Introduction to HPC
Overview

• System architecture
• Logging in
• Available Software
• Modules
• Environment
• Preparing jobs
• Monitoring job
• Example scripts
• Getting help
• Applying for an account
Phase 1

Standard Nodes:

• 96 Compute nodes (4 cores per node)
• 8GB RAM per node
• Infiniband High Speed Network
• IBM GPFS Filesystem: 11 TB Storage

Large Memory Nodes:

• 4 Compute Nodes (8 cores per node)
• 32 GB RAM per node
Phase 2

Standard Nodes:

• 416 Compute nodes (8 cores per node)
• 8GB RAM per node
• Infiniband High Speed Network
• IBM GPFS Filesystem: 100 TB Storage

Large Memory Nodes:

• 2 Compute Nodes (24 cores per node)
• 256 GB RAM per node
Phase 3

Standard Nodes:

• 312 Compute nodes (16 cores per node)
• 64GB RAM per node
• Infiniband High Speed Network
• Panasas Parallel Filesystem : 300 TB Storage

Large Memory Nodes:

• 18 Compute Nodes (16 cores per node)
• 256 GB RAM per node

GPU Nodes

• 76 GPU Enabled Nodes: NVIDIA Tesla K20
Phase 4

Standard Nodes:

• 525 Compute nodes (28 cores per node)
• 132GB RAM per node
• Infiniband High Speed Network
• IBM GPFS Filesystem : 300 TB Storage

Large Memory Nodes:

• 18 Compute Nodes (28 cores per node)
• 512 GB RAM per node

GPU Nodes

• 32 GPU Enabled Nodes: Dual NVIDIA Tesla P100
Bluecrystal Phase 1 and 2

Both of these systems have now been retired.

Use Bluecrystal 3 or 4
System Configuration
System Software

Operating System

GNU/Linux (Scientific Linux) BC 1, 2 and 3
GNU/Linux(Centos) BC 4

Queuing System

• Torque, PBS: BC 1 and 2
• Torque, MOAB: BC 3
• SLURM: BC 4
System Software

New to GNU/Linux attend the Introduction to Linux Course

E-mail caroline.gardiner@bristol.ac.uk for details

Or Take a look at the Following Online Tutorials:

www.ee.surrey.ac.uk/Teaching/Unix

http://swcarpentry.github.io/shell-novice/
Phase 1
Phase 2 and 3
Type Of Jobs

Serial Code:

• High Capacity/Throughput Jobs

Parallel Code:

• Requires Additional Programming
• Uses the Infiniband High Speed Network

Two Types of Parallelism:

• MPI Message Passing Interface
• OpenMP
Logging In
Windows
Logging In

Windows
Logging In

Windows
Logging In

Linux and Mac OSX

BC3 disk space

BC3 is running out of available disk space. Within the next few days this will become critical and the performance of the system will drop and users’ jobs may fail. Please could all users of BC3 tidy up their disk space as a matter of urgency and remove any files which they no longer need. It is important that everyone does this within the next 7 days. Thank you for your co-operation.

Simon Burbidge

Monday 04.06.2018:

Git 2.18.0 Update:

Please note that git has been updated on Bluecrystal3 to version 2.18.0. The new module is called:

tools/git-2.18.0

All previous version have been removed.

If you load it in your .bashrc file then please update to the new version.

{iuszczw@newblue4 -}5

Introduction to HPC
Logging In

Log onto Bluecrystal Phase 3:

bluecrystalp3.acrc.bris.ac.uk

Username:
Password:
Logging In

Transferring Data To The HPC Systems

Use:
• scp on Linux and Mac
• WinSCP on Windows
Logging In
Access From Outside The UoB Domain
Available Software

Languages:
- GCC-4.6.4
- Intel Compiler XE
- Intel Compiler XE
- Intel Compiler XE
- Java JDK 1.7.0-40
- Mono-3.0.1
- PERL 5.14.2
- Python 2.6.8
- Python 2.7.5
- Python 2.7.6
- R 2.15.1
- R 3.0.2

Libraries:
- GNU:
  - ATLAS
  - FFTW 3.3.4
  - GSL 1.16
- Intel:
  - ATLAS

Tools:
- CMAKE 2.8.1
- CMAKE 2.8.12
- GIT 1.8.4.2
- Subversion-1.8.4

Profiling:
- Intel VTune
- TAU 2.23
## Available Software

**Applications:**

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abaqus 6.12</td>
<td></td>
</tr>
<tr>
<td>Amber 12</td>
<td></td>
</tr>
<tr>
<td>Beast 1.7.5</td>
<td></td>
</tr>
<tr>
<td>Comsol 4.3b</td>
<td></td>
</tr>
<tr>
<td>GROMACS 5.0</td>
<td></td>
</tr>
<tr>
<td>GCTA 1.24.3</td>
<td></td>
</tr>
<tr>
<td>LS-DYNA 971R6.1.0</td>
<td></td>
</tr>
<tr>
<td>Matlab R2013b</td>
<td></td>
</tr>
<tr>
<td>Meep-1.2.1</td>
<td></td>
</tr>
<tr>
<td>NASTRAN 2012.1</td>
<td></td>
</tr>
<tr>
<td>Netlogo-5.0.5</td>
<td></td>
</tr>
<tr>
<td>OpenBabel 2.3.2</td>
<td></td>
</tr>
<tr>
<td>ParaView 4.0.1</td>
<td></td>
</tr>
<tr>
<td>PAMLM 4.7</td>
<td></td>
</tr>
<tr>
<td>PhyloBayes-3.3f</td>
<td></td>
</tr>
<tr>
<td>Plink-1.0.7</td>
<td></td>
</tr>
<tr>
<td>QuantumEspresso-5.1</td>
<td></td>
</tr>
<tr>
<td>Scilab 5.4.1</td>
<td></td>
</tr>
<tr>
<td>Trinity 2013.8.14</td>
<td></td>
</tr>
</tbody>
</table>

---

**Introduction to HPC**

22
Available Software

If there is any software that you need that’s not already installed contact us and we’ll install it for you.

This applies to Python and R packages as well
Modules

Module Commands

module avail

module add  module-name

module del  module-name

module list  module-name

Remember, modules that are required by a job need to be added to your .bashrc file
Environment

[iszcjw@newblue4 ~]$ pwd
/gpfs/cluster/isys/iszcjw

[iszcjw@newblue4 ~]$ ls -l .bashrc
-rw-r--r-- 1 iszcjw isys 7746 Aug 29 15:32 .bashrc

[iszcjw@newblue4 ~]$ more .bashrc
#/ .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
   . /etc/bashrc
fi

module add shared moab/5.2.3 torque/2.3.3
module add languages/R-2.15.1
etc.
Preparing Jobs

Steps Required To Run A Job

• Ensure the required application module is included in your .bashrc file

• Or Compile your code (If Required)

• Copy Any Required Data Onto The System

• Create a Job Submission Script

• Submit The Job Script To The Queuing System
Queuing System

How The Queuing System Works

• The job script contains the commands required to run the job
• Submit the job script to the queuing system
• The queuing system then executes the commands in the script on the compute nodes

• Don’t expect your jobs to start instantly
• The Queuing system runs a fair share policy
• Users with a lot of jobs can not take over the system
• Get jobs in the queue sooner rather than later
Workshop Package

Copy The Workshop tar File Into Your $HOME Directory

```
iszcjw@newblue2 ~]$ cd
iszcjw@newblue2 ~]$ cp ../workshop.tar .
iszcjw@newblue2 ~]$ ls -l workshop.tar
-rwlr-xr-x 1 iszcjw isys 10240 Sep 25 14:23 workshop.tar
iszcjw@newblue2 ~]$ 
```

Unpack The tar File

```
iszcjw@newblue2 ~]$ tar xvf workshop.tar
./workshop/job1.sh
./workshop/job2.sh
./workshop/job3.sh
./workshop/job4.sh
./workshop/job5.sh
iszcjw@newblue2 ~]$ 
```
Simple Job Scripts

Change Directory into workshop

[iszcjw@newblue2 ~]$ cd workshop
[iszcjw@newblue3 workshop]$ more job1.sh
#!/bin/bash
#
#
# Define working directory
export WORK_DIR=$HOME/workshop

# Change into working directory
cd $WORK_DIR

# Execute code
/bin/hostname
Queuing System Commands

qsub \textit{job\_script}

qstat \textit{job\_id\_number}

qdel \textit{job\_id\_number}

showstart \textit{job\_id\_number}

showq
Submit your first job

[iszcjw@newblue4 workshop]$ qsub
job1.sh
2630148.bluequeue1.cvos.cluster
[iszcjw@newblue4 workshop]$
Queuing System Commands

```
[iszcjw@newblue1 ~]$ qstat 8154519
Job ID    Name     User      Time     Use    S  Queue
-----------------  ---------  ---------  ------  ----  ---
8154519.master   hmbio_02  hm17547  06:24:12  R  veryshort
[iszcjw@newblue1 ~]$```

```
32
Introduction to HPC
```
Queuing System Commands

Default output files

-rw------ 1 iszcjw isys  0 Feb 25 15:53 job5.sh.e8154855
-rw------ 1 iszcjw isys  616 Feb 25 15:53 job5.sh.o8154855
Man pages:
Help with Commands

man command
Man pages:

### qstat (1B)

**NAME**

qstat - show status of PBS batch jobs

**SYNOPSIS**

```bash
[job_identifier... | destination...]
qstat -Q [-f] [-l] [-M] [site_specific] [-1] [destination...]
qstat -Q [-G] [-M] [-1] [destination...]
qstat -B [-f] [-l] [-M] site_specific | -1 | [server_name...]
qstat -c
```

**DESCRIPTION**

The `qstat` command is used to request the status of jobs, queues, or a batch server. The requested status is written to standard out.

When requesting job status, synopsis format 1 or 2, `qstat` will output information about each job identifier or all jobs at each destination. Jobs for which the user does not have status privilege are not displayed.

When requesting queue or server status, synopsis format 3 through 5, `qstat` will output information about each destination.

**OPTIONS**

- `-f` Specifies that a full status display be written to standard out.
- `-a` "All" jobs are displayed in the alternative format, see the standard output section. If the operand is a destination id, all jobs at that destination are displayed. If the operand is a job id, information about that job is displayed.
Queuing System Commands

[isz cjw@newblue1 ~]$ qstat -an 8154519

master.cm.cluster:

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>8154519.master.cm.cluster</td>
<td>hm17547</td>
<td>veryshor</td>
<td>hmbio_02</td>
<td>58543</td>
<td>1</td>
<td>16</td>
<td>--</td>
<td>02:00:00</td>
<td>R</td>
<td>00:28:49</td>
</tr>
<tr>
<td>node47-009/0+node47-009/1+node47-009/2+node47-009/3+node47-009/4+node47-009/5+</td>
<td>node47-009/6+node47-009/7+node47-009/8+node47-009/9+node47-009/10+node47-009/11+</td>
<td>node47-009/12+node47-009/13+node47-009/14+node47-009/15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[isz cjw@newblue1 ~]$
Monitoring Jobs

Use the –an1 switch on qstat to find where the job is running

```
[iszczw@newblue1 ~]$ qstat -an1 8154519
```

```
master.cm.cluster:

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>8154519.master.cm.cluster:</td>
<td>hm17547 veryshor hmbio_02</td>
<td>58543</td>
<td>1 16 --</td>
<td>02:00:00</td>
<td>R</td>
<td>00:28:49</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Then log into that node and run top

```
[iszczw@newblue1 workshop]$ ssh node47-009
Last login: Thu Oct 3 13:10:56 2013 from newblue3.cm.cluster
[iszczw@node47-009 ~]$ top
```
Monitoring Jobs

Tasks: 522 total, 18 running, 504 sleeping, 0 stopped, 0 zombie
Cpu(s): 8.0%us, 2.4%sy, 36.9%ni, 52.6%id, 0.1%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 66044848k total, 2370064k used, 63674784k free, 15540k buffers
Swap: 11999228k total, 180540k used, 11818688k free, 538092k cached

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>58741</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>674m</td>
<td>90m</td>
<td>54m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:51.73</td>
<td>explicit</td>
</tr>
<tr>
<td>58742</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>674m</td>
<td>90m</td>
<td>54m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:44.46</td>
<td>explicit</td>
</tr>
<tr>
<td>58743</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>674m</td>
<td>90m</td>
<td>54m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:52.49</td>
<td>explicit</td>
</tr>
<tr>
<td>58744</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>674m</td>
<td>90m</td>
<td>54m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:40.92</td>
<td>explicit</td>
</tr>
<tr>
<td>58745</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>674m</td>
<td>90m</td>
<td>54m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:53.04</td>
<td>explicit</td>
</tr>
<tr>
<td>58746</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>674m</td>
<td>90m</td>
<td>54m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:37.31</td>
<td>explicit</td>
</tr>
<tr>
<td>58747</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>675m</td>
<td>92m</td>
<td>55m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:53.70</td>
<td>explicit</td>
</tr>
<tr>
<td>58748</td>
<td>hm17547</td>
<td>39</td>
<td>19</td>
<td>675m</td>
<td>91m</td>
<td>55m</td>
<td>R</td>
<td>99.4</td>
<td>0.1</td>
<td>33:36.00</td>
<td>explicit</td>
</tr>
</tbody>
</table>
Etc ....
Simple Job Script

[iszczw@newblue3 workshop]$ more job2.sh
#!/bin/bash
#
#
#PBS -l nodes=1:ppn=1,walltime=1:00:00

# Define working directory
export WORK_DIR=$HOME/workshop

# Change into working directory
cd $WORK_DIR

# Execute code
/bin/hostname
[iszczw@newblue3 workshop]$ more job3.sh
#!/bin/bash
#
#
#PBS -l nodes=1:ppn=1,walltime=1:00:00

# Define working directory
export WORK_DIR=$HOME/workshop

# Define executable
export EXE=/bin/hostname

# Change into working directory
cd $WORK_DIR

# Execute code
$EXE
Simple Job Script

[iszczw@newblue3 workshop]$ more job4.sh
#!/bin/bash
#
#PBS -l nodes=1:ppn=1,walltime=1:00:00
# Define working directory
export WORK_DIR=$HOME/workshop
# Define executable
export EXE=/bin/hostname
# Change into working directory
cd $WORK_DIR

echo JOB ID: $PBS_JOBID

echo Working Directory: `pwd`
echo Start Time: `date`
# Execute code
$EXE

echo End Time: `date`
Submit A Parallel Job

Compare parallel and serial code: Serial hello world program

[iszczw@newblue1 workshop]$ more hello.c
#include <stdio.h>
int main()
{
    // printf() displays the string inside quotation
    printf("Hello, World! \n");
    return 0;
}
Submit A Parallel Job

Compile and run:

[iszczw@newblue1 workshop]$ gcc hello.c -o hello
[iszczw@newblue1 workshop]$ ls -l hello
-rwxr-xr-x 1 iszczw isys 6491 Feb 25 18:56 hello

[iszczw@newblue1 workshop]$ ./hello
Hello, World!
Submit A Parallel Job

Parallel hello world program:

```c
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
    // Initialize the MPI environment. The two arguments to MPI Init are not
    // currently used by MPI implementations, but are there in case future
    // implementations might need the arguments.
    MPI_Init(NULL, NULL);

    // Get the number of processes
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);

    // Get the rank of the process
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

    // Get the name of the processor
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Print off a hello world message
    printf("Hello world from processor %s, rank %d out of %d processors\n",
            processor_name, world_rank, world_size);

    // Finalize the MPI environment. No more MPI calls can be made after this
    MPI_Finalize();
}
```
Submit A Parallel Job

First compile the code:

```bash
[iszcjw@newblue1 workshop]$ mpicc hello.mpi.c -o hello

[iszcjw@newblue1 workshop]$ ls -l hello
-rwxr-xr-x 1 iszcjw isys 7864 Sep 26 11:14 hello

[iszcjw@newblue1 workshop]$ mpirun -np 4 -machinefile host ./hello
```
Simple Parallel Job

[iszczw@newblue3 workshop]$ more job4.sh
#!/bin/bash
#
#
#PBS -l nodes=2:ppn=4,walltime=1:00:00
# Define working directory
export WORK_DIR=$HOME/workshop

# Define executable
export EXE=$HOME/workshop/hello
# Change into working directory
cd $WORK_DIR

Continued on next slide
Simple Parallel Job

# Generate the list of nodes the code will run on --------------

cat $PBS_NODEFILE
export nodes=`cat $PBS_NODEFILE`
export nnodes=`cat $PBS_NODEFILE | wc -l`
export confile=inf.$PBS_JOBID.conf

for i in $nodes; do
    echo ${i} >>$confile
done

# Execute the code -----------------------------------------------

mpirun -np $nnodes -machinefile $confile $EXE
Submit A Parallel Job

Submit the parallel job script:

```
[iszczw@newblue1 workshop]$ qsub job5.sh
2630626.bluequeue1.cvos.cluster
[iszczw@newblue1 workshop]$`

Example Scripts

In order to run a number of parallel Abaqus jobs we can do the following:
Assume all the input file *.inp are in the working directory

Copy the following job submission template script into the working directory:

cut here ---------------------------------------------
#!/bin/bash
#
#
#-----------------------------------------------
#PBS -l walltime=12:00:00,nodes=1:ppn=4
#
#PBS -q abaqus

# 1. Edit this
export MYDIR "${HOME}/Test/loop_test"
#
#-----------------------------------------------
#
cd $MYDIR
#-----------------------------------------------
Example Scripts

Create a main.sh script containing the following:

```
#!/bin/bash
# Find each input file
# and strip off .inp to avoid confusing the script
for f in `ls *.inp | sed s/.inp//`
do

# Create a job script for each .inp file
cp qabaqus.parallel.sh qabaqus.parallel.sh.$f

# Add the execution line to the end of the job script
echo "abaqus job="$f  "cpus=4 analysis double interactive" >> qabaqus.parallel.sh.$f

# Submit the job script to the queue
qsub qabaqus.parallel.sh.$f

sleep 10

done
```

The above script searches the current directory for input files and creates a unique job submission script for each from the job submission script template. It then submits each of the unique job submission scripts to the queue.
Array Jobs

Array jobs allow us to submit a number of jobs with a single command

#!/bin/bash
#
#
#PBS -l nodes=2:ppn=4,walltime=1:00:00

# Define working directory
export WORK_DIR=$HOME/workshop

# Define executable
export EXE=$HOME/workshop/hello.$PBS_ARRAYID

# Change into working directory
cd $WORK_DIR

Continued on next slide
Array Jobs

Continued

# Generate the list of nodes the code will run on -----------------------

cat $PBS_NODEFILE
export nodes=`cat $PBS_NODEFILE`
export nnodes=`cat $PBS_NODEFILE | wc -l`
export confile=inf.$PBS_JOBID.conf

for i in $nodes; do
echo ${i} >>$confile
done

# Execute the code -------------------------------------------------------

mpirun -np $nnodes -machinefile $confile $EXE
Array Jobs

Submit an array job

[iszczjw@bigblue4 workshop]$ qsub -t 1-3 job6.sh
2631674.bluequeue1.cvos.cluster
[iszczjw@bigblue4 workshop]$ qstat -u iszczjw

bluequeue1.cvos.cluster:

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2631674-1.bluequ</td>
<td>iszczjw</td>
<td>veryshor</td>
<td>job6.sh-1</td>
<td>--</td>
<td>2</td>
<td>--</td>
<td>--</td>
<td>01:00 R</td>
<td>--</td>
</tr>
<tr>
<td>2631674-2.bluequ</td>
<td>iszczjw</td>
<td>veryshor</td>
<td>job6.sh-2</td>
<td>--</td>
<td>2</td>
<td>--</td>
<td>--</td>
<td>01:00 R</td>
<td>--</td>
</tr>
<tr>
<td>2631674-3.bluequ</td>
<td>iszczjw</td>
<td>veryshor</td>
<td>job6.sh-3</td>
<td>--</td>
<td>2</td>
<td>--</td>
<td>--</td>
<td>01:00 R</td>
<td>--</td>
</tr>
</tbody>
</table>
Array Jobs

[iszczw@bigblue4 workshop]$ qstat -an1 2631674-3

bluequeue1.cvos.cluster:

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Memory</th>
<th>Time</th>
<th>S Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2631674-3.blueque</td>
<td>iszcjw</td>
<td>veryshor job6.sh-3</td>
<td>--</td>
<td>2</td>
<td>--</td>
<td>01:00</td>
<td>R</td>
<td>--</td>
<td>u03n007+u03n007+u03n007+u03n007+u03n009+u03n009+u03n009+u03n009</td>
</tr>
</tbody>
</table>
Matlab Script

Matlab jobs must be run through the queing system not the Matlab GUI

#!/bin/bash
#
#PBS -l walltime=1:00:00
#PBS -j oe
#PBS -q testq

# Change into the working directory
cd /exports/gpfs/iszcjw/Test/matlab

# Execute the code
matlab -nodisplay -nojvm -nodesktop -nosplash < test.m
Getting Help

ACRC Website

https://www.acrc.bris.ac.uk

Service Desk

hpc-help@bristol.ac.uk
Applying For An Account

ACRC Website

https://www.acrc.bris.ac.uk
Application Form

BlueCrystal Application

Personal details
- First Name: Callum
- Surname: Wright
- Email: C.Wright@bristol.ac.uk
- Department: IT Services
- Faculty: Other
- Institution: University of Bristol
- Telephone: (0117) 331 4319
- User Type: STAFF

Project details
Staff may submit a new project proposal or choose to join an existing project. To join an existing project you will need to know the project code, which is created and advised to the user when a project is approved.

- [ ] Join an existing project
- [ ] Create a new project
- Project code:

Additional information
The following information is not essential, but if known, will help us to assess the requirements for your job

Preferred log-in shell: [ ] bash

You may use the box to provide details of the code you wish to run and, if known, compilation details and platform dependencies.
Application Form

Project details

Staff may submit a new project proposal or choose to join an existing project. To join an existing project you will need to know the project code, which is created and advised to the user when a project is approved.

Join an existing project ☐
Create a new project ☐

Project title *
Estimated CPU usage * (kilohours)
Estimated Disk usage * (GB)
Estimated duration of project * (Months)
Funding * (Please select...)

Project proposal (500 Words Max)