

# The DatabaseUtil service

(+ EVD endpoint projection update)  
(+ Geant4 endless loop debugging)

Andrzej

# DatabaseUtil service

# Why? Why Now?

- A database is useful if you have parameters that can change in the course of your data taking and that influence the analysis.
- A prime example is the electron lifetime, which is important in calorimetry in a LArTPC but changes from run to run.
- The Calorimetry package needs this value as a parameter.
- A database eliminates the need of having a text file and makes it easier for different users to access this data.

# The DatabaseUtil service

- **New files in SVN Utilities:** `DatabaseUtil.cxx`, `DatabaseUtil.h`, `databaseutil.fcl` and `DatabaseUtil_service.cc`
- New service added to `services.fcl` and `evdservices.fcl`:  

```
DatabaseUtil:          @local::argoneut_database
```
- The service connects to a temporary PostgreSQL database (FNAL CD standard) and can read the parameters from there.
- For now, the idea is that the end user should rarely or never call the service explicitly. Instead this is taken care of by services which return detector or run properties.
- For example, the `LarProperties` service calls the database when a new run starts and sets the value of `fElectronLifetime` accordingly. Most code will only need to call `LarProperties::ElectronLifetime()`.
- This is (and will be for some time) a work in progress.

# The current ArgoNeut table

The table is filled outside of LarSoft.

( awk generated SQL script )

Currently the writing to the DB can be done mainly by specific users with writing privileges. These users need to be defined for each experiment or it can be decided to use a generic account. I would say this should be mostly an experiment\_offline domain?

```
. run | duration | pot | tottime | tau | tau_err | temp
. ----+-----+-----+-----+----+-----+-----
. 609 | 1277 | 175000 | 1277 | 732 | 29 | 88.4
. 618 | 1278 | 175000 | 2555 | 732 | 29 | 88.4
. 620 | 190 | 80000 | 3100 | 710 | 21 | 88.4
. 621 | 915 | 430000 | 4015 | 734 | 7 | 88.4
. 627 | 922 | 370000 | 5827 | 757 | 31 | 88.4
. 628 | 1443 | 810000 | 7270 | 765 | 5 | 88.4
. 629 | 1428 | 430000 | 8698 | 789 | 11 | 88.4
. 632 | 1428 | 430000 | 10126 | 789 | 11 | 88.4
. 634 | 1466 | 480000 | 12990 | 744 | 12 | 88.4
. 633 | 1398 | 800000 | 11524 | 737 | 13 | 88.4
. 635 | 1111 | 700000 | 14101 | 764 | 14 | 88.4
. 640 | 277 | 180000 | 14405 | 769 | 45 | 88.4
. 644 | 1519 | 740000 | 15935 | 763 | 10 | 88.4
. ...
. ...
. 842 | 1434 | 950000 | 221068 | 495 | 17 | 88.4
. 845 | 1205 | 940000 | 225389 | 559 | 21 | 88.4
. 846 | 468 | 380000 | 225868 | 571 | 36 | 88.4
. 847 | 874 | 750000 | 226759 | 584 | 14 | 88.4
. 849 | 1574 | 1.3e+06 | 228333 | 593 | 19 | 88.4
. 850 | 1282 | 900000 | 229615 | 622 | 14 | 88.4
. 851 | 72 | 30000 | 229687 | 554 | 22 | 88.4
. (135 rows)
```

# To be determined:

- One Database for each experiment or separate tables in one Database (I'd say separate Databases)?
- External LArSoft installs – do they hit the FNAL DB (probably needs IP enabling for each external install) – or should a separate copy of the DB be installed with an external version (need to solve DB synchronization between institutions)?
- Currently MC and data are not differentiated. This can cause problems – do we want this differentiation (possibly a separate table for MC) or is it up to the experiments to set up a run numbering scheme so that MC runs do not extract data run parameters?
- What happens if the DB is offline (currently defaults to previous values, this can cause problems)?
- DB backups – apparently this is handled by CD.

# ToDo list:

- Update the Wiki.
- Add Temperature and Electric Field to the readout options.
- Flesh out the DB design (possibly not detector agnostic → shift some responsibility to offline codes?)
- Separate experiments and migrate to proper devoted DB(s).
- Jon Paley from Argonne created a DB API for NOVA – I'm sure we can learn from his experiences for the development further down the road.

# Trouble shooting

- If you don't have Utilities and EventDisplay directories in your local directory you should be fine. Otherwise:

```
lar_update_testrel -rel development
```

and then:

```
lar_build -t
```

If you still get an error saying that the `<util::DatabaseUtil>` service is not found that means you need to add a line like:

```
DatabaseUtil:          @local::argoneut_database  
#or the      detector you're using
```

to the `.fcl` file where you define a list of services.

- Let me know if you encounter any problems!

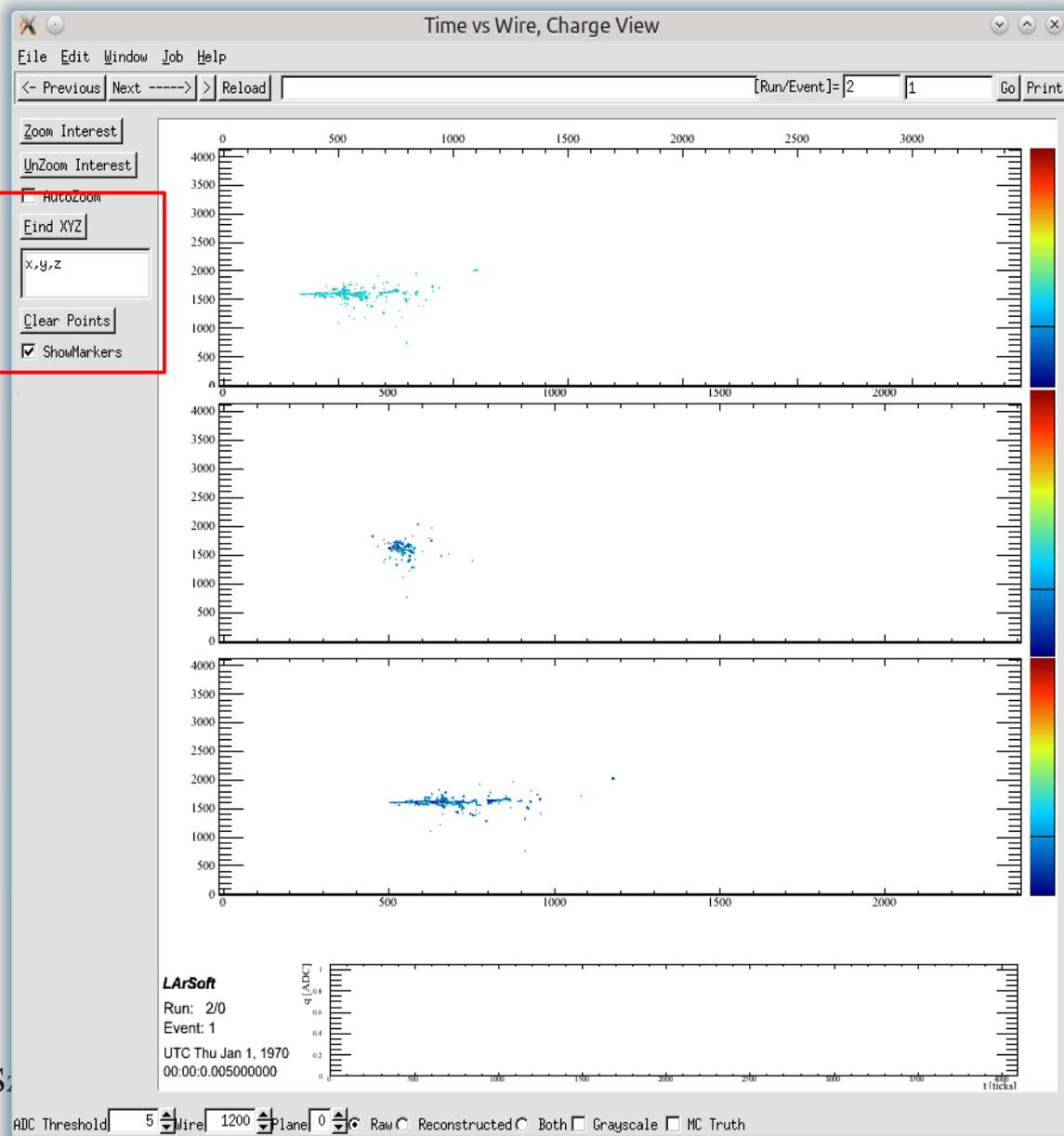
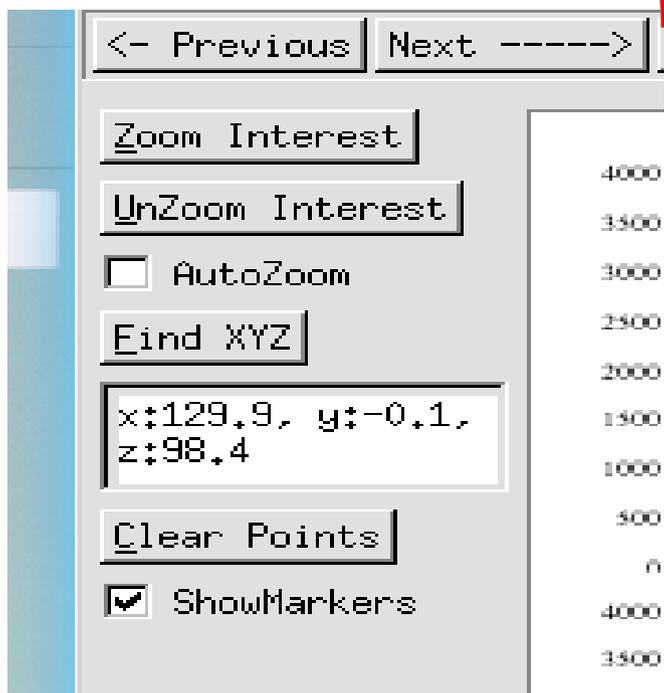
# Track End Point projection in the EVD

# Track EndPoint in a LArTPC

- It's always hard to understand where a given track ends in the TPC (at least for me) due to the wire plane geometry.
- This information is particularly useful for hand scans etc. since knowledge that a track is exiting can help understand the event.
- The idea is that you could click on a point and know if it is close to the edge or not (or where in the chamber).

# Addition to the EVD sidebar

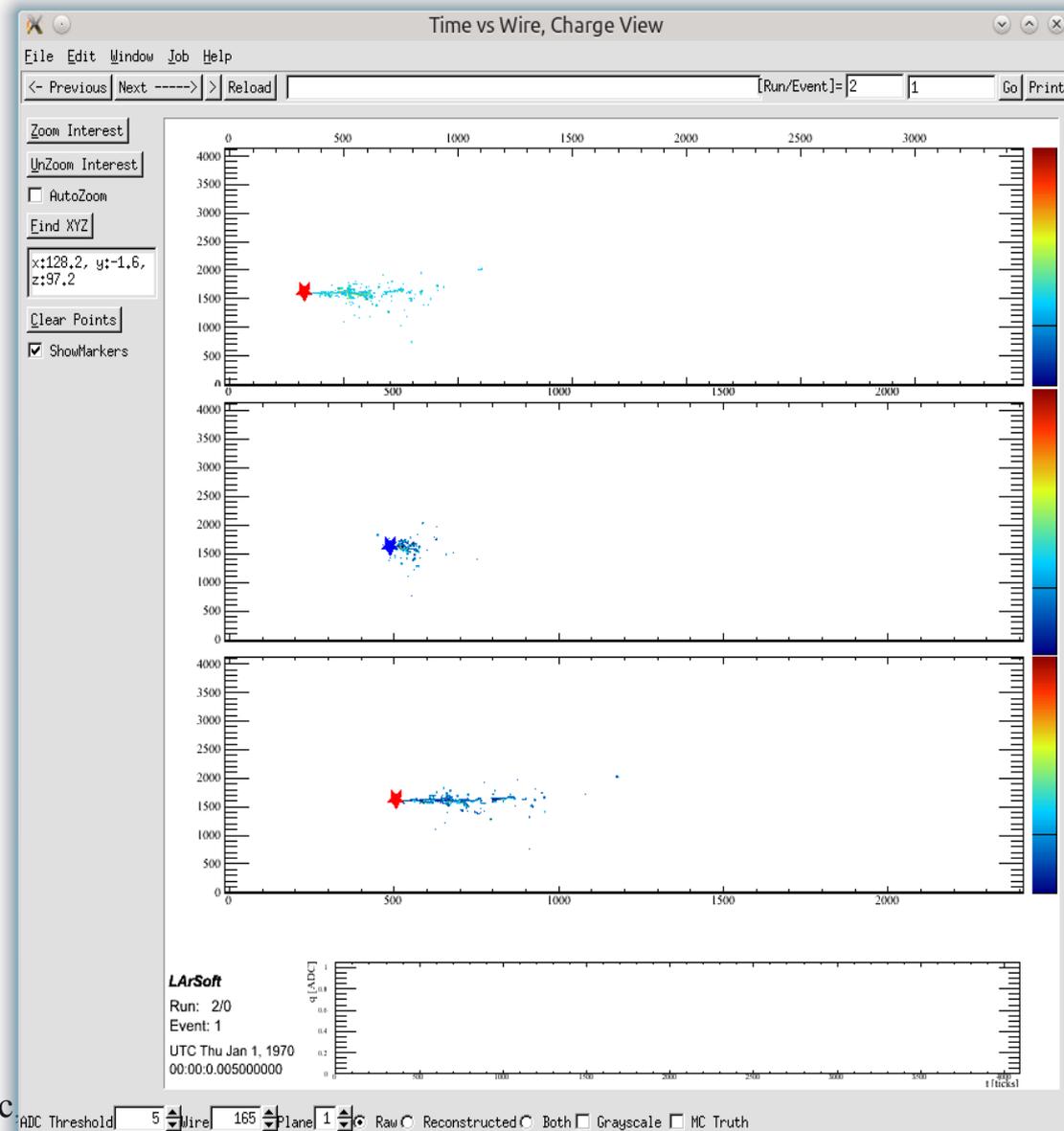
New elements in the sidebar + new mouse action.



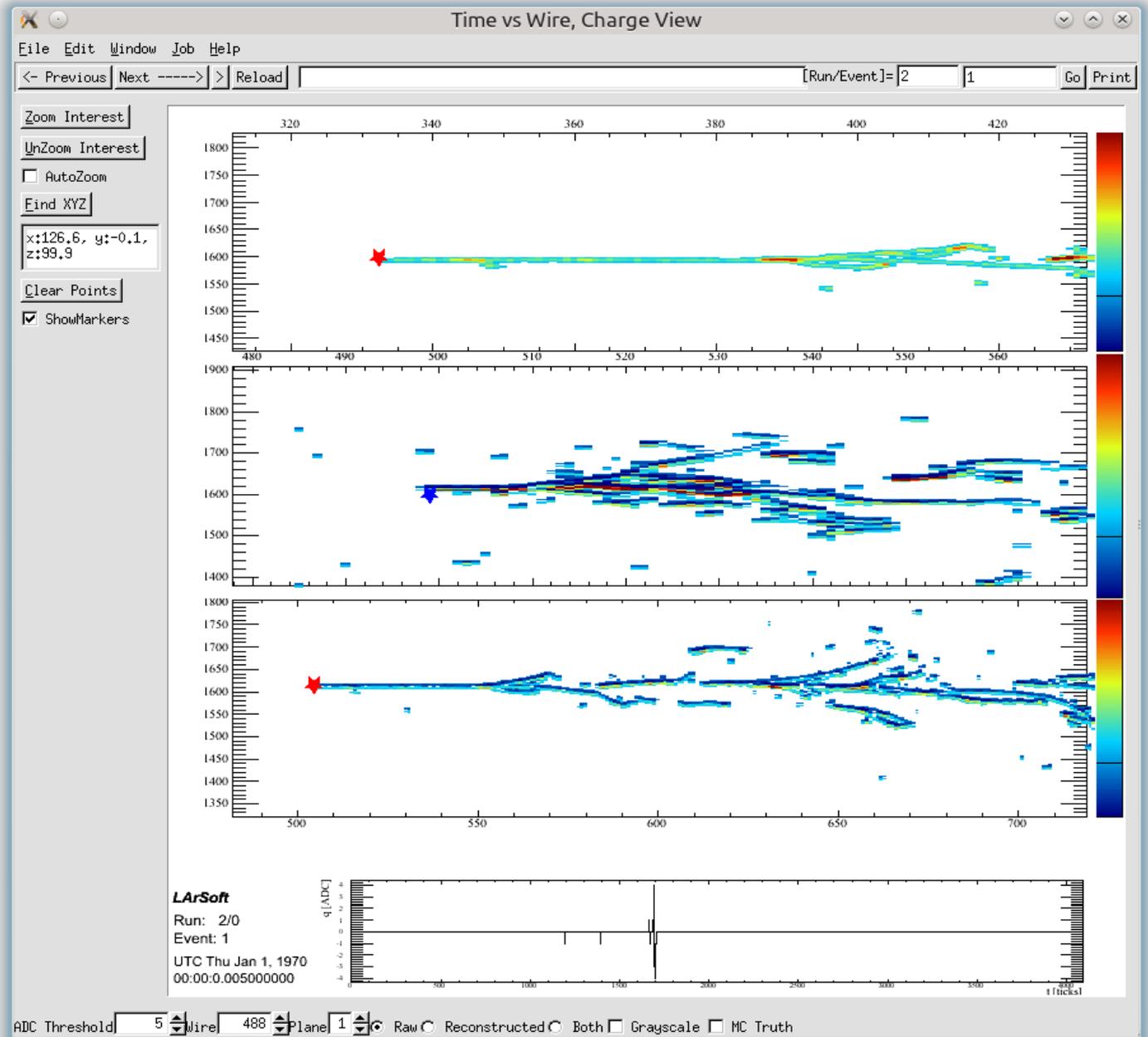
A. S.

# HowTo:

- Shift+L Mouse Click on points in two planes: red markers appear
- Click on Find XYZ button: real world x,y,z coordinates appear in the text display and a blue projection marker appears in the third plane.
- Any combination of two planes is ok.
- If wires don't cross or time distance is too long an error message gets printed in the text window.



# Zoom:



# Status:

- Checked in to SVN
- New options in evdservices.fcl:

standard\_evdlayoutopt:

```
{
  ShowSideBar:      1          # toggle extra sidebar visibility option
  AutoZoomInterest: 1          # toggle the auto zoom to interesting
                          region option
  PrintTotalCharge: 1          # Print out the sum of collected charge
  ShowEndPointSection: 1      # Show Sidebar section with EndPoint
                          extrapolation
  ShowEndPointMarkers: 1      # toggle visibility of markers for
                          EndPoint finding
}
```

- Turned off by default for ArgoNeut.
- I noticed that on some screens the text display window adds scrollbars – will try to understand and fix this.

# Geant4 endless loop debugging

# When running jobs (usually $e^-$ )

- Jobs sometimes seem to hang.
- When pouncing on them with gdb, the symptoms showed that the Geant4 stepsize was infinitesimally small  $\sim 10^{-14}$ .
- It seems that a particle gets stuck at a voxel boundary (one of coordinates pos. is a multiplier of  $\sim 0.3$  mm)
- This particle is usually a photon (but not always)
- This problem has apparently been solved in the newer versions of Geant?

# What is happening

- Presumably the particle gets stuck at the edge of a Geant4 intrinsic voxel and does not calculate the step correctly.
- Since this happens most often for photons it is most often present in EM shower events.
- Worst case: up to 3-4 events in 20 jobs of 500 showers (ArgoNeut).
- This sometimes happens for particles where `fparticle==0` and so do not enter `SteppingAction(...)` in `ParticleActionList.cxx`

# Temporary Solution in ParticleListAction.cxx

- The solution is thanks to Bill Seligman.
- Applied only if the step number is over 50k AND the step number is smaller  $\sim 10e-12$  mm (usual values are 2-3k and  $\sim 0.1$  mm )
- The particle gets a kick by (0.001,0.001,0.001) mm and a warning is printed out:

```
##### In endless loop. Kicking particle by  
(+0.001,+0.001,+0.001)-- fparticle: 0x1eaba7f0 PDG and  
encoding 22 gamma current step number: 50001 stepsize:  
2.32731e-14 x,y,z 265.628 106.5 521.55
```

- Don't try this at home! The fix is temporary and should go away, once we migrate to a newer geant4 version.
- I have run over a 100 jobs with over a third e showers. As of this writing the problem did not repeat.