Job Submission to McFarm – Revised Interim Procedures
April 2003

McFarm operates on both *official* requests (Monte Carlo Production that is done to satisfy another user), and *internal* requests (processing done for local consumption, though it may still be stored in SAM). This document describes the recommended protocols for handling an official request. It is expected that more robust and automated methods will be available in the near future.

In order to run the request scripts, your system must have installed:

- SAM
- sam_admin v4_3_4
- globus on both the machine you are using to run these scripts and on the *target farm* where the request is to be sent (may be your own farm).
- User mcfarm must be in the map file on the target machines.

Also, you create a configuration file that specifies your normal operating disposition for various otuput types. See the end of this document. The file is $FARM_ABSE/conf_files/request.conf.

Processing steps:

1. If your mcfarm login procedures do not already do so, do the following:

   - setup sam
   - setup sam_admin v4_3_4
   - unsetup ups (necessary to avoid conflicting perl versions)
   - . $FARM_BASE/globus_setup (note the leading period)
   - grid-proxy-init -valid hh:mm (say 96:00 -- four days – if necessary)

2. Obtain the ID of the next official request in priority sequence from SAM

   - cd $FARM_BASE
   - python python/Queue.py

3. Check to be sure the target farm can **do** this request:

   - Visit the web site **www-d0.fnal.gov/computing/mcprod/mcc.html** / **Current Requests** and click on the request ID to see it.
   - Examine the request to be certain that **if** it contains any lines with "**D0Release pNN.NN.NN**" (on a "cfg" line) that the target farm **has** that release installed. Do not be confused by other appearances of a release number, in comments or other descriptions. Load the specified release if necessary before proceeding, then repeat step 1.
   - Examine the request to be certain that you have the necessary **cardfiles** installed. The version needed must be found in your /d0dist/dist/packages/cardfiles directory. Load the cardfiles from ups if necessary before proceeding (repeat step 1). Use these commands as root:

     source /fnal/ups/etc/setups.sh
     setup upd
     upd install -h www-d0.fnal.gov cardfiles vNN-NN-NN

4. Reserve the request for your farm, and get its parameters.

- Note: if you took a lengthy amount of time to get through step 3, you should **repeat step 2** to be sure the request is still the next current one.
- get_request requestID

The above command will "reserve" it for you, and obtain a copy of the request details (into a "work" subdirectory). Normally, this request would **not** already be "running", but it is possible that the request is already flagged as running (by somebody). If you know for sure that it is you who has previously reserved this request, then you will have to add a "—ignore_status" argument to the above command in order to get past the error.

5. Parallelize the job and dispatch it to a target farm (which may be your own farm).

- launch_request requestID target_farm_base events job-type <input_file>

where: **requestID** is the ID just reserved (e.g., 1234)

target_farm_base is the URL and directory name of the target farm. If this is your farm, then this argument is simply $FARM_BASE without a node-name (and globus is **not** used).
If this is a different farm, where globus and McFarm have been installed, then this argument is URL: farm_base, for example:
hepfm007.uta.edu:/home/mcfarm

e**vents** is the number of events to be run. NOTE: Currently, both d0gstar and d0reco fail, about 10% of 100-event jobs do not complete. You will probably want to increase the number of events that you start with to allow for this (in groups of 1000). Alternatively, you can assess the results after the request is done, and make more if necessary (repeat step 5).

job-type may be one of these values. Normally, you use PDSRT for regular jobs and PD for minbias creation.

PDSRT – parallelize the request by making simple chains that begin with pythia and end with d0reco (reco and thumbnail output). Each parallel segment of the request, a job, will be independent of other segments (jobs).

PD – parallelize the request by making simple chains that begin with pythia and end with d0gstar. The gen file is declared to SAM and the d0gstar file is cached. Each parallel segment of the request, a job, will be independent of other segments (jobs).

DSRT – parallelize the request by making **one** large pythia file, then making several DSRT jobs that each processes a portion of that pythia file.

D – parallelize this request by making **one** large pythia file, and then make several d0gstar-only jobs that each process a portion of that pythia file. You use this job-type when making **minbias** data files.

SRT – make a single sim/reco job using a d0gstar file (already created) for input. The input_file must be supplied (e.g., d0g-xxx), and the "events" must be the total number of events in that file. Use this job type when you are re-running sim/reco on a known d0gstar file.
If that file is **not** present in your cache, but is present in SAM, then the input filename must start with "SAM:d0g", and it will be acquired from SAM (and you must start the acquire daemon).

Assuming there are no error messages, the request has been dispatched to the target farm, and will run as cpu's become available there.

6. When you have determined that all the jobs for this request have been finished and stored in SAM, do the following to "close" the request (flag it as "finished").

Note: The best way to know that this request has been finished is to “grep ReqNNNN gather.conf”. That tells you whether or not there are any jobs left for that request. But, you must also check for any staged output files that are not yet stored in SAM. Do this using “ls $FARM_MERGE_STAGE_DIR | grep ReqNNNN”. If you find nothing in either output, then the request is done.

- close_request requestID

Sample **conf_files/request.conf** file

```
################################################
## CONFIGURATION FILE TO SET PARAMETERS
##      TO SUBMIT MCFARM JOBS TO REMOTE CLUSTERS
################################################

## FARM NAME - This is the name of the jobserver on
##             your farm. It should also be the gatekeeer.
FARM_NAME=hepfm007.uta.edu

##  JOB CREATION DIR - This is the directory under which sub-directories
##                     will be created for each request - i.e. under this
##                     directory, the req_NNNN dir will be created, and the
##                     Request_NNNN.py will be transferred here to be operated
##                     upon subsequently.
REMOTE_JOB_CREATION_DIR=/home/mcfarm/job_submit

## xxx_DISPOSITION - Specify the Gather Dispositions for the various stages of the DSRT
##                   jobs

GENERATE_DIS=metadata

D0GSTAR_DIS=sam

D0SIM_DIS=metadata

D0RECO_DIS=sam

TMB_DIS=merge

## DELETE_INPUT - Specify whether or not to delete input
DELETE_INPUT=yes

## NUMBER OF EVENTS PER JOB
NUM_EVENTS_PER_JOB=500
```