

# Job Management Utilities

W. Roethel  
UC Irvine

# Validating Jobs

## Problem:

Using the extended collection syntax BbkDatasetTcl does not necessarily report the correct number of events for a given job.

→ How to validate a framework job?

**FjrCheckJob** (FrameJobReport) solves this issue.

- Requires Job Report file.
- Checks for consistency of the event counters. These event counters are not available in the log file.
- Checks for successful shutdown of the Framework (by definition, last lines in the job report file).

Run FjrCheckJobs on the job report file:

```
roethel@noric04> FjrCheckJob jobReport_1.trd  
Job checked Ok!
```

Get the number of processed events:

```
roethel@noric04> FjrCheckJob -p jobReport_1.trd  
Job checked Ok!  
Events processed: 50
```

Some failures:

```
roethel@noric04> FjrCheckJob jobReport_bad1.trd  
Job failed! - Bbr::JobReport.stopTime not found!
```

```
roethel@noric04> FjrCheckJob jobReport_bad2.trd  
Job failed! - Event count mismatch!
```

# Simple Job Manager (SJM)

<http://www.slac.stanford.edu/BFROOT/www/Computing/Distributed/Bookkeeping/SJM/SJMMain.htm>

Utility to manage processing of a framework application running over collections in a dataset.

## Does:

- Process collections available in datasets.
- Automates most processing. In the extreme case, start the job monitoring script and get an email when all is done.
- Simple to set up.
- 'Supports' any kind of output created.

Does not:

- No support for merging output collections (if any).
- No support for data transfers or inserts of collections to bbkr14/bbkr18.
- No fancy stuff. Does not do well treating special cases.

**Real Strength is its Simplicity!**

# Getting Started

Starting Point: Working framework Application + tcl file.

## 1. Create a tcl snippet template:

- Define job specific parameters with FwkCfg vars.
- Source the main tcl file on the last line of the snippet.
- *SJM*: Use available tags to define job specific values.

Available tags are:

- `<ID>`, `<ID100>` - The (unique) job id.
- `<INPUTTCL>` - Name of the tcl file defining the input data.
- `<JOBREPORT>` - Name of the job report file.
- `<RUNDIR>` - Run directory as defined in the job config.
- `<NAME>` - Name of the *SJM* project.

# Example tcl Snippet Template

## TestConfig.tcl

```
set ProdTclOnly true
sourceFoundFile <INPUTTCL>
set jobReportName <JOBREPORT>
set rootName /afs/slac.stanford.edu/.../root/ntup_<ID>.root
sourceFoundFile /afs/slac.stanford.edu/u/br/roethel/\
/.../Chi2To4pi.tcl
```

## Chi2To4pi.tcl:

```
FwkCfgVar histFileName $rootName
#..Number of Events defaults to 0 (run on full tcl file)
FwkCfgVar NEvents 0
# set the job report filename
jobReport filename $jobReportName
```

## 2. 'Fill out' the config file parameters:

### TestConfig.cfg

```
# File to configure a Simple Job Manager
# define the name of the SJM
SJMName = TestProject

# name of the input dataset
DatasetName = SP-1235-TwoPhotonPentaquark-Run3-R16a

# max. number of events per tcl file (as used by the --tcl option in
#                                     BbkDatasetTcl)
MaxEvents = 50000

# raw command options to be passed to BbkDatasetTcl
# WARNING: SJM adds the --tcl --basename and --splitruns options. Make sure
#          that adding options does not interfere with these defaults
BbkDatasetTclRaw =

# Template file for the creation of the Tcl Snippet files used
#          for job configuratin
TclSnippet = TestConfig.tcl
```

```
# Define the run directory - this is the directory where the logfile
# and jobreport file will be written. The tag <ID> will be replaced by the
# job specific job id. Please make sure that the run directory does not
# overwrite already existing data.
RunDirectory = /afs/slac.stanford.edu/g/babar/work/r/roethel/analysis/
TestConfig/log/<ID>

# The name of the executable that should be run
Executable = BtaTupleApp

# This is the batch command that will be used to submit jobs to the
# batch queue. The tags <LOG> and <TCL> will be replaced with the job
# specific logfile name and tcl snippet name
BatchCommand = bsub -q kanga -C 0 -o <LOG> <WRAPPER>

# umask = 2
Share = 1

# some optional commands - useful for running on other sites but slac

# The following string at the end of the log file signals that the job
# has completed in the batch queue
# JobFinishedString = Resource usage summary

# This string indicates that the job was executed successfull, i.e. exited
# with an exit code 0
# JobSuccessfulString = Successfully completed
```

### 3. run sjm prepare :

```
roethel@noric04> sjm prepare TestConfig.cfg
Running BbkDatasetTcl --tcl 50000 --basename TestProject --splitruns
SP-1235-TwoPhotonPentaquark-Run3-R16a ...
BbkDatasetTcl: wrote TestProject-1.tcl (1 collections, 50000 events)
BbkDatasetTcl: wrote TestProject-2.tcl (2 collections, 50000 events)
...
Selected 26 collections, 5380143/86492000 events, ~0.0/pb
done. Creating tcl snippets in directory 'prepared' now...
done!
```

### 4. Ready to go

```
roethel@noric04> sjm show TestProject
...updating job status
```

name	prepared	submitted	done	failed	ok
-----	-----	-----	-----	-----	-----
TestProject	108	0	0	0	0

# Behind the Scenes...

Project Directory:

```
TestProject/
```

```
    TestProject.cfg
```

```
    TestProject.tcl
```

```
    tcl/
```

```
    prepared/
```

```
    submitted/
```

```
    done/
```

```
    ok/
```

```
    failed/
```

} job states

Jobs are managed moving their tcl snippet files from one job state directory to another.

# Submit Jobs etc.

- submit jobs: `sjm submit -n 2 TestProject`
- check jobs: `sjm check TestProject`

Job validation is the same as used in `FjrCheckJobs`.

## Automated job management:

SJM provides a job manager daemon that submits and checks jobs:

- `sjm sprite` : Start/stop the daemon.
- `sjm sprited` : The daemon process itself.

Daemon is configured in the SJM configuration file.

# Outlook

- This is just a short overview. For more details check the web site.
- SJM is fairly mature and stable.
  - been around for > 1 year.
  - been used by a couple of people (that I know of).
- Runs anywhere (so far...)
- SJM is simple and small enough (still) to be customizable if wished. E.g. it could be mutated into a utility for managing merge jobs if Dataset were replaced with a list of files to be merged.
- Up for grabs. Anyone wishing to take over development and/or support - be my guest 😊