Slurm Batch System on HPCF and ECS

Xavier Abellan

xavier.abellan@ecmwf.int

ECMWF

Interactive vs Batch

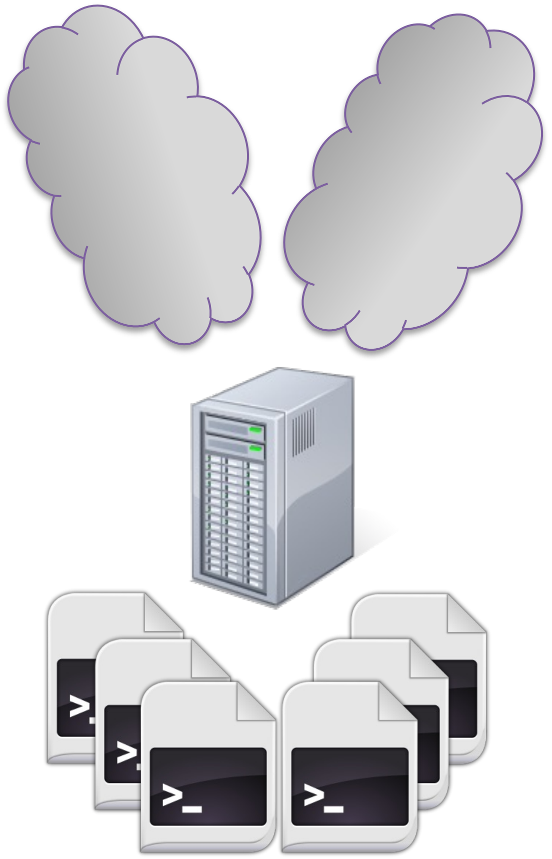
- To run a script or a program **interactively**, enter the executable name and any necessary arguments at the system prompt.

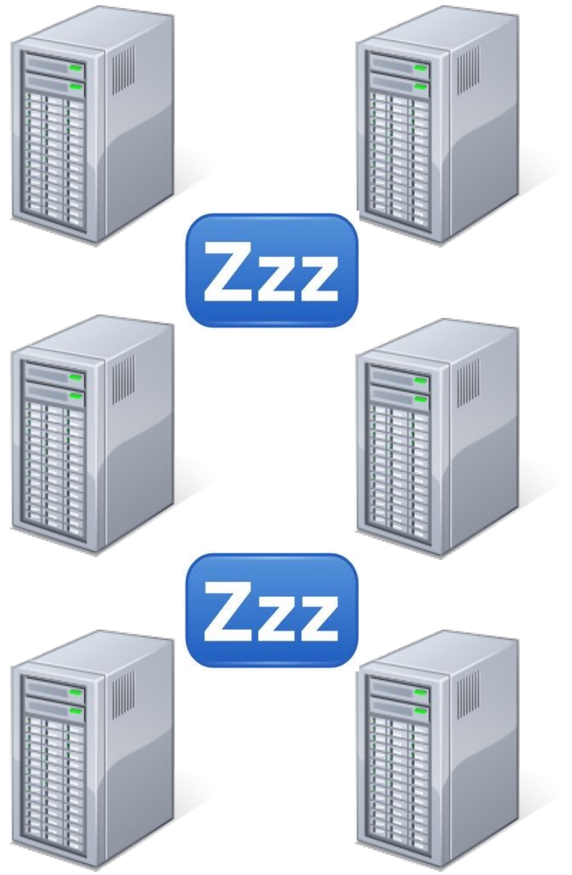
```
$> ./your-program arg1 arg2
```

- You can also run your job in **background** so that other commands can be executed at the same time…

```
$> ./your-program arg1 arg2 &
$>
```

Interactive vs Batch

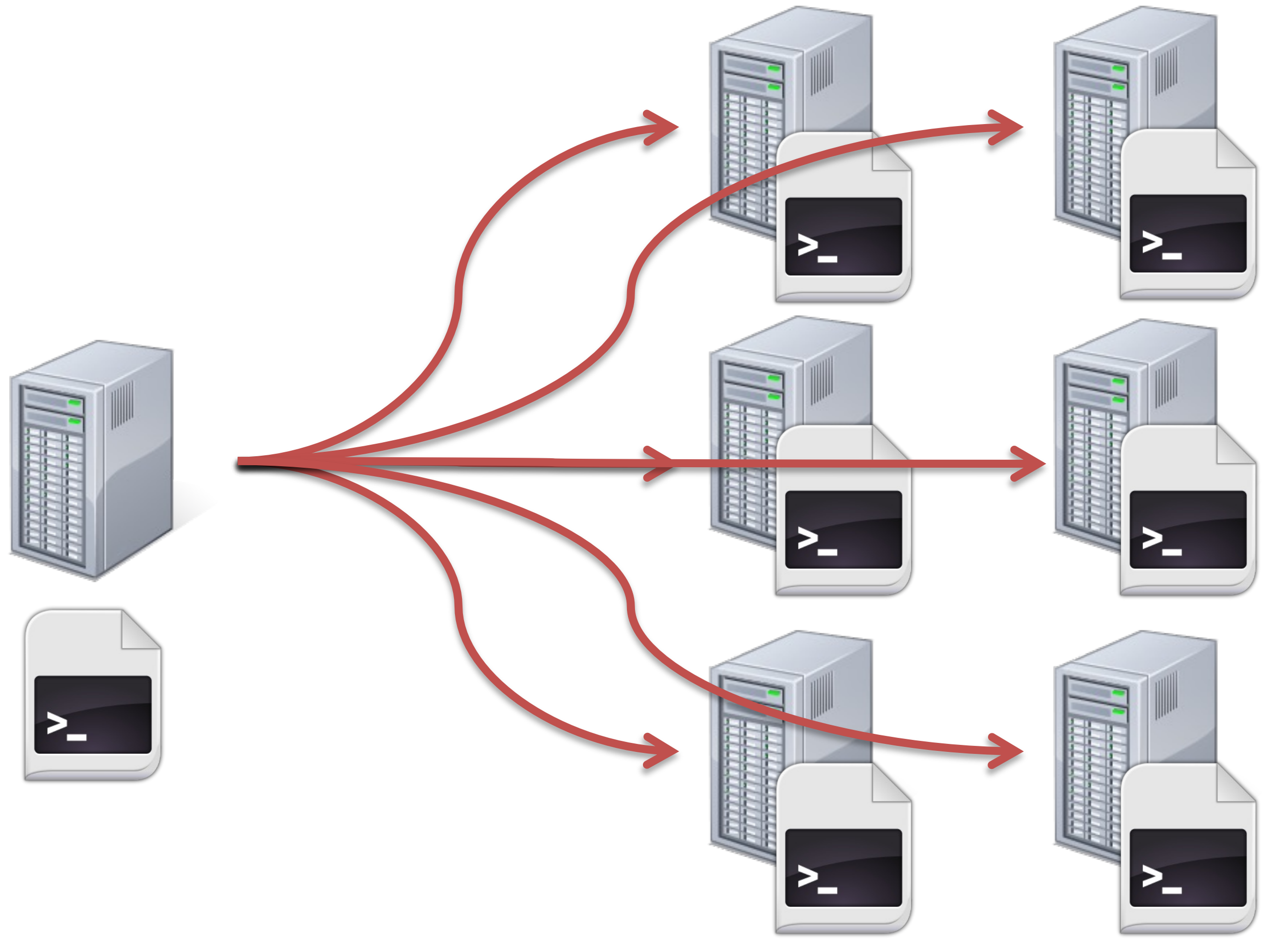




Login node

Computing (batch) nodes

Interactive vs Batch



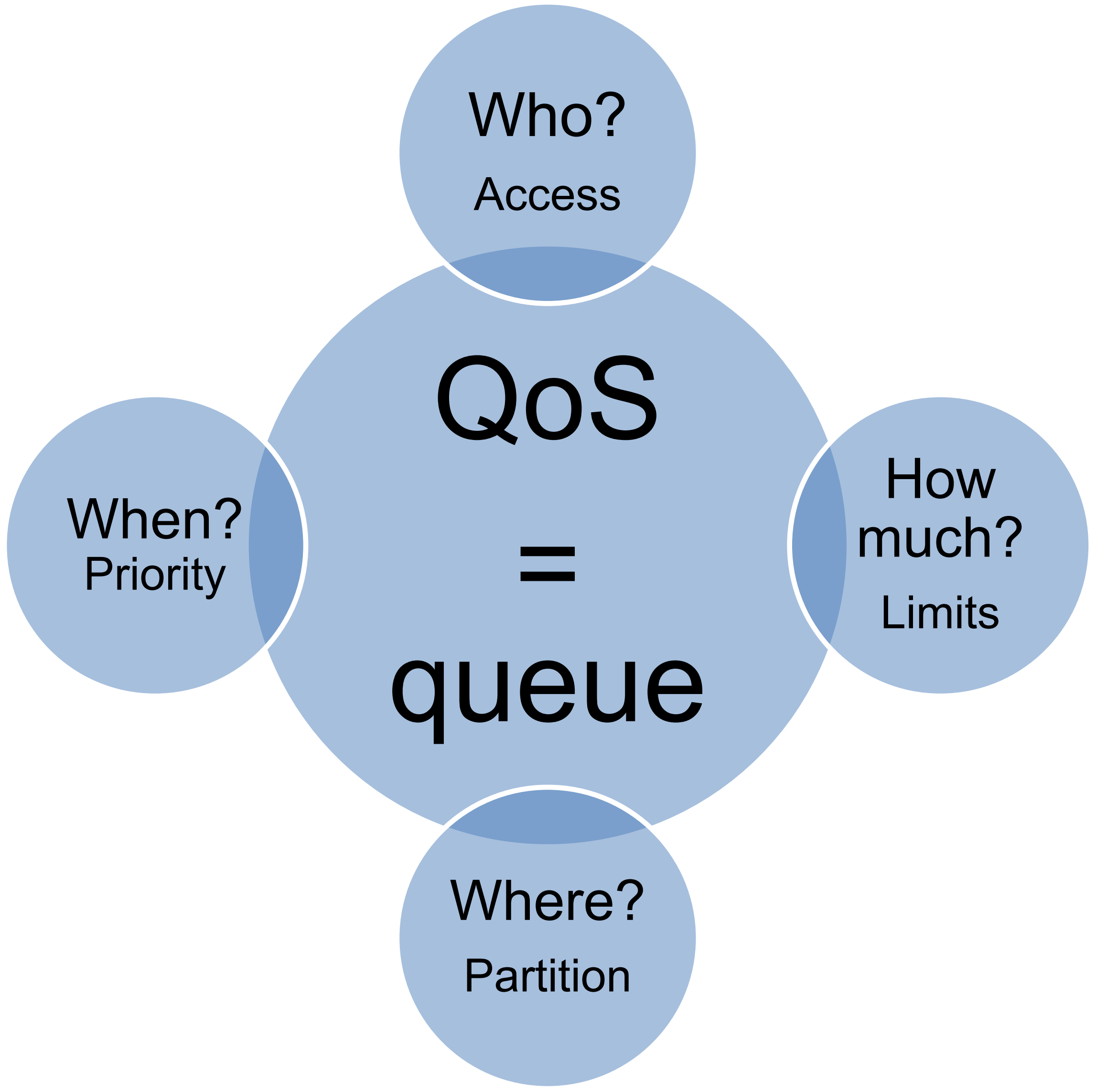
Login node

Computing (batch) nodes

Batch system on Atos HPCF and ECS

- Slurm: Cluster workload manager:
  - Framework to execute and monitor batch work
  - Resource allocation (where?)
  - Scheduling (when?)
- **Batch job**: shell script that will run unattended, with some special directives describing the job itself

Quality of Service (queues)



ECS Quality of Service (standard queues)

QoS	Type	Suitable for...	Shared nodes	Max jobs per user	Default / Max Wall Clock Limit	Default / Max CPUs	Default / Max Memory
ef	fractional	**serial** and **small parallel jobs - DEFAULT**	Yes	-	average job runtime + standard deviation / 2 days	1 / 8	8 GB / 16 GB
ei	interactive	**serial** and **small parallel interactive jobs with ecinteractive**	Yes	1	12 hours / 7 days	1 / 4	8 GB / 8 GB
el	long	**Long-running serial** and **small parallel jobs**	Yes	-	average job runtime + standard deviation / 7 days	1 / 8	8 GB / 16 GB
et	Time-critical Option 1	**serial** and **small parallel Time-Critical jobs. Only usable through ECACCESS Time Critical Option-1**	Yes	-	average job runtime + standard deviation / 12 hours	1 / 8	8 GB / 16 GB

https://confluence.ecmwf.int/x/ZBhbDg

Atos HPCF Quality of Service (standard queues)

QoS	Type	Suitable for...	Shared nodes	Max jobs per user	Default / Max Wall Clock Limit	Default / Max CPUs	Default / Max Memory
nf	fractional	**serial** and **small parallel jobs - DEFAULT**	Yes	-	average job runtime + standard deviation / 2 days	1 / 128	8 GB / 128 GB
ni	interactive	**serial** and **small parallel interactive jobs with ecinteractive**	Yes	1	12 hours / 7 days	1 / 32	8 GB / 32 GB
np	parallel	**parallel jobs** requiring more than half a node	No	-	average job runtime + standard deviation / 2 days	1 / -	240GB per node (all usable mem)

https://confluence.ecmwf.int/x/ZBhbDg

Batch job script

- A job is typically a shell script
  - bash/ksh
- Directives are shell comments:
  - starting with **#SBATCH**
  - Lowercase only
  - No spaces in between
  - No variable expansion
- All directives are optional
  – System defaults in place

```
#!/bin/bash
# The job name
#SBATCH --job-name=helloworld
# Set the error and output files
#SBATCH --output=hello-%J.out
#SBATCH --error=hello-%J.out
# Set the initial working directory
#SBATCH --workdir=/scratch/usxa
# Choose the queue
#SBATCH --qos=ef
# Wall clock time limit
#SBATCH --time=00:05:00
# Send an email on failure
#SBATCH --mail-type=FAIL

# This is the job
echo "Hello World!"
sleep 30
```

Submitting a job: sbatch

- **sbatch:** Submits a job to the system. Job is configured:
  - including the directives in the job script
  - using the same directives as command line options
- The job to be submitted can be specified:
  - As an argument of sbatch
  - If no script is passed as an argument, sbatch will read the job from standard input

```
$> sbatch hello.sh
Submitted batch job 64241253
$> cat hello-64241253.out
Hello world!
$>
```

- The corresponding job id will be returned if successful, or an error if the job could not be submitted

Submitting a job from cron

```
$> ssh hpc-cron
```

```
$> ssh ecs-cron
```

```
$> crontab -e
```

```
05 12 * * * sbatch -Q $HOME/youcronjob
```

General job directives

Directive	Description	Default

`--job-name=…`
`-J …`
	A descriptive name for the job	Script name

`--output=…`
`-o …`
	Path to the file where standard output is redirected. Special placeholders for job id (%j) and the execution node (%N)	slurm-%j.out

`--error=…`
`-e …`
	Path to the file where standard error is redirected. Special placeholders for job id (%j) and the execution node (%N)	output value

`--chdir=…`
`-D …`
	Working directory of the job. The output and error files can be defined relative to this directory.	submitting dir

`--qos=…`
`-q …`
	Quality of service (queue) where the job is to be submitted	nf on Atos HPCF
		ef on ECS

`--time=…`
`-t …`
	Wall clock limit of the job (not cpu time limit!)
	Format: m, m:s, h:m:s, d-h, d-h:m or d-h:m:s
		QoS default

`--mail-type=…`
`-m …`
	Notify user by email when certain event types occur. Valid type values are NONE, BEGIN, END, FAIL, REQUEUE, and ALL	disabled

`--mail-user=…`
`-M …`
	Email address to send the email	submit user

`--account=`
`-A …`
	Project account for the job where the usage will be accounted for.
	Relevant for HPCF only
		Default user project account

https://confluence.ecmwf.int/x/WKrRAg

Resource allocation Job directives

Directive	Description	Default

`--nodes=…`
`-N …`
	Number of nodes for the job	1

`--ntasks=…`
`-n …`
	Number of tasks in the job (i.e. MPI tasks)	1

`--cpus-per-task=…`
`-c …`
	Cpus to allocate per each tasks (i.e. threads, OpenMP)	1

`--tasks-per-node=…`	Tasks to allocate on each node. Useful for parallel tasks requiring considerable memory	fill the node
`--mem=…`	Memory per node	Partition Default

`--hint=multithread`
`--hint=nomultithread`
	Use Hyperthreading
	Don’t use Hyperthreading
		Hyperthreading enabled

`--gres=ssdtmp:<size>G`	Size of the TMPDIR on SSD (fractional jobs only)	3GB

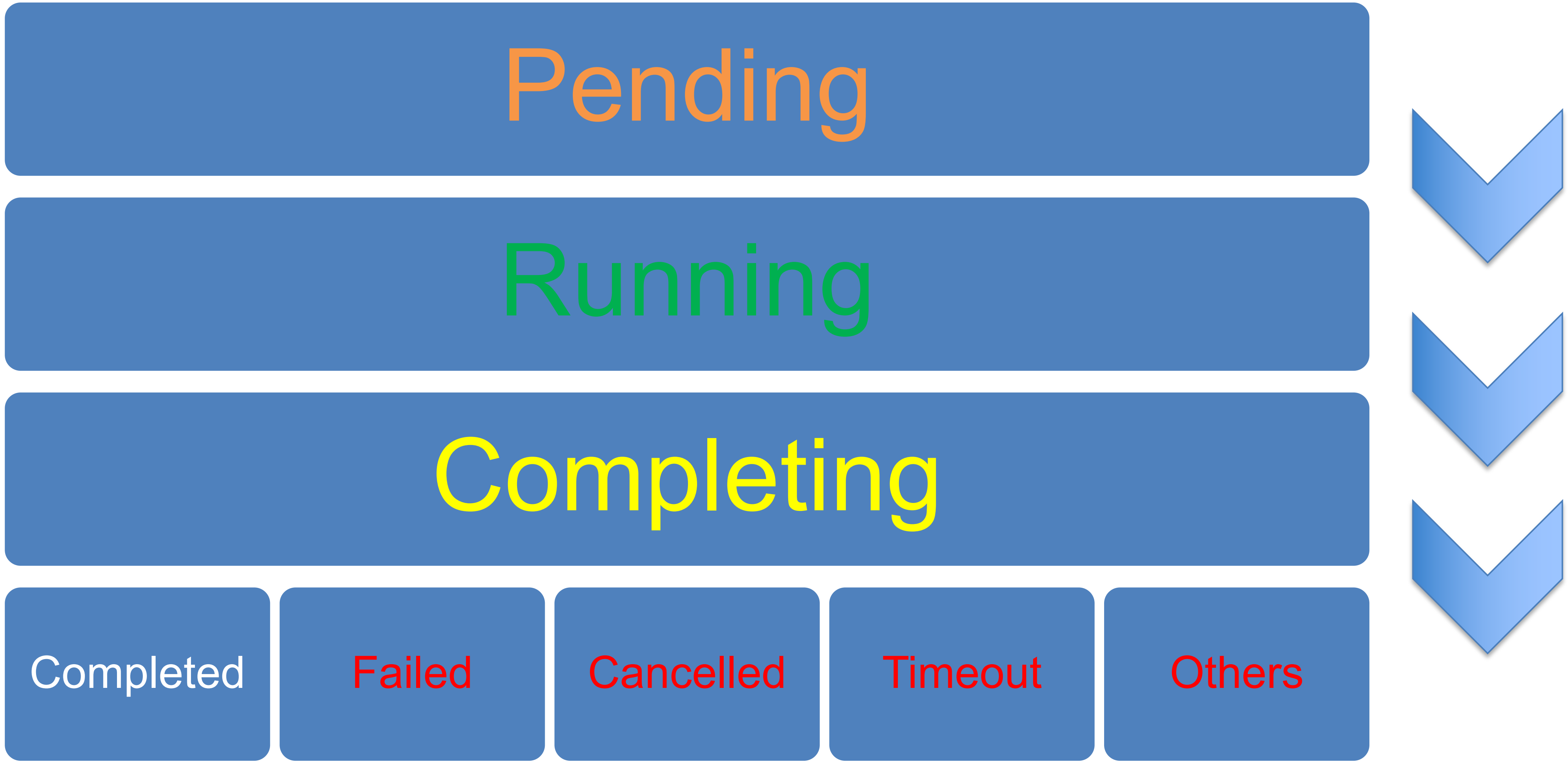
https://confluence.ecmwf.int/x/WKrRAg

Some useful Slurm environment variables

Variable	Description
`$SLURM_JOB_ID`	The Job ID.
`$SLURM_JOB_NODELIST`	Contains the definition (list) of the nodes that is assigned to the job.
`$SLURM_CPUS_PER_TASK`	Number of CPUs per task.
`$SLURM_MEM_PER_CPU`	Memory per CPU. Same as --mem-per-cpu .
`$SLURM_MEM_PER_NODE`	Memory per node. Same as --mem .
`$SLURM_NTASKS`	Same as -n, –ntasks. The number of tasks.
`$SLURM_NTASKS_PER_NODE`	Number of tasks requested per node.
`$SLURM_NNODES`	Total number of nodes in the job's resource allocation.
`$SLURM_ARRAY_JOB_ID`	Job array's master job ID number.
`$SLURM_ARRAY_TASK_ID`	Job array ID (index) number.
`$SLURM_ARRAY_TASK_COUNT`	Total number of tasks in a job array.
`$SLURM_ARRAY_TASK_MAX`	Job array's maximum ID (index) number.
`$SLURM_ARRAY_TASK_MIN`	Job array's minimum ID (index) number.

https://confluence.ecmwf.int/x/WKrRAg

Job States


Pending
Running
Completing
Completed
Failed
Cancelled
Timeout
Others

Checking the queue: squeue

- **squeue:** displays information about the jobs currently **running** or **waiting**

Option	Description
`--me`	**View all my jobs**

`--name <jobname>`
`-n <jobname>`
	View all the jobs with the specified job name

`--state <state>`
`-t <state>`
	View all the jobs that are in the specified state (i.e. PENDING/RUNNING)

`--qos <qos>`
`-q <qos>`
	View all the jobs on the specified QoS

`--account <account>`
`-A <account>`
	View all the jobs on the specified account

`--interactive` `-i`	Interactive option: ask for confirmation before cancelling jobs

`--signal <signal>`
`-s <signal>`
	Signal to send the job instead of SIGKILL

```
$> squeue --me
    JOBID       NAME   USER    QOS    STATE        TIME  TIME_LIMIT NODES      FEATURES NODELIST(REASON)
 64241519 helloworld   usxa     ef  RUNNING        0:03    12:00:00     1        (null)  aa6-203
```

Canceling a job: scancel

- The most common usage of scancel is:

```
$> scancel <jobid1> <jobid2> <jobid3>
```

Option	Description
`--me`	**Cancel all my jobs**

`--name <jobname>`
`-n <jobname>`
	Cancel all my jobs with the specified job name

`--state <state>`
`-t <state>`
	Cancel all my jobs that are in the specified state (i.e. PENDING/RUNNING)

`--qos <qos>`
`-q <qos>`
	Cancel all my jobs on the specified QoS

`--account <account>`
`-A <account>`
	Cancel all my jobs on the specified account

`--interactive` `-i`	Interactive option: ask for confirmation before cancelling jobs

`--signal <signal>`
`-s <signal>`
	Signal to send the job instead of SIGKILL

Canceling a job: scancel

- A job can be cancelled either if it is running or still waiting on the queue
- You will see a message like this in your job error output:

```
slurmstepd: error: *** JOB 64243399 ON ad6-203 CANCELLED AT 2023-10-24T13:41:02 ***
```

Why doesn't my job start?

- Check the last column of the squeue output for a hint…

```
$> squeue -j 64243399
   JOBID       NAME  USER   QOS    STATE       TIME TIME_LIMIT NODES      FEATURES NODELIST(REASON)
64243399 helloworld  user    ef  PENDING       0:00   03:00:00     1        (null) (Priority)
```

Reason	Description
Priority	There are other jobs with more priority
Resources	No free resources are available
AssocMaxJobsLimit	You have reached a limit in the number of jobs you can submit to the system
QOSMaxJobsPerUserLimit	You have reached a limit in the number of jobs you can submit to a QoS
ReqNodeNotAvail	A System Session or outage may be going on. Check our service status on https://www.ecmwf.int/en/service-status

- `man squeue` for the complete list of reason codes

Checking limits

```
scontrol show partition [partition]
```

- Default Wall Clock Time
- Default and Max Memory Per Node
- Overtime Limit

```
sacctmgr show qos
```

- Max Wall Clock Time
- Max Jobs Per User in the QoS
- Max Jobs Submitted Per User in the QoS
- Maximum Resources (TRES) allowed per Job

```
sacctmgr show assoc user=$USER
```

- Maximum Jobs (in any state) per Association*
- Maximum Submitted Jobs per Association*

** Association (complex-partition-account-user)*

Information about past and present jobs: sacct

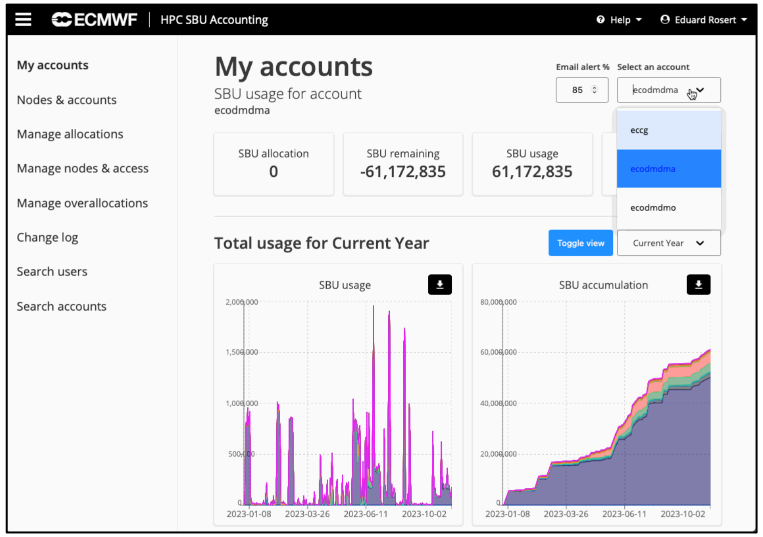
- By default, **sacct** will return information about your jobs that started today

Option	Description
`-j <jobid>`	Show the job with that jobid
`-u <user>`	Show jobs for the specified user. Use option –a for all users
`-E <endtime>`	Show jobs eligible before that date and time
`-S <starttime>`	Show jobs eligible after that date and time
`-s <statelist>`	Show jobs on the states (comma-separated) given during the time period. Valids states are: CANCELLED, COMPLETED, FAILED, NODE_FAIL, RUNNING, PENDING, TIMEOUT
`-q <qos>`	Show jobs only for the qos selected
`-o <outformat>`	Format option. Comma-separated names of fields to display
`-e`	Show the different columns to be used for the –o option
`-X`	Hide the job step information, showing the allocation only

Accounting – HPCF only

- Every job run on HPCF will be charged an amount of **System Billing Units (SBUs)**
    - ECS usage is not charged
- **Check your project account in the job!**
- Examples:
    - A serial job using 1 hour of elapsed time will be charged about 18.91 SBU.
    - A parallel job requesting 2 nodes, running for 3 hours elapsed time will be charged 14522.43 SBU.
- SBUs used by a job reported at the end of the job output
- Check your overall usage at HPC usage portal regularly:
    - https://hpc-usage.ecmwf.int/

```
[ECMWF-INFO -ecepilog] ----------------------------------------------------------
[ECMWF-INFO -ecepilog] This is the ECMWF job Epilogue
[ECMWF-INFO -ecepilog] +++ Please report issues using the Support portal +++
[ECMWF-INFO -ecepilog] +++ https://support.ecmwf.int                     +++
[ECMWF-INFO -ecepilog] ----------------------------------------------------------
[ECMWF-INFO -ecepilog] Run at 2022-08-24T09:09:01 on ac
[ECMWF-INFO -ecepilog] JobName                   : myjob
[ECMWF-INFO -ecepilog] JobID                     : 37015044
[ECMWF-INFO -ecepilog] Submit                    : 2022-08-24T09:08:55
[ECMWF-INFO -ecepilog] Start                     : 2022-08-24T09:08:55
[ECMWF-INFO -ecepilog] End                       : 2022-08-24T09:09:01
[ECMWF-INFO -ecepilog] QueuedTime                : 0.0
[ECMWF-INFO -ecepilog] ElapsedRaw                : 6
[ECMWF-INFO -ecepilog] ExitCode                  : 0:0
[ECMWF-INFO -ecepilog] DerivedExitCode           : 0:0
[ECMWF-INFO -ecepilog] State                     : COMPLETED
[ECMWF-INFO -ecepilog] Account                   : myaccount
[ECMWF-INFO -ecepilog] QOS                       : np
[ECMWF-INFO -ecepilog] User                      : user
[ECMWF-INFO -ecepilog] StdOut                    : /home/user/slurm-37015044.out
[ECMWF-INFO -ecepilog] StdErr                    : /home/user/slurm-37015044.out
[ECMWF-INFO -ecepilog] NNodes                    : 1
[ECMWF-INFO -ecepilog] NCPUS                     : 256
[ECMWF-INFO -ecepilog] SBU                       : 4.083
[ECMWF-INFO -ecepilog] ----------------------------------------------------------
```



My accounts

SBU usage for account
ecodmdma

Email alert %: 85
Select an account: ecodmdma

SBU allocation	SBU remaining	SBU usage
0	-61,172,835	61,172,835

Total usage for Current Year
Current Year

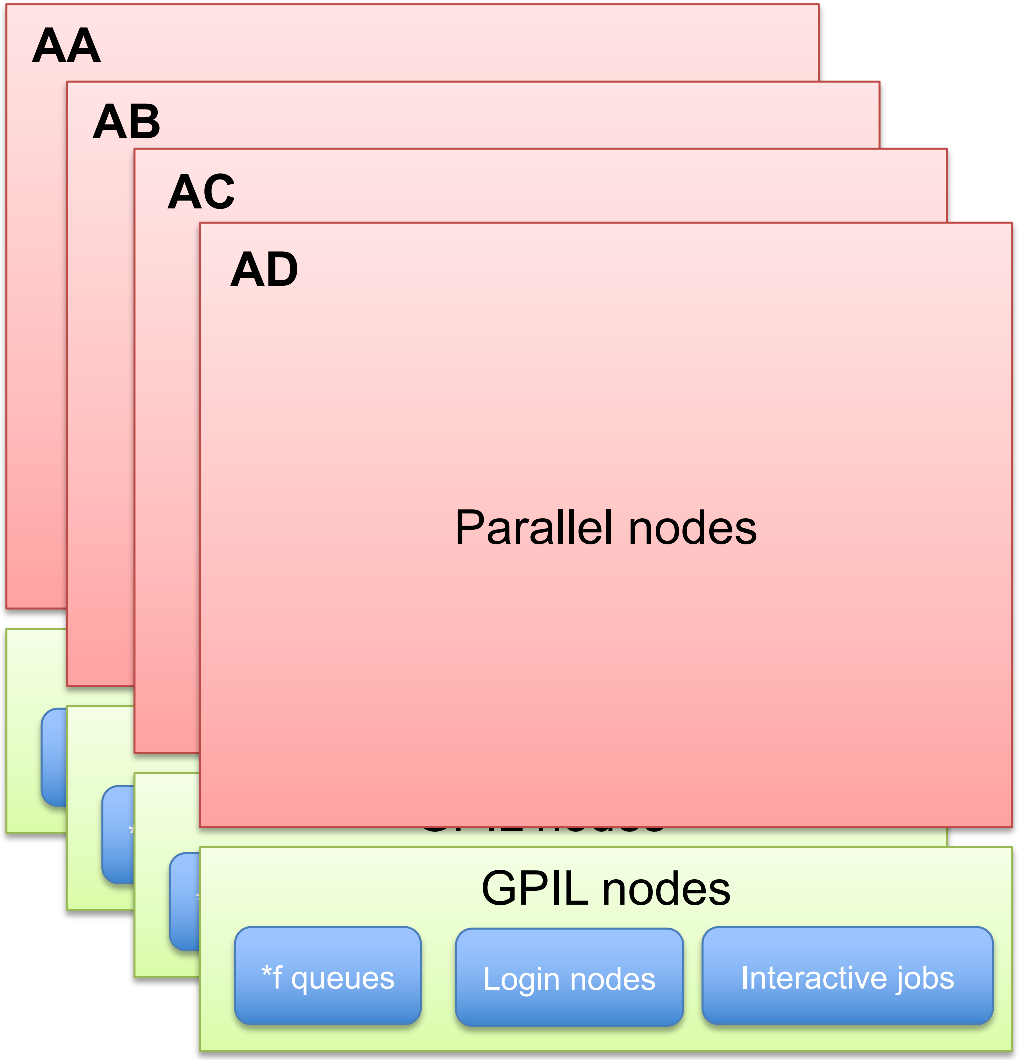
https://confluence.ecmwf.int/x/NrgvEQ

Multi-complex setup

- One Slurm scheduler in each complex
- Sbatch may submit jobs to a different complex
  - System session or outage
- All the other standard Slurm commands will only show information about the local complex
- You may use multi-complex-enabled wrappers:
  - ecsqueue
  - ecscancel

Hint: First digit of Job ID indicates its complex

1... = AA, 2… = AB, 3… = AC, 4… = AD, 6… = ECS



Running parallel workloads: srun

- Spawn parallel applications within a job
- Similar options as **`sbatch`** for resources
- Geometry inherited from job by default
  - cpus-per-task must be always specified
- To be used for MPI, OpenMP or Hybrid
- CPU binding done by default
- Use `--hint=nomultithread` to disable HyperThreading if not needed

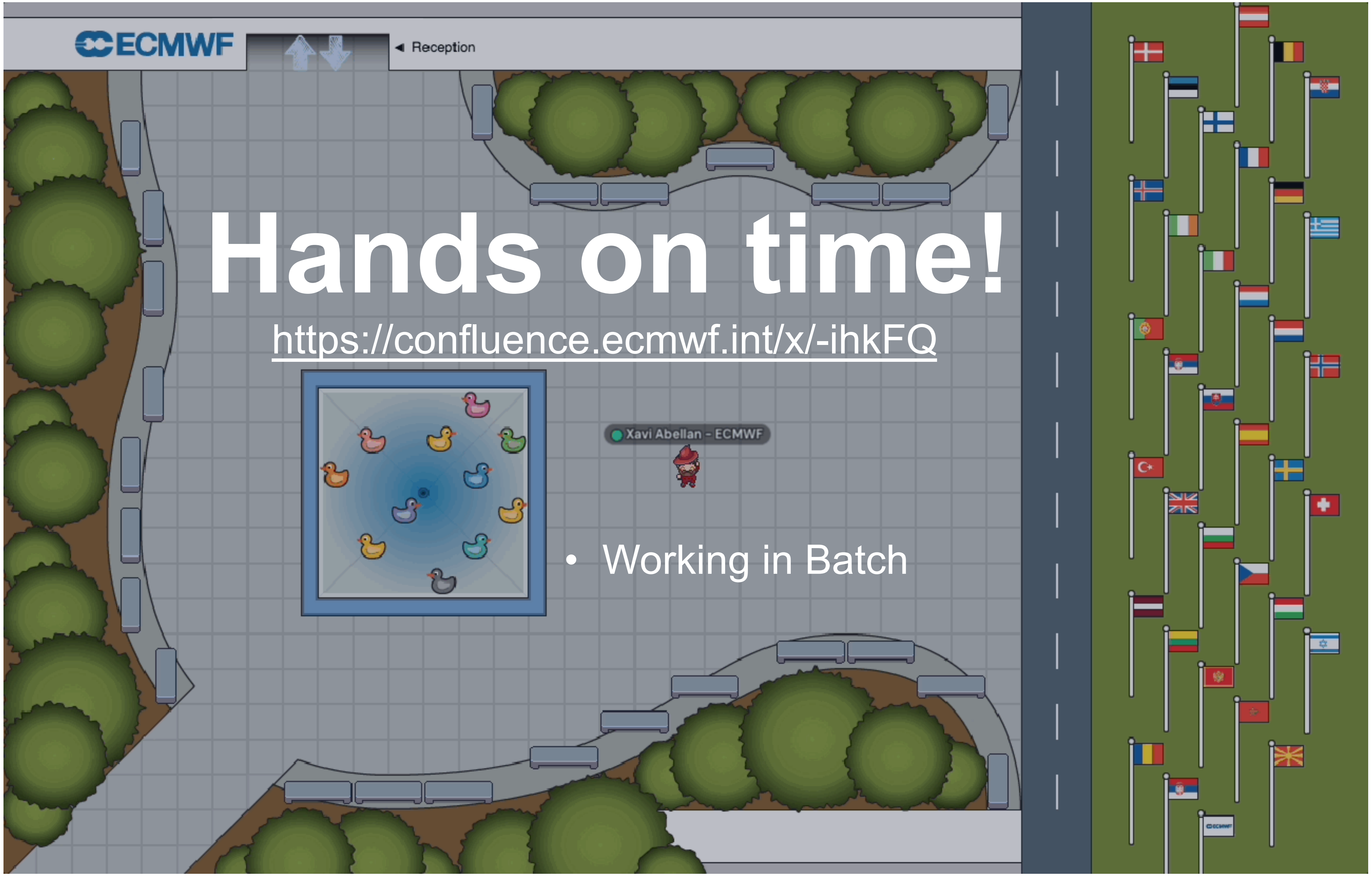
```
#!/bin/bash
#SBATCH --job-name=test-hybrid
#SBATCH --qos=np
#SBATCH --ntasks=128
#SBATCH --cpus-per-task=4
#SBATCH --hint=nomultithread
#SBATCH --time=10:00
#SBATCH --output=test-hybrid.%j.out
#SBATCH --error=test-hybrid.%j.out

# Ensure OpenMP correct pinning
export OMP_PLACES=threads

srun -c $SLURM_CPUS_PER_TASK my_mpi_openmp_app
```

Note for ECS users:
Only small parallel jobs up to 8 CPUs may run on queue ef

Questions?



Hands on time!

https://confluence.ecmwf.int/x/-ihkFQ

- Working in Batch