# Generation and Upload of Datacards for Running on the GRID

Ryan Bayes

March 10, 2015

## 1 Introduction

Further analyses of data or simulation may require specific settings in MAUS. There is provision in the configuration data base to store data cards to accommodate such analyses. The requester must produce settings to communicate the analysis conditions to the scripts that run the grid job, be it data analysis or simulation. This guide will describe how to make that inquiry and how the information therein is stored and used.

## 2 How to Make a Data Card Request

Requests for simulations or data runs should be posted to the issue tracker in the analysis subsection of the MICE MINE wiki as a stand-alone issue. The request should contain the purpose of the run, the required version of MAUS, the particular range of runs (for data) or geometry specifications (for pure MC), and the contents of the data cards. The analysis coordinator and the acting MC coordinator should be set as watchers for the issue. The purpose of this is to generate a record of requests and to allow the analysis group as a whole to make suggestions as needed. The contents of the request will be validated by the coordinators before being assigned a batch iteration (data) or MC iteration number. It is then submitted to the grid.

### 2.1 Format for the Data Cards

The data cards will be stored as a string to the configuration database (CDB) with the MAUS version number. An example of the commands that may be used in the data cards and their format may be found in the file `\$\{MAUS\_ROOT\_DIR\}/src/common\_py/ConfigurationDefaults.py`. It is highly suggested that the data cards only contain the settings altered with respect the to configuration defaults for the purpose of the run submission. The resulting data cards should be posted as an attachment to the request issue.

Note that a subset of data cards may not be altered as they are given values by the grid scripts.

```
simulation_geometry_filename
output_root_file_name
verbose_level
will_do_stack_trace
configuration_file
reconstruction_geometry_filename
geometry_download_run_number
geometry_download_directory
```

The majority of these parameters are redundant or dictated by the run number. The simulation does require the a value for the `geometry_simulation_id`. This information can be entered into the `execute\_MC.py` as a command line argument.

### 2.2 Quality Checks for the Request

Requests should be tested prior to submission to the issue tracker. This involves running one or two incidences of the requested analysis on a local machine prior using the appropriate execution command (i.e. `${MAUS_ROOT_DIR}/bin/utilities/execute_against_data.py` or `$\{MAUS_ROOT_DIR}/bin/utilities/execute_MC.py`) with the intended arguments. If possible example figures of merit and log files should be posted to the issue tracker.

## 3 Uploading to the Configuration Database

The data cards will be uploaded to the configuration database by the analysis coordinators once the request has been approved. The configuration database stores the data cards as part of the batchIteration and MCiteration classes. Uploads are achieved with the use of a python API that is available in the MAUS software package. The required commands are listed below.

### 3.1 Upload data batch iterations

Uploads to the batch iteration data base can be conducted with a few lines of python code. The required interface algorithms are contained in the `batchiteration_supermouse` subset of the `cdb` package included in `MAUS`. To upload to the database, the user must be inside of MICENet (i.e. on a computer in the MCLR). The code required follows the following pattern.

```
from cdb._batchinteration_supermouse import BatchIterationSuperMouse

obj = BatchIterationSuperMouse()

iterationNumber = N
comment = "Some information motivating the reconstruction here."
recocards = """
         reconstruction specific data card information here
         """
mccards = """
         simulation specific data card information here
         """

res = obj.set_datacards(iterationNumber, comment, recocards, mccards)
```

### 3.2 Upload MC batch iterations

Data cards can be added to the MC iteration data base through use of a similar python API to that of the data batch iterations.

```
from cdb._mcserialnumber_supermouse import MCSerialNumberSuperMouse

obj = MCSerialNumberSuperMouse()

iterationNumber = N
comment = "Some information motivating the simulation here."
mccards = """
         simulation specific data card information here
         """
softwareVersion = "MAUS version here"
res = obj.set_datacards(iterationNumber, mccards, softwareVersion, comment)
```

As with the reconstruction batch submission this must be conducted from inside MICEnet. Only approved data cards may be submitted by authorized users.

## 4 Conclusion

Batch analysis and simulation requests will soon be an important and routine part of the MICE experiment. A transparent process is required to ensure that the batch submissions are done correctly and to make the most of available resources (data storage, flops, time, etc.). It is hoped that, by making requests visible to the entire collaboration while still requiring that the final request be submitted through an analysis coordinator, that mistakes will be reduced through a combination of communal oversight and operational focus.