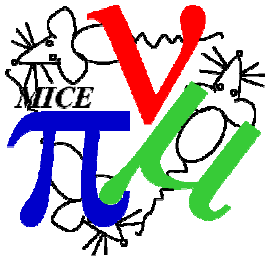
# The Online Buffer

J.J. Nebrensky, (Brunel University, Uxbridge UB8 3PH, UK),




MICE-NOTE-COMP-264

This is a discussion document regarding the proposed use of the Online Buffer.

The Online Buffer is used to store locally the RAW data files created by the Event Builder, before they are uploaded to Castor by the data mover (figure 3). The files may also be used by the online monitoring and reconstruction activity.

At the Trigger-DAQ-Controls Review, the reviewers warned that this three-way activity might saturate the disks, and also that the file uploads to the Grid could conflict with the writing of DAQ data. It was proposed to ameliorate this by splitting the buffer into a set of independent volumes into which the DAQ data would be written on a round-robin basis; outgoing files would meanwhile be read only from one of the other volumes. Further, files being uploaded to the Grid would be staged on the transfer box' system disk, as the (local) staging process is expected to be more deterministic and easier to control than transfers across the WAN.

## 1 Online Buffer

The online buffer consists of a pair of Linux PCs (MICERAID01 and MICERAID02, in the MICE Rack Room) each equipped with a Promise Technology SuperTrak EX8350 RAID controller and a pair of Promise Technology SuperSwap 4600 hot-swap enclosures, each containing 4 SATA hard drives (WD5000ABYS, 500 GB).

Currently all 8 disks are combined into a single RAID 6 volume of about 2 TB on each host. It may be useful to split these into two volumes, so as to separate reads and writes even if a host fails.

## 2 Data Mover

T2K have agreed to allow us to modify their archiver ("QOS") for MICE:

http://www-pnp.physics.ox.ac.uk/~west/t2k/discussions/daq_archive_docs/Archiver_User_Manual.html

The work has been started by a summer student at RAL, and is to be continued by Dr. Janusz Martyniak at Imperial College (GridPP funding).

The data mover should upload each RAW file to the Tier 1 using Grid interfaces, so that each file is registered in the LFC and is created with a SURL with a corresponding path and name. The data mover will need to ensure that the file is genuinely on tape before allowing the original to be deleted, and it should also monitor Castor status to prevent data transfers during downtimes. There should be a record, either log file or in a database, including at least the local filename, LFC filename, SURL and checksum (preferably Adler32, else CRC32) value.

The process will run on a Linux PC (MICEACQ05) also located in the MICE Rack Room. It has a single SATA hard drive to provide a staging area where files from the buffer can be held for transfer to the Grid, in order to decouple the WAN data transfers from the DAQ writes to the buffer.

At present, for a data file named `file`, the Data Mover assumes that it is still being written until a semaphore (zero-length file) `file.complete` is created. It will then create a second semaphore, `file.archiving`, and upload the file. When the transfer is complete it will remove both semaphores (and ultimately the original file).

# 3 File Access Foxtrot

Data from the individual sub-detectors is combined into a single datastream by the Event Builder (is it pushed in or pulled?). The Event Builder writes out the data as a series of files to its local RAID array. The Online Monitoring and Reconstruction will then either access the datastream by connecting to the Event Builder via a network interface, or else by reading the files from the buffer once they become available (fig. 1).

## *3.1 Plan A*

The staging directory on the transfer box is NFS mounted by all the Buffer nodes (fig. 1). If the Event Builder process knows when it is writing data to its local storage, then consequently it knows that if it is not writing out a file then the local storage is available for read access, and the Builder can stage the file (e.g. using `cp`) to the transfer box and create the semaphore.

This approach has the advantage that access to the Buffer is controlled directly by a process that knows what's going on. The downside is that the Data Mover cannot easily remove the original files from the Buffer once they are on tape, as it doesn't have direct access to the Buffer storage – it would be necessary to define more semaphores and run a cleanup daemon on the Buffer nodes.

## *3.2 Plan B*

The storage volumes on the Buffer nodes are NFS mounted by the transfer box. In this case the Event Builder will need to set a semaphore such as `DAQ.writing` in the root directory of the volume whilst writing out a file (as well as setting the relevant per-file semaphore), so that the Data Mover knows which volumes it is safe for it to stage in data from (e.g. using `cp`). The Data Mover will then be able to easily delete the original files

and associated semaphores directly from the Online Buffer, but it will still be necessary to agree and implement a separate set of semaphores to indicate any files on the Buffer that are still needed by the Online Reconstruction and so should not yet be deleted.

### 3.3 Plan C

I've assumed that the staging is done by copying to/from an NFS-mounted volume. SSH (`scp`) doesn't look attractive because of the encryption overhead. Since we have machines with pre-determined IP addresses on the same LAN, it should be possible to adequately secure rsh/`rcp` instead, though it won't be as straightforward checking for new files and semaphores in that case.

Also there is the possibility of borrowing a lightly used 16 bay SCSI-to-SATA enclosure with 500 GB drives from Brunel University, which would provide another 6 TB of space but I'm not sure how best to use it and it would be a single point of failure. Exchanging the old drives for new 2 TB units would give provide enough storage to hold **all** the expected MICE data in 4U of rack space. One possibility would be to run a Grid SE in the MLCR, and write data directly into it from the Event Builders using a "local" protocol such as RFIO.

## 4 Integrity

It would be wise to checksum files as soon as possible. Does DATE have the capability of calculating the checksum of a file as it writes it?

If the checksum is to be calculated by a discrete process, note that doing so separately from the copy will require the data to be read from the buffer *twice*, increasing the chances of a clash. It would be better to combine the checksum and copy into one operation, so e.g. for plan B the `cp` command would actually be something more like

```
cat /mnt/buffer/file | tee /local/file | md5sum > /local/file.md5
```

(and in the case of plan B the CPU load is also on the transfer box, not the Buffer).

## 5 Merging

The Castor tape system at RAL would prefer files to be about 1 GB in size. If the DAQ ends up producing many small files, could these be merged into larger files by the Data Mover? (Would the process require more than simple concatenation?)

## 6 Online Reconstruction

It would be nice to make the histograms from the Online Reconstruction available off-site. It should be possible for the Data Mover to upload them to the Grid where they could be accessed by standard clients or a web browser. They could be written to a dedicated space on the online buffer (obeying the relevant semaphores), but it would be easier to simply NFS-mount on the Transfer Box a suitable space from the Online Reconstruction farm. The Online systems would need to create any semaphores needed by the Data Mover.

# Conclusions

Discussion occurred at the MICE Online Meeting on 10th September 2009 and after:

- Plan B is the way to go, at least in the short/medium term.
- The DAQ experts will make ensure that the Event Builder does the right thing creating/removing semaphores.
- The Event Builder(s) will produce a series of ~100 MB files per run. The Data Mover will aggregate these into a single tarball per run. The DAQ system will limit the number of events per run such that any tarball will be less than 4 GB in size.
- We'll start off with a single Event Builder and RAID array, and see what happens…
- The files will be named `X.Y` where `X` is the run number and `Y` starts at `000` and then increments to `001`, `002`, etc. The top level of organisation will be one directory per MICE step with a new subdirectory created every hundredth run; thus data from run 987 will be stored in `/RAID/mice/MICE/StepS/00900/00987.0nn`.
- Online Reconstruction and Online Monitoring will both read the data directly from the Event Builder via a network socket, rather than from disk. Histograms from both are to be made public.
- The Online Monitoring and Online Reconstruction output should be arranged in a hierarchy corresponding to the global namespace used for the data. The proposal is that the files themselves should be named e.g. `OnMon.00587.root` and `OnRec.00587.root` for run 587, etc. Currently the Monitoring histograms should be in `/RAID/mice/OnlineMonitoring/MICE/Step1/00900/OnMon.00987.root` from where they will be uploaded to Grid storage externally visible via web browser.
- The Transfer Box will NFS-mount relevant areas on the reconstruction and monitoring systems, and the Data Mover will transfer the histograms (ROOT files) to the DPM SE at Brunel University where they will be accessible via Grid clients or web browser. The files are presumed to be small enough that this is not a significant load on the Transfer Box and that we can dispense with the top level semaphores for disk access. This implies correspondence of uid and gid for the DAQ and Data Mover users on the Buffer, Transfer Box and other systems.
- Plan C, a local SE written to directly by RFIO, is a nice idea but will need to be left for later.

## Acknowledgements

I've just slapped this document together – ideas and systems have been provided by many MOGuls.

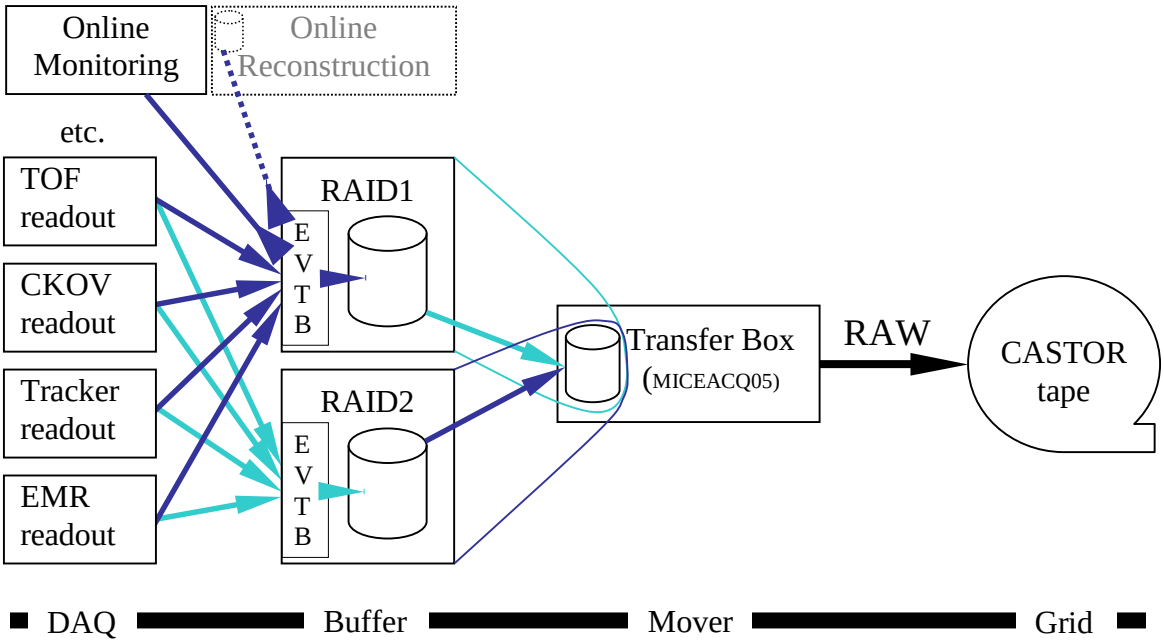



**Figure 1: Operation of the online buffer (plan A).**
**Colours represent synchronous events; conventional arrows are a "push" of data. The staging area of the transfer box is NFS mounted by each storage array. While one is writing data to disk, the other knows it's free and can therefore stage the previous output file. After the end of a file or run, the DAQ will send data to the other array, and the first can stage out.**

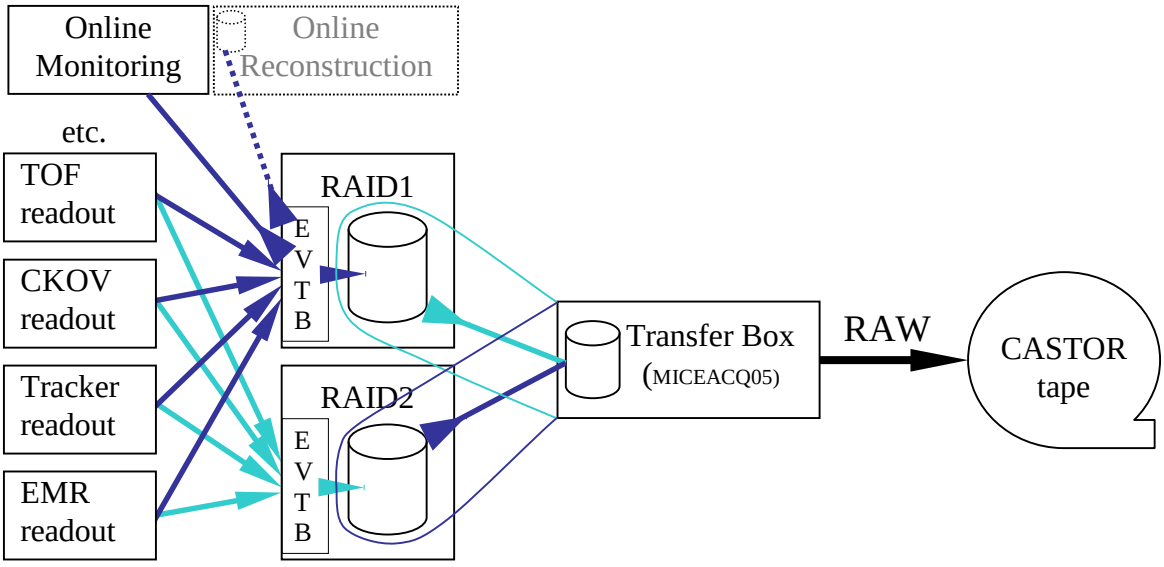



**Figure 2: Operation of the online buffer (plan B).**
**Colours represent synchronous events; conventional arrows are a "push" of data. Each storage array is NFS mounted by the transfer box. While one is writing data to disk, it creates a semaphore in the root directory of the volume. The transfer box stages in files from the volume with no semaphore. After the end of a file or run, the DAQ will send data to the other array, and the first can be staged out.**

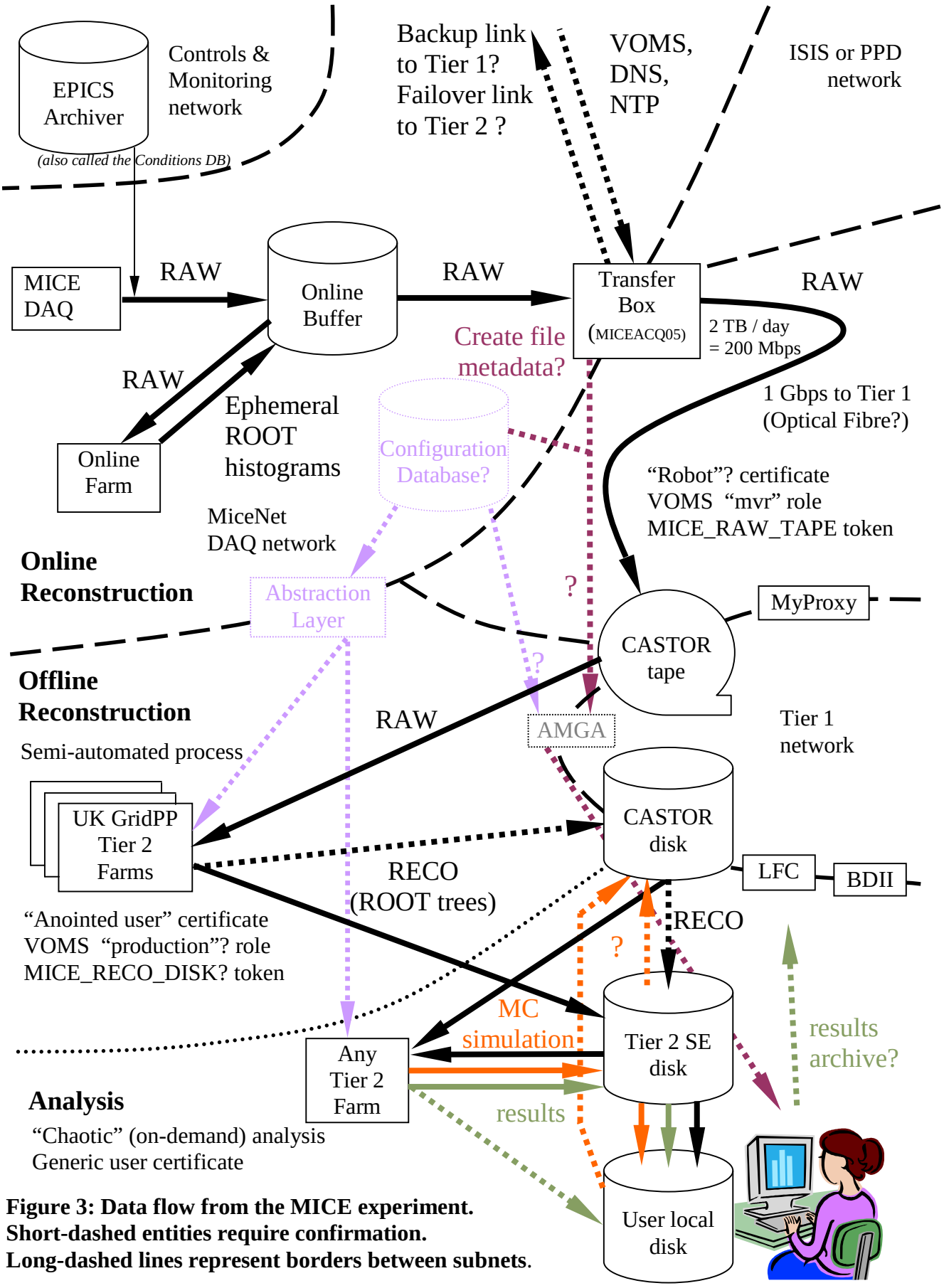



**Figure 3: Data flow from the MICE experiment. Short-dashed entities require confirmation. Long-dashed lines represent borders between subnets.**