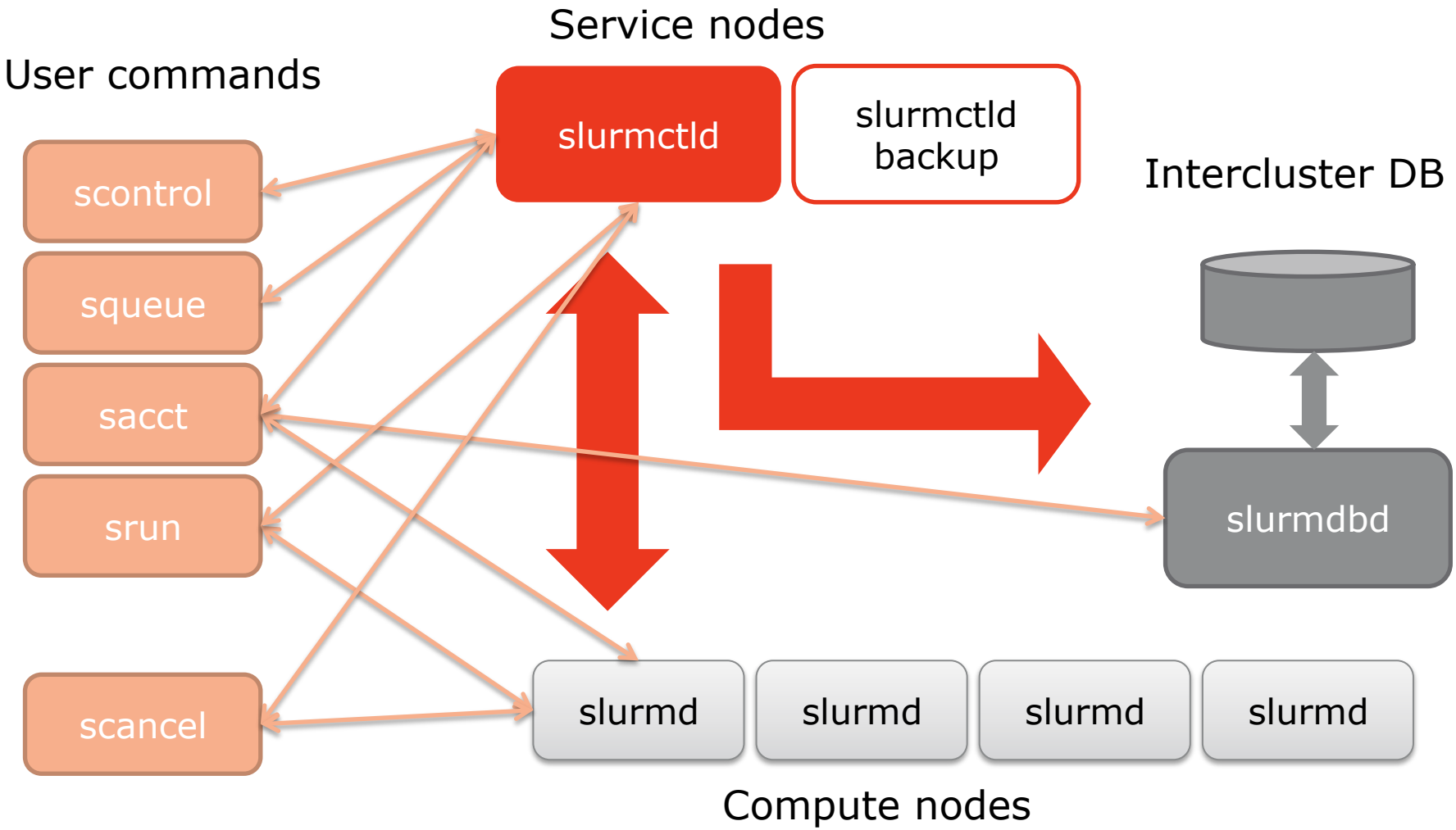
Slurm overview

- "Slurm is an open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters.."

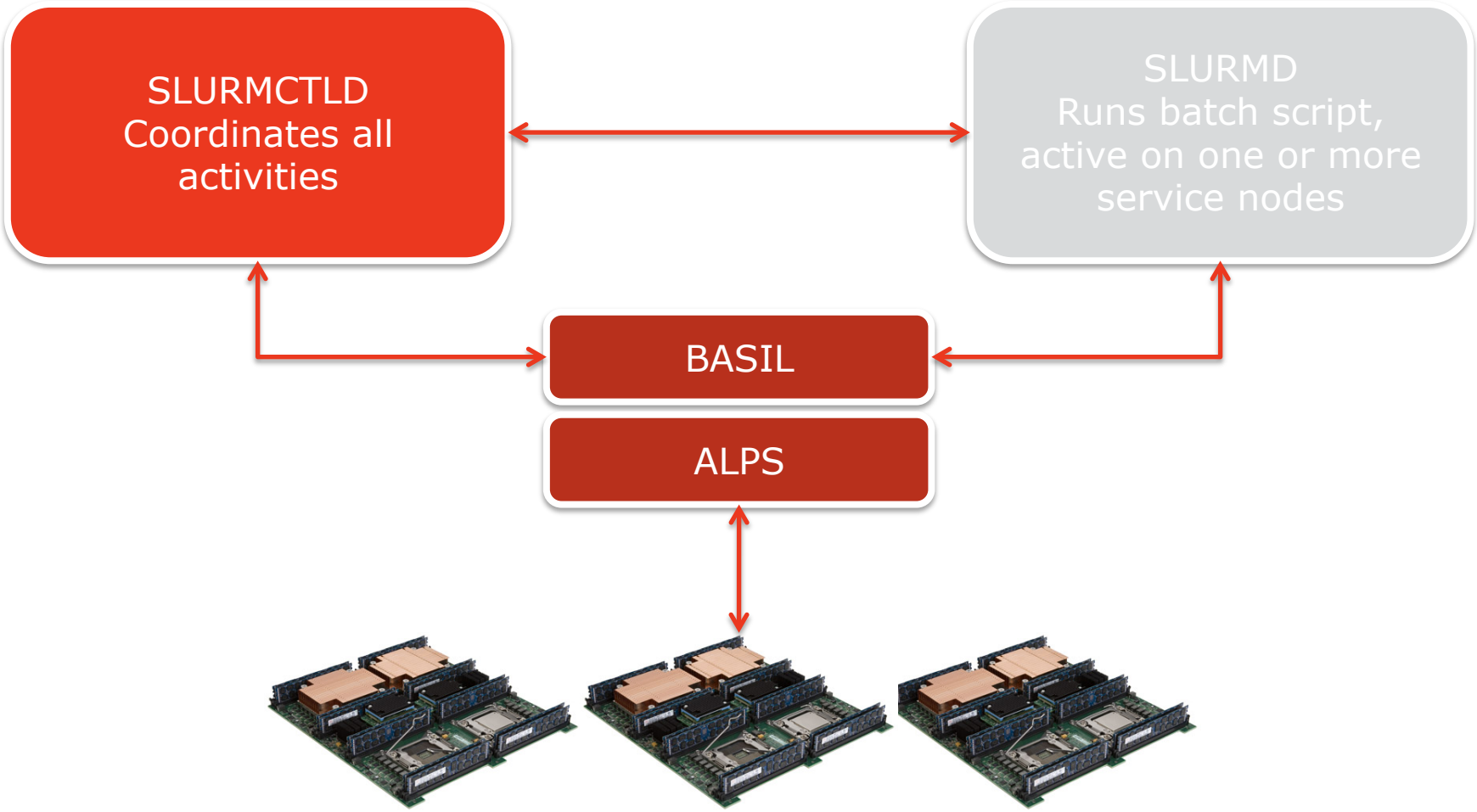
Slurm has three key functions:

- Allocates exclusive and/or non-exclusive access to resources (compute nodes) to users for some duration of time
- It provides a framework for starting, executing, and monitoring work (normally a parallel job) on the set of allocated nodes
- It arbitrates contention for resources by managing a queue of pending work

Architecture in a general Linux cluster



Arhitecture on the Crays



Plugins

General purpose plugin mechanism -> Flexibility

- Accounting Storage
- Generic Resources
- Job Submit
- Priority
- Scheduler
- Task affinity
- Node Selection
- ….

Slurm@CSCS

Machine	Arch Type	# of nodes	# of cores	Node Layout	GPU	Node memory
Daint	XC30	5272	42176	1x8x1	5272 Tesla K20X	32GB
Rosa	XE6	1496	47872	2x16x1	None	32GB
Todi	XK7	272	4352	1x16x1	272 Tesla K20X	32GB
Julier	non-Cray	12	288	2x6x2	None	10-48GB 2-256GB
Pilatus	non-Cray	44	704	2x16x2	None	64G
Albis	XE6	72	1728	2x12x1	None	32GB
Lema	XE6	168	4032	2x12x1	None	32GB

Slurm@CSCS 2/2

Machine	Arch Type	# of nodes	# of cores	Node Layout	GPU	Node memory
Castor	non-Cray	32	384	2x6x1	2 Fermi M2090 per Node	24GB
Eiger	non-Cray	21	300	2x6x1 2x12x1	-Fermi GTX480 -Geforce GTX285 -Tesla s1070 -Fermi c2070 -Fermi m2050	4GB 24GB 48GB
Dom & dommic (R&D cluster)	non-Cray	16	512	2x8x1	K20c, K20X, Xeon Phi (MIC)	32

Generic Resource Plugin

- Mechanics of how to set up Slurm for GPUs.
- Traditionally, resources have been processors and memory (organized into nodes, socket, cores, threads (HW threads)).
- With the advent and increased popularity of GPU's (Graphical Processing Units) this list has now been expanded to include “generic resources” (GRES) which, for the time being, are typically GPU's.
- To enable GPU support within SLURM, the **slurm.conf** must be modified and there must exist a **gres.conf file** (in the same directory as slurm.conf) on each compute node of the system.

slurm.conf

- **GresTypes** a comma delimited list of generic resources to be managed (e.g. *GresTypes=gpu,mic*). This name may be that of an optional plugin providing additional control over the resources.

- **Gres** the specific generic resource and their count associated with each node (e.g. *NodeName=linux[0-999] Gres=gpu:1,mic:2*).

- snippet of Eiger's slurm.conf:

```
GresTypes=gpu
# Individual node configurations
NodeName=eiger180 Feature="fermi,gtx480" Gres=gpu:1
NodeName=eiger181 Feature="fermi,gtx480" Gres=gpu:1
NodeName=eiger[200-204]  Feature="geforce,gtx285" Gres=gpu:1
NodeName=eiger[205-206]  Feature="fermi,gtx480" Gres=gpu:1
NodeName=eiger[240-241]  Feature="tesla,s1070" Gres=gpu:2
NodeName=eiger[242-243]  Feature="fermi,c2070" Gres=gpu:2
NodeName=eiger[207-208]  RealMemory=48000 CoresPerSocket=12 Feature="fermi,m2050" Gres=gpu:2
NodeName=eiger[209-210]  RealMemory=48000 CoresPerSocket=12 Feature="fermi,c2070" Gres=gpu:2
NodeName=eiger[220-223]  RealMemory=48000 Feature="geforce,gtx285" Gres=gpu:1
```

gres.conf

- Must be present on every compute node.
- If all the nodes have the same type of GRES put the gres.conf file in a shared directory is OK otherwise they must be different for every kind of compute node.

```
root@eiger220:/apps/eiger/slurm# cat /etc/gres.conf
###############################################################
# SLURM's Generic Resource (GRES) configuration file
###############################################################
Name=gpu File=/dev/nvidia0
```

- Optionally, a “CPUs=...” clause may be specified telling Slurm which CPU's on the node may access the GPU. If this option is omitted than CPU's on the node should have access to the GPU.

Usage examples:

User can more precisely identify some of the requirements of the GPU's through a “--constraint=” clause.

```
sbatch -N 1 -n 4 --gres=gpu:1 –constraint=”tesla,s1070”
```

The above request is for 1 GPU per node of family ”Tesla s1070”.

```
sbatch -N 1 -n 4 --gres=gpu:1 –constraint=”tesla,s1070|geforce,gtx285”
```

The above request is for one GPU per node and requires the GPU to be of either the “Tesla s1070” or “geforce” family of GPU's.

```
sbatch -N 1 -n 4 --gres=gpu:2 –constraint=”tesla,s1070|geforce,gtx285”
```

The above request is for two GPU per node and requires the GPU to be of either the “Tesla s1070” or “geforce” family of GPU's.

GPU amount of memory

GPU memory is treated just like another generic resource.

- User specifies gpu_mem as an additional gres resource type in the –gres clause:

```
sbatch -N 1 --gres=gpu,gpu_mem:2000
```

- Daint's slurm.conf snippet and gres.conf:

```
GresTypes=gpu,gpu_mem

# Per-node configuration for ROSA XE6 dual-socket nodes: each socket a 16-core xxx node
#NodeName=DEFAULT Sockets=2 CoresPerSocket=8 ThreadsPerCore=2 RealMemory=32768 State=UNKNOWN
NodeName=DEFAULT Sockets=1 CoresPerSocket=8 ThreadsPerCore=2 RealMemory=32768 State=UNKNOWN

# List the 1496 thirty two-way nodes of the compute partition below (service nodes are not allowed to appear)
#NodeName=nid0[0004-0009,0010-0099,0100-0191,0196-0383,0388-0575,0580-0767,0772-0959,0964-0999,1000-1151,1156-1343,1348-1
535,1540-1727,1732-1919,1924-2111,2116-2303]
#NodeName=nid0[0008-0191,0200-0383,0392-0575,0584-0767,0776-0959,0968-1151]
NodeName=nid0[0004-0191,0196-0383,0388-0451,0456-0767,0772-0835,0840-1151,1156-1535,1540-1919,1924-1987,1992-2303,2308-23
71,2376-2687,2692-2755,2760-3071,3076-3139,3144-3455,3460-3523,3528-3839,3844-3907,3912-4223,4228-4291,4296-4607,4612-467
5,4680-4991,4996-5059,5064-5375] Feature="UNKNOWN,gpumodedefault" Gres=gpu_mem:6144,gpu:1
```

```
##############################################################
Name=gpu Count=1 File=/dev/null #File=/dev/nvidia0
Name=gpu_mem Count=6144
```

GPU utilization accounting

- Basic tracking of the number of GPU's **requested** and the number **allocated.**

- Creation of additional fields in the SQL database's tables and the modification of the API of at least the sacctmgr and possibly sacct commands.

```
+--------+----------+----------------------+-----------+
| id_job | gres_req | gres_alloc           | gres_used |
+--------+----------+----------------------+-----------+
| 240648 | gpu:1    | gpu:1,gpu_mem:6144   |           |
| 240649 | gpu:1    | gpu:1,gpu_mem:6144   |           |
| 240656 | gpu:1    | gpu:1,gpu_mem:6144   |           |
+--------+----------+----------------------+-----------+
```

- Future work: the number of GPU's **actually used** by the job. Problematic as there are no known interfaces for Slurm to use to obtain this.

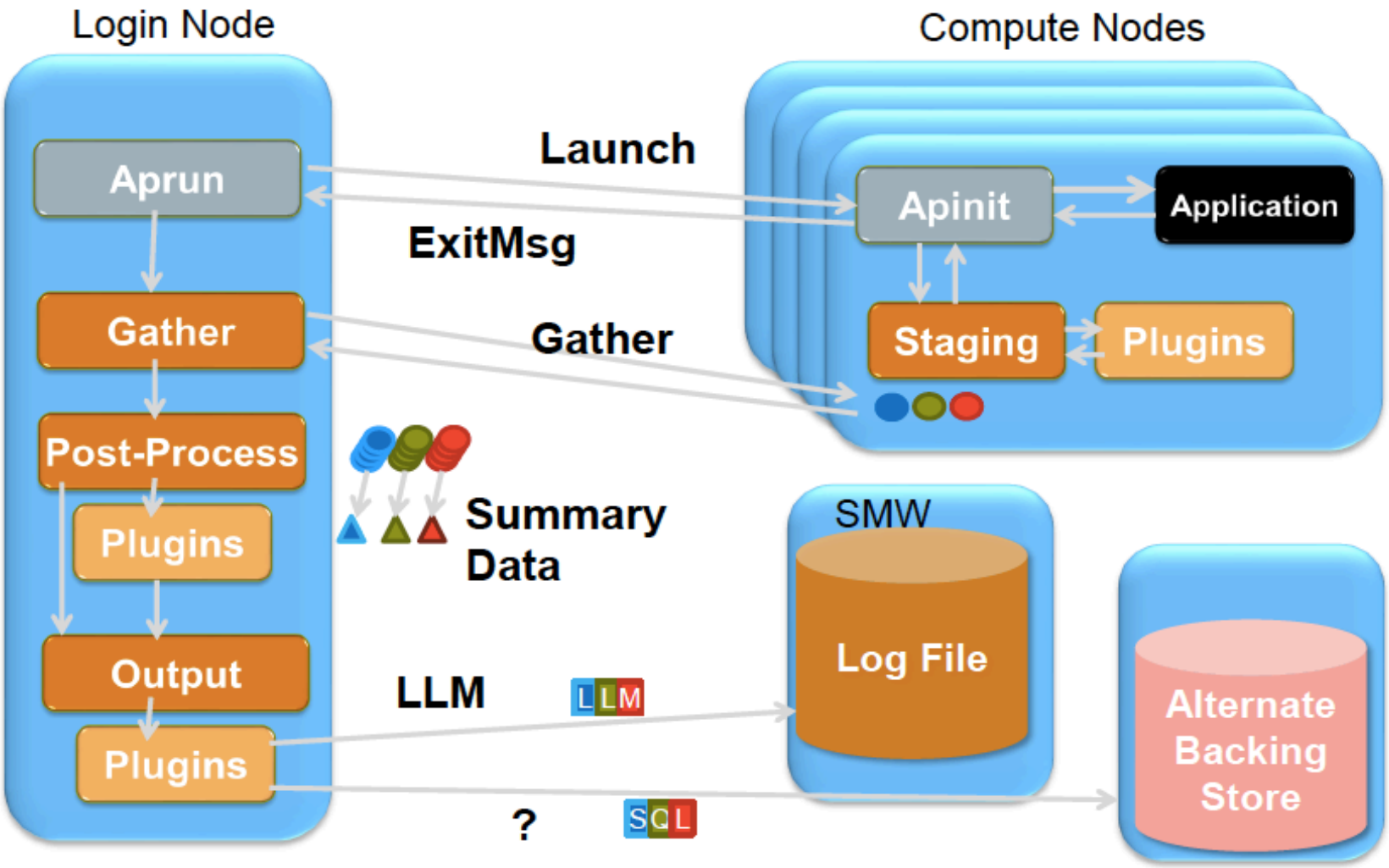
Resource Utilization Reporting RUR

- RUR is a **Cray tool** for gathering statistics on **how system resources are being used by applications**

- RUR is a low-noise, scalable infrastructure that collects compute node statistics **before** an application runs and again **after** it completes.

- The extensible RUR infrastructure allows plugins to be easily written to collect data uniquely interesting to each site administrator. Cray supports plugins that collect process **accounting data, energy usage data, and GPU accounting data**.

Resource Utilization Reporting (RUR)



Resource Utilization Reporting (RUR)

RUR is enabled by default on Piz Daint. With the default setting, outputs are recorded in ~/rur.jobid.

```
uid: 22007, apid: 2320296, jobid: 269975, cmdname: /apps/daint/system/
PE_testing/20140320-1339/idaint/bibw_622_daint taskstats ['utime', 5660000,
'stime', 508000, 'max_rss', 88008, 'rchar', 726
132, 'wchar', 2624, 'exitcode:signal', ['0:0'], 'core', 0]

uid: 22007, apid: 2320296, jobid: 269975, cmdname: /apps/daint/system/
PE_testing/20140320-1339/idaint/bibw_622_daint gpustat ['maxmem', 79167488,
'summem', 158334976, 'gpusecs', 6]

uid: 22007, apid: 2320296, jobid: 269975, cmdname: /apps/daint/system/
PE_testing/20140320-1339/idaint/bibw_622_daint energy ['energy_used', 971]
```

Q+A