Job submission

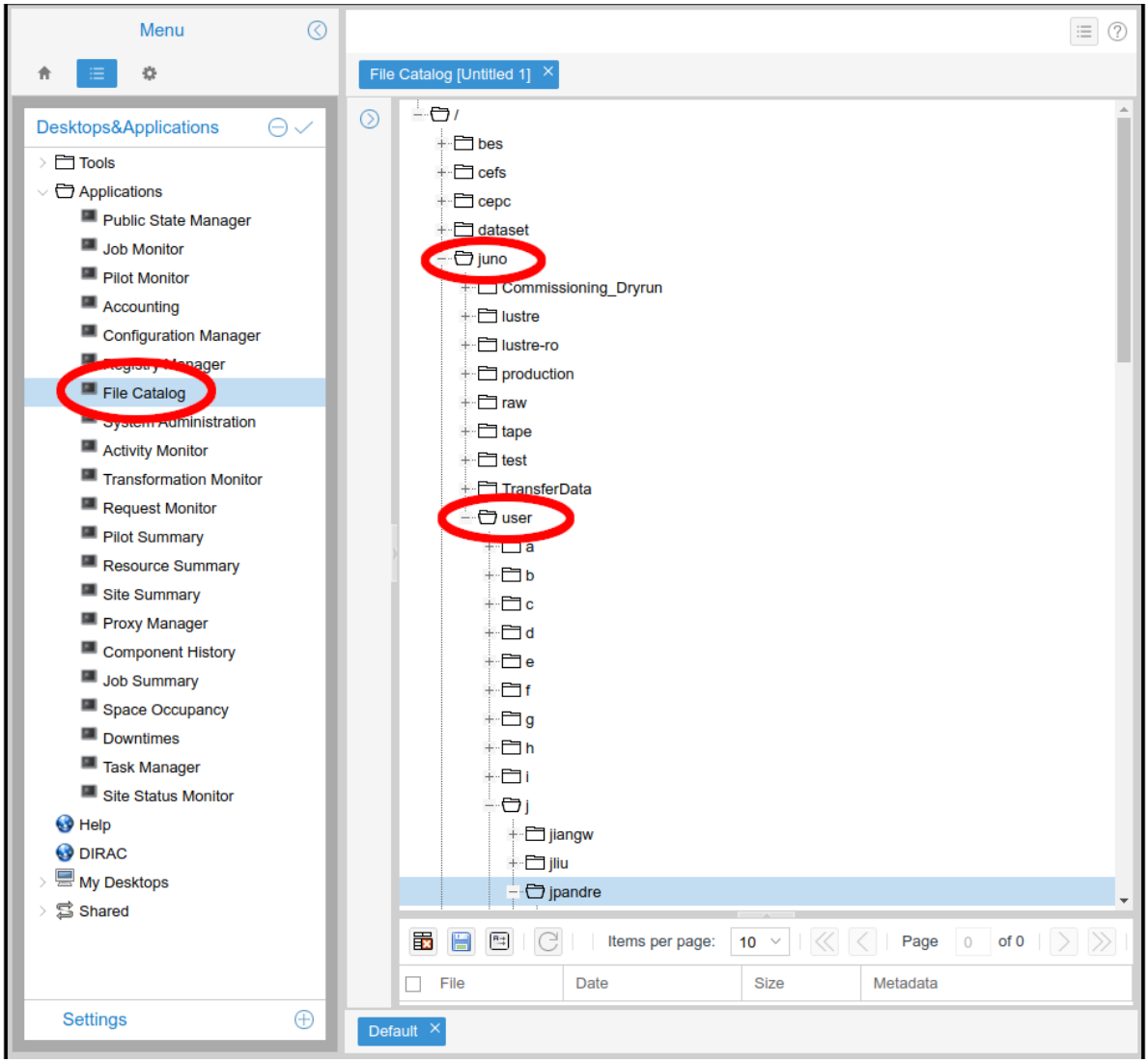
João Pedro Athayde Marcondes de André

May 2023

1. Standard submission (JDL)
   - Submit generic job
   - Submit JUNO jobs
2. JSUB
   - Submit JUNO jobs
3. Production tool
   - Not covered today – see DocDB-8164
   - To use production tool need to be in `production` group in VOMS

Before starting. . .

- This time we inverted the job submission & data access parts. . .
- However to check job output it's useful to check files in file catalog
- So, for now please use the DIRAC web interface (`https://dirac.ihep.ac.cn/`) to do that!



- After job submission you'll learn to access the data from terminal.

Part #1
Standard submission (JDL)

Submitting your first job

`myscript.sh`

```
#!/bin/sh

echo "=====  Begin  ====="
date
echo "The program is running on $HOSTNAME"
```

`test.jdl`

```
JobName = "mysimplejob";
Executable = "/bin/bash";
Arguments = "myscript.sh";
StdOutput = "stdout.out";
StdError = "stderr.err";
InputSandbox = { "myscript.sh" };
OutputSandbox = { "stdout.out", "stderr.err" };
VirtualOrganisation = "vo.juno.ihep.ac.cn";
```

- Create the `myscript.sh` and `test.jdl` files
- Submit job:

```
% dirac-wms-job-submit test.jdl
JobID = 5923594
```

Job status

% dirac-wms-job-submit test.jdl
JobID = 5923594

% dirac-wms-job-status 5923594
JobID=5923593 Status=Waiting; MinorStatus=Pilot Agent Submission; Site=ANY;

% dirac-wms-job-status 5923594
JobID=5923594 Status=Done; MinorStatus=Execution Complete; Site=CLOUD.IHEPCLOUD.cn;

% dirac-wms-job-get-output 5923594
Job output sandbox retrieved in /afs/ihep.ac.cn/users/j/jpandre/dirac/5923594/

% cat 5923594/stdout.out
===== Begin =====
Sun Jul 21 10:01:13 CST 2019
The program is running on idirac-20190721-095925-12c787ff

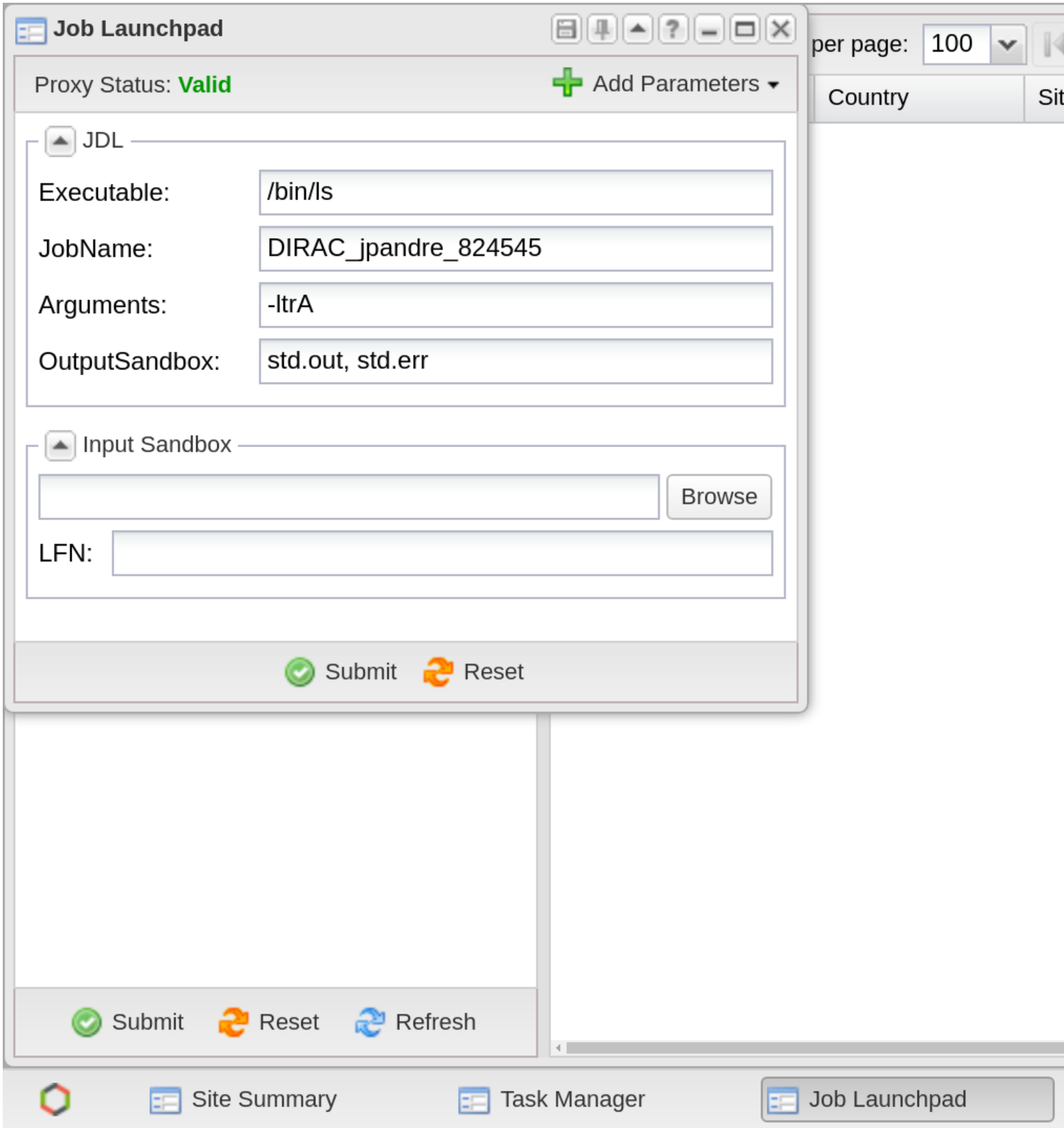
DIRAC API (advanced)

- DIRAC API written in python
- Python API provides more flexibility than pre-made commands
- CLI tools written in python
  - possible to read/adapt them

```
#!/usr/bin/env python

from DIRAC.Core.Base import Script
Script.parseCommandLine(ignoreErrors=False)
from DIRAC.Interfaces.API.Job import Job
from DIRAC.Interfaces.API.Dirac import Dirac

dirac = Dirac()

j = Job()
j.setCPUTime(500)
j.setExecutable('/bin/echo Begin')
j.setExecutable('/bin/date')
j.setExecutable('/bin/hostname')
j.setExecutable('/bin/echo End')
j.setName('test-API')
#j.setNumberOfProcessors(2)

jobID = dirac.submitJob(j)
print("Submission result:", jobID)
if jobID['OK']:
    print("status:", dirac.getJobStatus(jobID['JobID']))
```

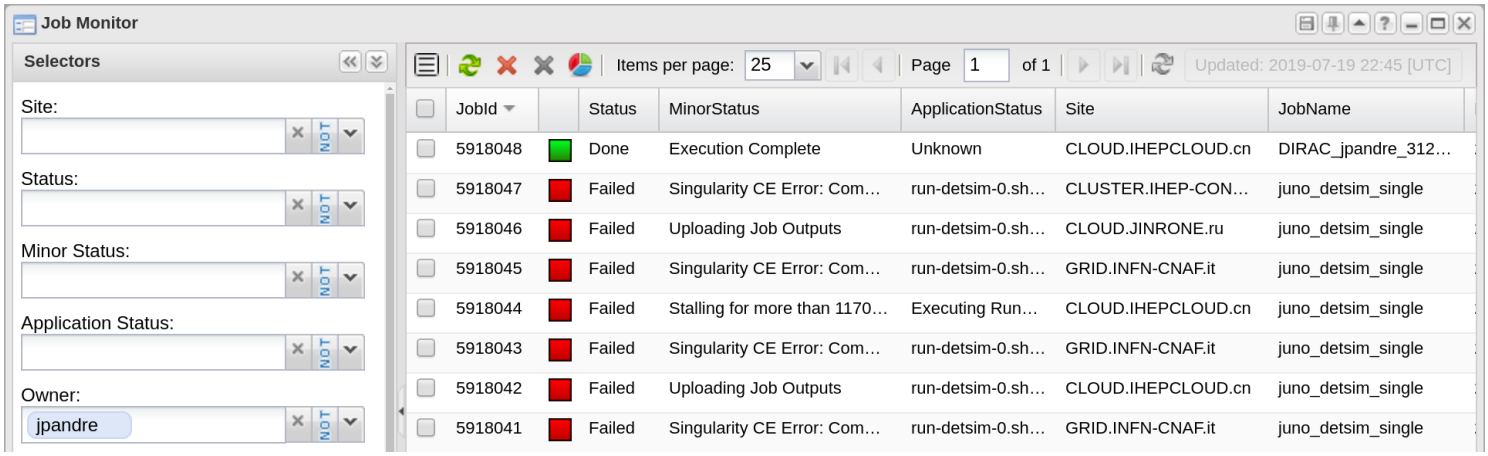
Job submission also possible via website

- Menu → Tools → Job Launchpad

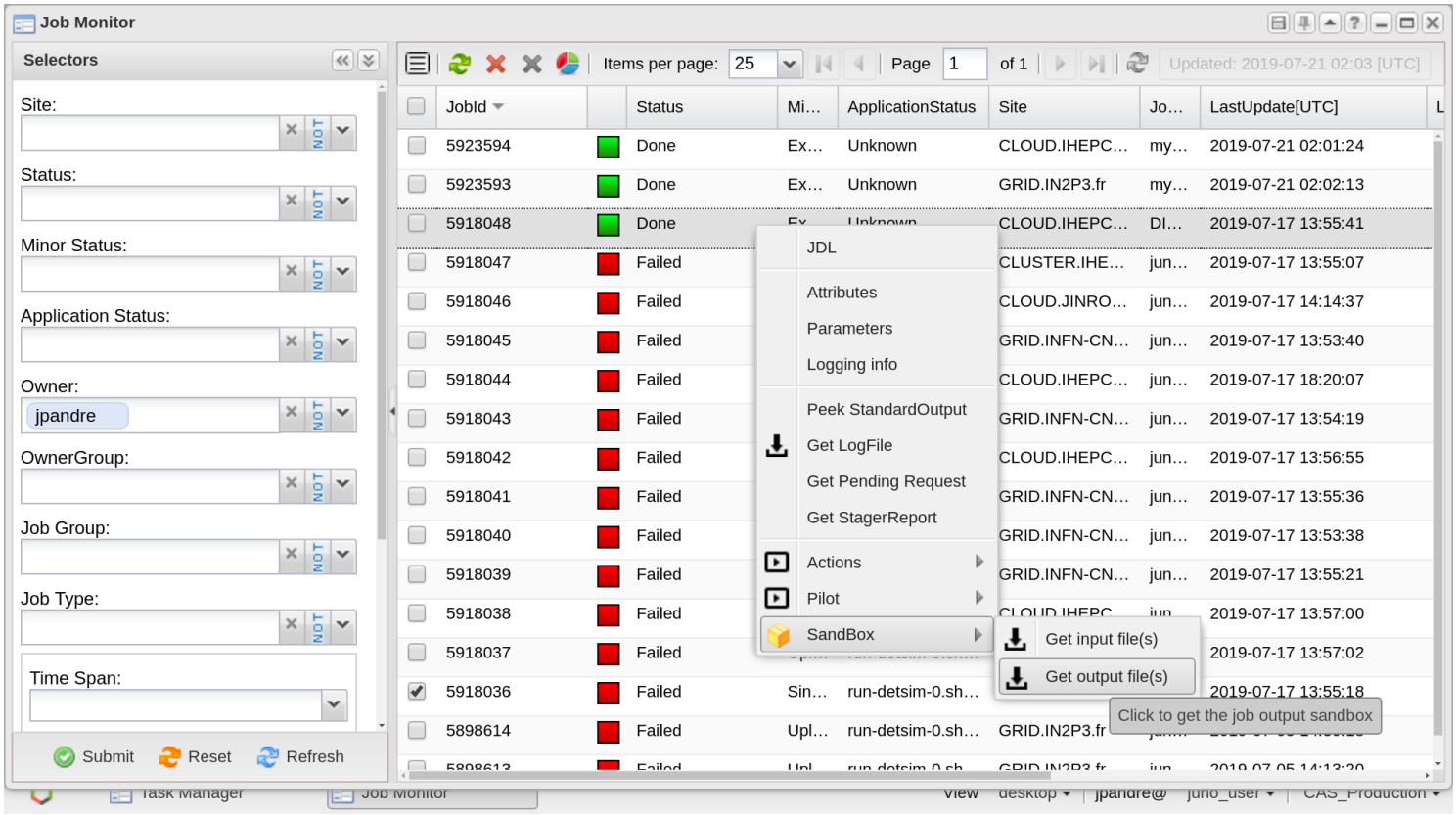


Listing jobs with web interface

- Menu → Applications → Job Monitor

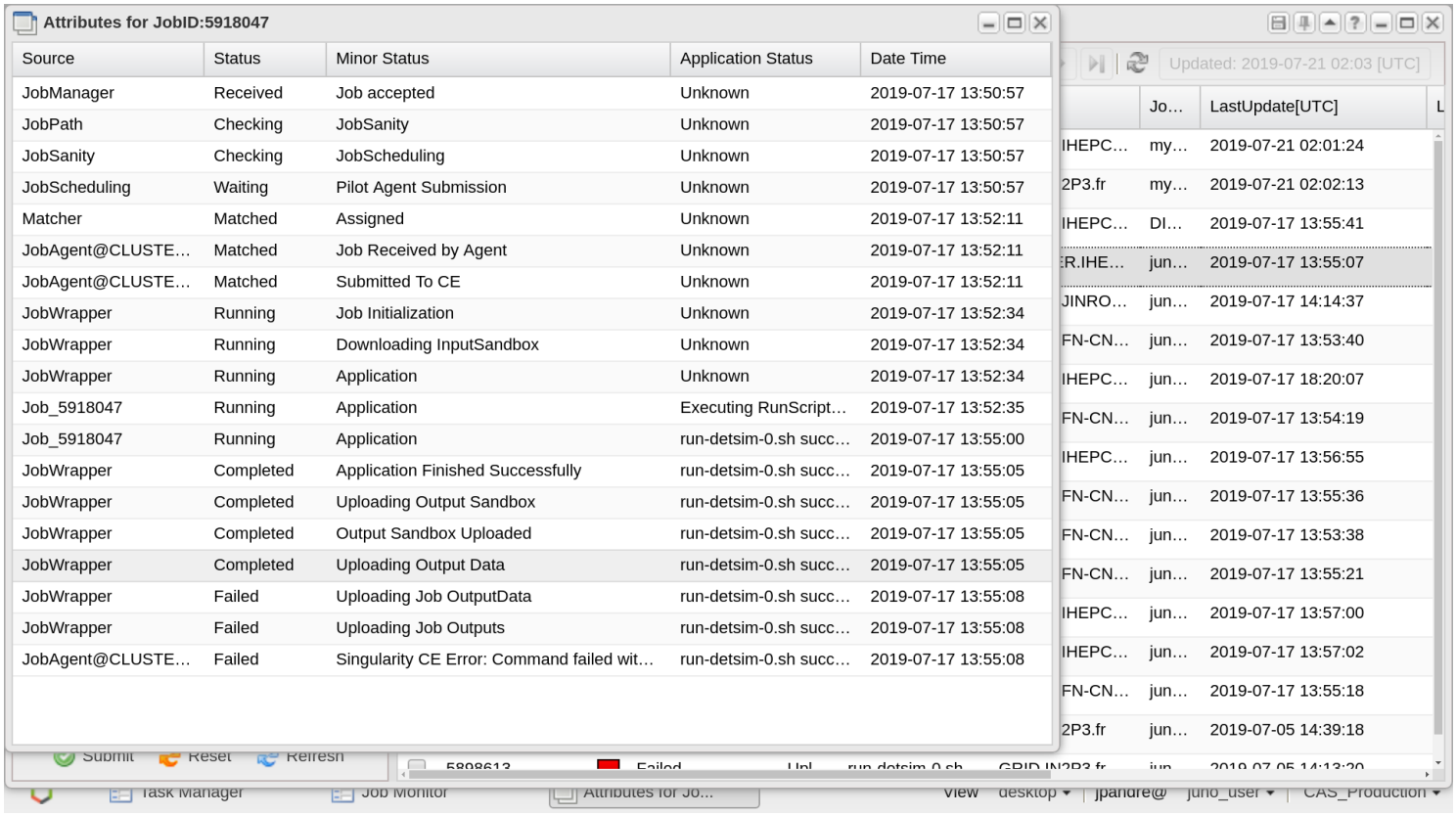


- Right click on job name → Sandbox → Get output files



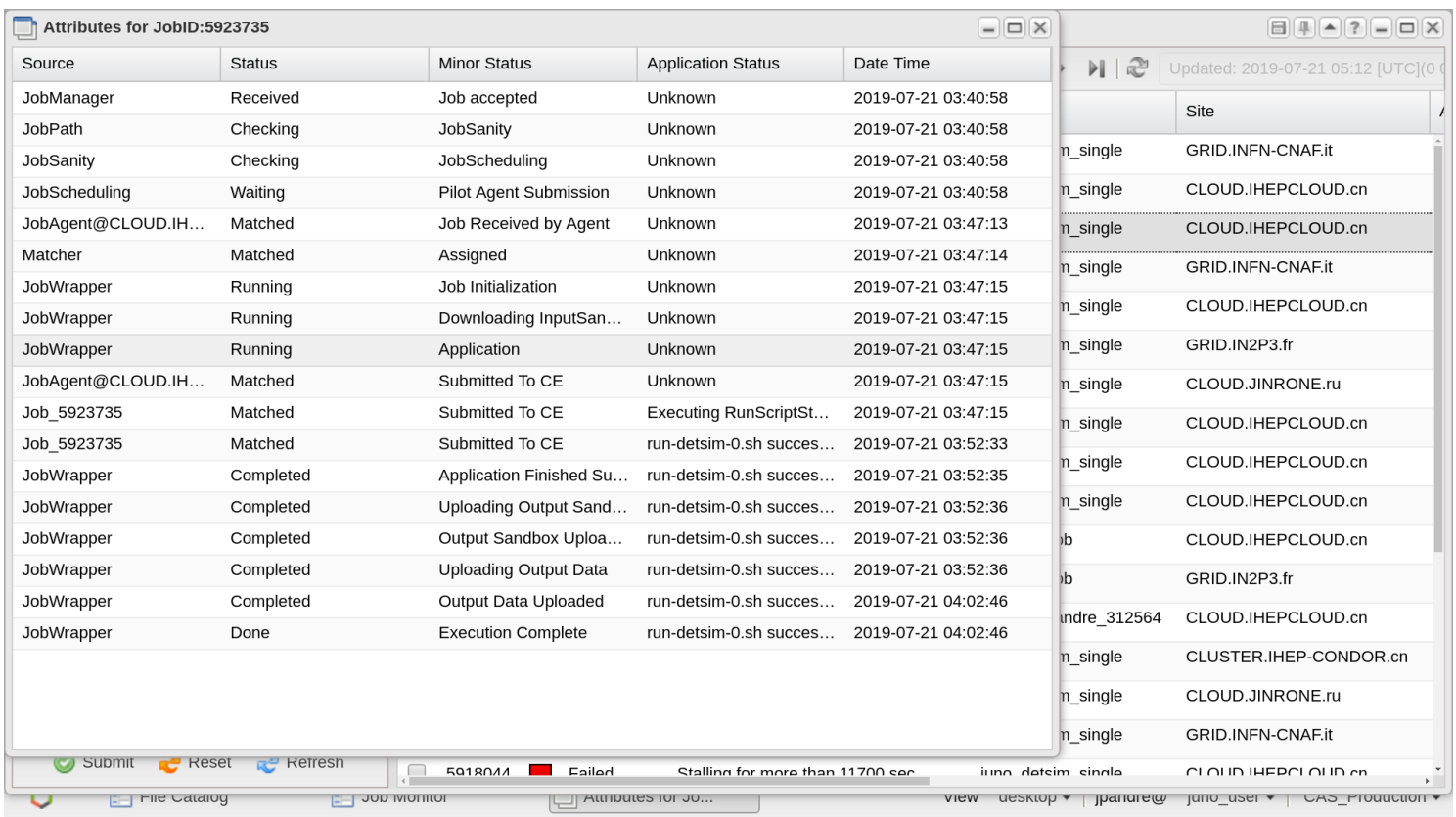
Listing jobs with web interface: logging

- Right click on job name → Logging Info



Attributes for JobID:5918047

Source	Status	Minor Status	Application Status	Date Time
JobManager	Received	Job accepted	Unknown	2019-07-17 13:50:57
JobPath	Checking	JobSanity	Unknown	2019-07-17 13:50:57
JobSanity	Checking	JobScheduling	Unknown	2019-07-17 13:50:57
JobScheduling	Waiting	Pilot Agent Submission	Unknown	2019-07-17 13:50:57
Matcher	Matched	Assigned	Unknown	2019-07-17 13:52:11
JobAgent@CLUSTE…	Matched	Job Received by Agent	Unknown	2019-07-17 13:52:11
JobAgent@CLUSTE…	Matched	Submitted To CE	Unknown	2019-07-17 13:52:11
JobWrapper	Running	Job Initialization	Unknown	2019-07-17 13:52:34
JobWrapper	Running	Downloading InputSandbox	Unknown	2019-07-17 13:52:34
JobWrapper	Running	Application	Unknown	2019-07-17 13:52:34
Job_5918047	Running	Application	Executing RunScript…	2019-07-17 13:52:35
Job_5918047	Running	Application	run-detsim-0.sh succ…	2019-07-17 13:55:00
JobWrapper	Completed	Application Finished Successfully	run-detsim-0.sh succ…	2019-07-17 13:55:05
JobWrapper	Completed	Uploading Output Sandbox	run-detsim-0.sh succ…	2019-07-17 13:55:05
JobWrapper	Completed	Output Sandbox Uploaded	run-detsim-0.sh succ…	2019-07-17 13:55:05
JobWrapper	Completed	Uploading Output Data	run-detsim-0.sh succ…	2019-07-17 13:55:05
JobWrapper	Failed	Uploading Job OutputData	run-detsim-0.sh succ…	2019-07-17 13:55:08
JobWrapper	Failed	Uploading Job Outputs	run-detsim-0.sh succ…	2019-07-17 13:55:08
JobAgent@CLUSTE…	Failed	Singularity CE Error: Command failed wit…	run-detsim-0.sh succ…	2019-07-17 13:55:08



Attributes for JobID:5923735

Source	Status	Minor Status	Application Status	Date Time
JobManager	Received	Job accepted	Unknown	2019-07-21 03:40:58
JobPath	Checking	JobSanity	Unknown	2019-07-21 03:40:58
JobSanity	Checking	JobScheduling	Unknown	2019-07-21 03:40:58
JobScheduling	Waiting	Pilot Agent Submission	Unknown	2019-07-21 03:40:58
JobAgent@CLOUD.IH…	Matched	Job Received by Agent	Unknown	2019-07-21 03:47:13
Matcher	Matched	Assigned	Unknown	2019-07-21 03:47:14
JobWrapper	Running	Job Initialization	Unknown	2019-07-21 03:47:15
JobWrapper	Running	Downloading InputSan…	Unknown	2019-07-21 03:47:15
JobWrapper	Running	Application	Unknown	2019-07-21 03:47:15
JobAgent@CLOUD.IH…	Matched	Submitted To CE	Unknown	2019-07-21 03:47:15
Job_5923735	Matched	Submitted To CE	Executing RunScriptSt…	2019-07-21 03:47:15
Job_5923735	Matched	Submitted To CE	run-detsim-0.sh succes…	2019-07-21 03:52:33
JobWrapper	Completed	Application Finished Su…	run-detsim-0.sh succes…	2019-07-21 03:52:35
JobWrapper	Completed	Uploading Output Sand…	run-detsim-0.sh succes…	2019-07-21 03:52:36
JobWrapper	Completed	Output Sandbox Uploa…	run-detsim-0.sh succes…	2019-07-21 03:52:36
JobWrapper	Completed	Uploading Output Data	run-detsim-0.sh succes…	2019-07-21 03:52:36
JobWrapper	Completed	Output Data Uploaded	run-detsim-0.sh succes…	2019-07-21 04:02:46
JobWrapper	Done	Execution Complete	run-detsim-0.sh succes…	2019-07-21 04:02:46

Exercise: Submitting JUNO jobs

- Now we can start to submit JUNO jobs. . .
- Can you modify the previous JDL & script to submit a short simulation job?
- Remember:
  - Given job might run anywhere you **cannot** use any “local” file paths!
  - CVMFS is your friend
  - Alternative put tar-ball in DFC and download & extract it before using software.
- Add `OutputData = { "*.root" };` to JDL file
  - What happens if you don’t put it?
  - Did you find your ROOT output files?
- JDL files support many more options still. . .
  - This tutorial is **not** meant to cover them all
  - and I don’t know them all either. . .
  - `https://dirac.readthedocs.io/en/latest/UserGuide/GettingStarted/UserJobs/JDLReference/index.html`

Submitting JUNO jobs (solution)

run_detsim.sh

```
#!/bin/bash

export CMTCONFIG=amd64_linux26

source /cvmfs/juno.ihep.ac.cn/centos7_amd64_gcc830/
      Pre-Release/J22.1.0-rc4/setup.sh

python ${TUTORIALROOT}/share/tut_detsim.py gun
```

---

run_detsim.jdl

```
JobName = "run_detsim";
Executable = "/bin/bash";
Arguments = "run_detsim.sh";
StdOutput = "stdout.out";
StdError = "stderr.err";
InputSandbox = { "run_detsim.sh" };
OutputSandbox = { "stdout.out", "stderr.err" };
OutputData = { "*.root" };
VirtualOrganisation = "vo.juno.ihep.ac.cn";
```

```
FC:/juno/user/j/jpandre/11186/11186429>ls
sample_detsim.root
sample_detsim_user.root
```

Exercise: Submitting JUNO jobs 2

- In previous exercise we ran `detsim`
  - No input needed $\Rightarrow$ easier to handle
- Now, let's try to run a job using an input!
  - for example, run `elecsim` on previous output
- Option 1:
  - Add `InputData` field to JDL file
    - ★ path should correspond to DFC path!
- Option 2 (advanced!):
  - Use `xrootd` for local access
  - Need to know where the file is
  - Need to know xrootd path for each cluster

```
root :// junoeos01.ihep.ac.cn:1094//eos/juno/dirac
root :// xfer-archive-03.cr.cnaf.infn.it:1094//
    dirac
root :// ccxrootdegee.in2p3.fr:1094//pnfs/in2p3.fr
    /data/juno/dirac
root :// eos.jinr.ru:1094//eos/juno/dirac
```

  - Add `Site` field to JDL file with location

Submitting JUNO jobs 2 (solution, opt #1)

```
#!/bin/bash

export CMTCONFIG=amd64_linux26

source /cvmfs/juno.ihep.ac.cn/centos7_amd64_gcc830/
      Pre-Release/J21v1r0-Pre0/setup.sh

python ${TUTORIALROOT}/share/tut_det2elec.py
```

```
JobName = "run_elecsim";
Executable = "/bin/bash";
Arguments = "run_elecsim.sh";
StdOutput = "stdout.out";
StdError = "stderr.err";
InputSandbox = { "run_elecsim.sh" };
OutputSandbox = { "stdout.out", "stderr.err" };
InputData = { "/juno/user/j/jpandre
      /10093/10093688/sample_detsim.root" };
OutputData = { "*.root" };
VirtualOrganisation = "vo.juno.ihep.ac.cn";
```

```
FC:/juno/user/j/jpandre/10094/10094023>ls
sample_detsim.root
sample_elecsim.root
sample_elecsim_user.root
```

Submitting JUNO jobs 2 (solution, opt #2)

```
#!/bin/bash

export CMTCONFIG=amd64_linux26

source /cvmfs/juno.ihep.ac.cn/centos7_amd64_gcc830/
    Pre-Release/J21v1r0-Pre0/setup.sh

python ${TUTORIALROOT}/share/tut_det2elec.py --input
    test:root://eos.jinr.ru:1094//eos/juno/dirac/juno/user/j/
    jpandre/10093/10093688/sample_detsim.root --rate
    test:1.0
```

```
JobName = "run_elecsim_opt2";
Executable = "/bin/bash";
Arguments = "run_elecsim_opt2.sh";
StdOutput = "stdout.out";
StdError = "stderr.err";
InputSandbox = { "run_elecsim_opt2.sh" };
OutputSandbox = { "stdout.out", "stderr.err" };
OutputData = { "*.root" };
VirtualOrganisation = "vo.juno.ihep.ac.cn";
Site = "GRID.JINR-CONDOR.ru";
```

```
FC:/juno/user/j/jpandre/10094/10094091>ls
sample_elecsim.root
sample_elecsim_user.root
```

More exercise ideas!

If you've completed previous exercises, try finding out how you'd like to do a few different things & test them out:

- Send job to specific site
- Send output to specific SE
- Submit many similar jobs
  (ie, change only random seed)
- Group multiple jobs
- Put output in specified folder

More exercise ideas!

If you've completed previous exercises, try finding out how you'd like to do a few different things & test them out:

- Send job to specific site
  - Add `Site` to JDL
- Send output to specific SE
  - Add `OutputSE` to JDL
- Submit many similar jobs
  (ie, change only random seed)
  - Use parametric jobs!
  - Add to jdl: `Parameters`, `ParameterStart`, `ParameterStep`, ...
  - Change `Arguments` to include parameter
  - Change script to take in CLI option
- Group multiple jobs
  - Add `JobGroup` to JDL
  - Use `-g` flag in various commands
    (check their help!)
- Put output in specified folder
  - Add `OutputPath` to JDL
  - Note: folder is relative path to your user folder!

Part #2
JSUB

What is JSUB

- Goal: make JUNO DCI (user) job submission easier & more efficient
- Tool developped by Xianghu Zhao and Yifan Yang (postdoc @IHEP)
- Resources from Yifan:
  - DocDB-7303: JSUB tutorial
  - `https://jsubpy.github.io/`
- Examples in CVMFS:
  `/cvmfs/dcomputing.ihep.ac.cn/frontend/jsub/1.2/install/jsub/examples/juno/`
- On the down side original developpers finished their postdocs...
  - For now things work, and we will try mantain it...
  - But current DCI manpower is very limited...
- Let us know if you'd like to help maintain JSUB!

Getting started with JSUB

- JSUB config file (~/.jsubrc):

```
package: [jsub_juno, jsub_dirac]

taskDir:
  location: /path/to/my/jsub/manager/folder
  #location:  /home/jp/jsub

backend:
  default: dirac
```

- Activate JSUB:

```
% source /cvmfs/dcomputing.ihep.ac.cn/frontend/
      jsub/activate.sh –e juno
```

Getting started with JSUB 2

% jsub ––help

```
Usage: jsub [OPTIONS] COMMAND [ARGS]...

Options:
  ––jsubrc TEXT Configuration file to run JSUB with.
  ––help        Show this message and exit.

Commands:
  create      Create a task from a task description file .
  getlog      Retrieve log files of selected subjobs.
  jobvar      View the values of jobvar lists
  ls          List all tasks.
  package     Show active packages.
  register    Upload files to SE and register them to
              DFC.
  remove      Delete a task.
  rename      Rename a task.
  reschedule  Reschedule selected subjobs.
  resubmit    Equivalent to 'jsub submit –r' command
  run         Create from a task profile , and submit.
  show        Show detailed description of a task.
  status      Show the backend status of a task.
  submit      Submit a task to backend.
  version     Show the version of the software.
```

JUNO simulation – job definition file

based on `101_detsim.yaml` example from Yifan

`detsim.yaml`

```
taskName: juno_sim
experiment: juno
#softVersion:  'centos7_amd64_gcc830/Pre-Release
    /J20v1r0-Pre2'
softVersion:
    arch: 'centos7_amd64_gcc830/'
    release: 'J20v1r0-Pre2'

backend:
    type: dirac
    outputSubDir: 'temporary_jsub_tests'

splitter:
    mode: splitByEvent
    evtMaxPerJob: 5
    njobs: 2

workflow:
    steps: [detsim]

    detsim:
        seed: 1
        additionalArgs: 'gun'
```

JUNO simulation – submitting job

2 options for submitting job:

- create job, then submit:
  - Gives the chance to check if that was indeed the job you wanted to submit

```
% jsub create detsim.yaml
Task created successfully
– ID          : 1
– Name        : juno_sim
– Job Number : 2

% jsub submit 1
Submitting task 1
[2022–05–15 22:05:21.442 +0200 CEST][JSUB|INFO
      ]: 2 jobs successfully submitted to backend.
```

- Submit in a single step:

```
% jsub run detsim.yaml
[2022–05–15 22:13:40.400 +0200 CEST][JSUB|INFO
      ]: 2 jobs successfully submitted to backend.
Task submitted successfully
– ID          : 2
– Name        : juno_sim
– Job Number : 2
```

Job management with JSUB

```
% jsub ls
[2022-05-15 22:20:16.003 +0200 CEST][JSUB|INFO]:
    Fetching backend status info update for tasks. May
    take some time.
Task ID  Name      Experiment Backend Status (D|F|R|W|
    O) Creation Time (UTC) Info  Updated (UTC)
-------- --------- ---------- --------
    -------------------- --------------------
    ----------------------
1           juno_sim  juno           dirac      2|0|0|0|0
                2022-05-15 20:03:32 2022-05-15
    20:20:20
2           juno_sim  juno           dirac      2|0|0|0|0
                2022-05-15 20:13:32 2022-05-15
    20:20:23

% jsub getlog 1 -s D
Fetching the log  files  of  task  1
Specifying job  status:  D
[2022-05-15 22:21:25.596 +0200 CEST][JSUB|INFO]:
    Retrieved log files of 2 subjobs

% ls ~/jsub/1/logfiles
0/   1/
% ls ~/jsub/1/logfiles/0/unit/detsim
detsim.log
```

JSUB output ROOT files

```
FC:/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim>ls
detsim_1.root
detsim_2.root
detsim_user_1.root
detsim_user_2.root
```

- Saves file in DFC based on `taskName` and `outputSubDir`
- Easier to know what corresponds to each file
- Better natural organization between `detsim/elecsim/...`
- Note: possible to do that with JDL also, but in JSUB it gets done automatically

Exercise: Submit a detsim+elecsim job

- Modify the previous YAML script to run also `elecsim`
- Bonus: change some configuration from `elecsim`
- How are the output files organized?
- How are the log files organized?

Submit a detsim+elecsim job (solution)

```
taskName: juno_sim_elec
experiment: juno
#softVersion:  'centos7_amd64_gcc830/Pre-Release
    /J20v1r0-Pre2'
softVersion:
    arch: 'centos7_amd64_gcc830/'
    release: 'J20v1r0-Pre2'

backend:
    type: dirac
    outputSubDir: 'temporary_jsub_tests'

splitter:
    mode: splitByEvent
    evtMaxPerJob: 5
    njobs: 2

workflow:
    steps: [detsim, elecsim]

    detsim:
        seed: 1
        additionalArgs: 'gun'
```

Simulating multiple similar configurations

- This is very useful if you want to simulate many similar jobs with varying inputs
- Change splitter from `splitByEvent` to `splitByJobvar` for more flexibility
- `splitByJobvar` also creates variables that can be used when configuring jobs
  - a list of isotopes can be provided
  - some of those informations used to define filename
  - any other variable (like a seed) could be added. . .

Simulating multiple similar configurations

```
# [...]

splitter:
  mode: splitByJobvars
  maxSubJobs: 5
  evtMaxPerJob: 10
  jobvarLists:
    nuclear:
      type: enumerate
       list : [ 'U-238', 'Th-232']
      group: nuclear
    subjob:
      type: range
      group: same_nuclear

workflow:
  steps: [detsim]
  detsim:
    output: '$(nuclear).$(subjob).detsim.root'
    userOutput: '$(nuclear).$(subjob).user.detsim.
        root'
    additionalArgs: 'gendecay␣--nuclear␣$(nuclear
        )␣--volume␣pTarget␣--material␣LS'
```

Exercise: Submit an elecsim job

- Now, if you had to run `elecsim` from the `detsim` generated, how would you do?
  - For simplicity consider the files produced by the first `detsim` with JSUB
- Option 1:
  - Pass an `input:` to `elecsim`
- Option 2 (advanced):
  - Use the file catalog to know which files to use as input!
  - Need to use file metadata to pick files

Submit an elecsim job (solution, opt. #1)

```
taskName: juno_sim
experiment: juno
softVersion: 'centos7_amd64_gcc830/Pre-Release/
    J20v1r0-Pre2'

backend:
    type: dirac
    outputSubDir: 'temporary_jsub_tests'

splitter:
    mode: splitByJobvars
    maxSubjobs: 2
    jobvarLists:
        idx:
            type: range

workflow:
    steps: [elecsim]

    elecsim:
        input: '/juno/user/j/jpandre/
            temporary_jsub_tests/juno_sim/detsim/
            detsim_$(idx).root'
        rate: 1.0
```

Submit an elecsim job (solution, opt. #2.1)

```
FC:/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim>meta set detsim_1.root userdata 0
/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim/detsim_1.root {'userdata': '0'}
FC:/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim>meta set detsim_2.root userdata 0
/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim/detsim_2.root {'userdata': '0'}
FC:/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim>meta set detsim_user_2.root userdata 1
/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim/detsim_user_2.root {'userdata': '1'}
FC:/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim>meta set detsim_user_1.root userdata 1
/juno/user/j/jpandre/temporary_jsub_tests/juno_sim/
      detsim/detsim_user_1.root {'userdata': '1'}
```

Submit an elecsim job (solution, opt. #2.2)

```
taskName: juno_sim
experiment: juno
softVersion: 'centos7_amd64_gcc830/Pre-Release/
    J20v1r0-Pre2'
backend:
  type: dirac
  outputSubDir: 'temporary_jsub_tests'

splitter:
  mode: splitByJobvars
  jobvarLists:
    input_filename:
      type: find_lfns
      path: '/juno/user/j/jpandre/
          temporary_jsub_tests/juno_sim/detsim'
      metaspec: '"userdata=0"'
    seed:
      type: range

workflow:
  steps: [elecsim]
  elecsim:
    input: '$(input_filename)'
    rate: 1
```

More exercise ideas!

If you've completed previous exercises, try finding out how you'd like to do a few different things & test them out:

- Submit job to specific site
- Submit 'gun' with specific positions along z axis
- Provide input file to JSUB
- Run an executable from CVMFS (like Muon.exe)
- Create a user defined task

More exercise ideas!

If you've completed previous exercises, try finding out how you'd like to do a few different things & test them out:

- Submit job to specific site
  - Add `site` to backend
- Submit 'gun' with specific positions along z axis
  - splitByJobvar should allow this configuration
- Provide input file to JSUB
  - Add `input` section before `workflow`
- Run an executable from CVMFS (like Muon.exe)
  - Create a block in `workflow` of type `junosoft`
  - Need to provide `software`, `arguments` and `outputUpload`
- Create a user defined task
  - Create a block in `workflow` of type `user_script`
  - Need to provide `file` to run and `args`
  - Note: could be a bit tricky to get a root output from this step though...

One final example...

```
# [...]

input:
  run-elecsim-1_jsub.sh: run-elecsim-1_jsub.sh
  fileLists.tgz: fileLists.tgz
  tut_det2elec_hx.py: tut_det2elec_hx.py

workflow:
  steps: [elecsim, my_elecsim]

  # JP: I needed to include elecsim so it adds the
      output rootfiles to the list of files to be
      uploaded
  elecsim:
    fullArgs: '--help'
    input: '/juno/user/j/jpandre/jsub_test_2022/
        rad_v2/detsim/K-40.1.detsim.root'
    output: 'elecsim'
    user_output: ''

  my_elecsim:
    type: user_script
    file: 'run-elecsim-1_jsub.sh'
    args: ''
```

Part #3
Production Tool

Introduction

- As I mentioned before, this requires your VOMS to have `production` role

```
% dirac-proxy-init -U -g juno_production -M
 [...]
Added VOMS attribute /juno/Role=production
 [...]
DIRAC group : juno_production
 [...]
VOMS fqan   :   ['/juno/Role=production', '/juno']
 [...]
```

- It is meant for large scale production
  - less flexible than JSUB or JDL
  - however tuned to work with large datasets
- Getting started:

```
% ihepdirac-juno-make-productions --example
      > production.ini
```

- Submitting jobs:

```
% ihepdirac-juno-make-productions --ini
      myprod.ini --dryrun

% ihepdirac-juno-make-productions --ini
      myprod.ini
```

- Note: I've never used the production system...

Production example

I just stripped the comments of the example file by Xiaomei

```
[ all ]
process = Chain
softwareVersion = centos7_amd64_gcc830/Pre-Release/
      J21v2r0-Pre0
prodName = JUNOProdTest
transGroup = JUNO_prod_test
outputType = user
outputSubDir = positron/prd_2021
outputMode = closest
moveTargetSE = IHEP-STORM CNAF-STORM

[Chain]
seed = 42
evtmax = 2
njobs = 10
max2dir = 10000
tagParser = (.*)_(.*)MeV
tags = e+_0.0MeV e+_1.398MeV e+_4.460MeV
workflow = detsim elecsim_rec
moveType = detsim
userOutput = 0
detsim-mode = gun --particles {0} --momentums {1} --
      positions 0 0 0
elecsim_rec-mode = --rate 0.001 --enableWP --
      enableWPDarkPulse --no-evtrec
```

The end

Thank you for your attention!