Database Service Interface Update

Brandon Eberly

August 11, 2015

The plan

- Larsoft will provide database retrieval services for two distinct ways of using the Fermilab conditions database system
  - The single-IOV DB used by MicroBooNE, based on a design for MINERvA
    - All channels/elements in a DB folder have identical intervals of validity
  - The multi-IOV DB used by DUNE, based on a design for NOvA
    - Channels/elements may have different intervals of validity
- These two collections of services should be designed such that most future larsoft experiments can use them without modification
- Since these services and their helper classes will be retrieving the same kinds of information, it makes sense to have common interfaces
  - Provides a template for future experiments to create their own implementations, should their needs not be met by Larsoft

The plan

One DataProviderInterface per database folder

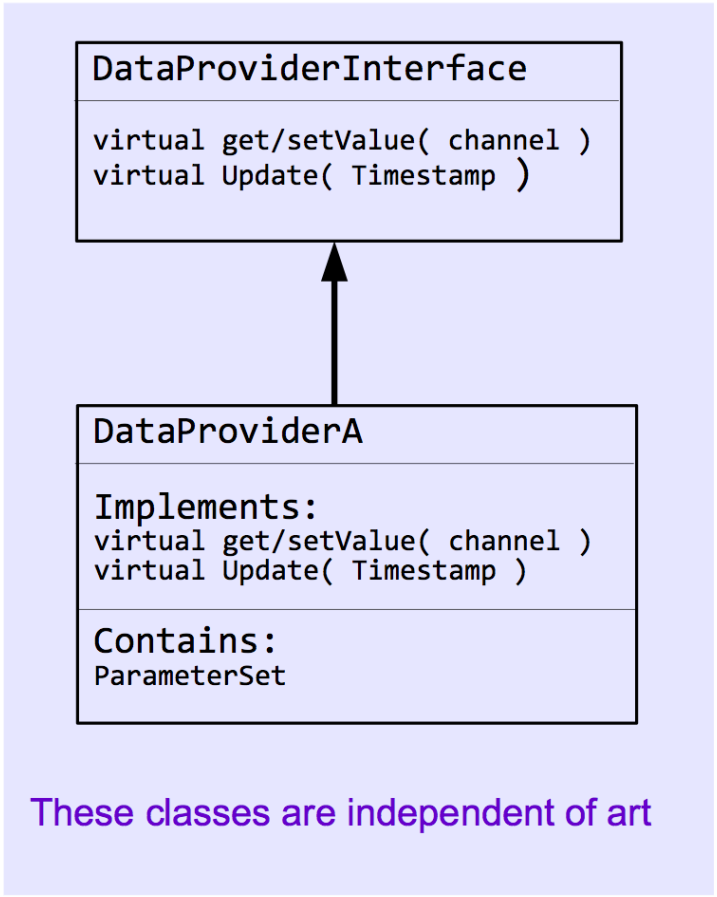
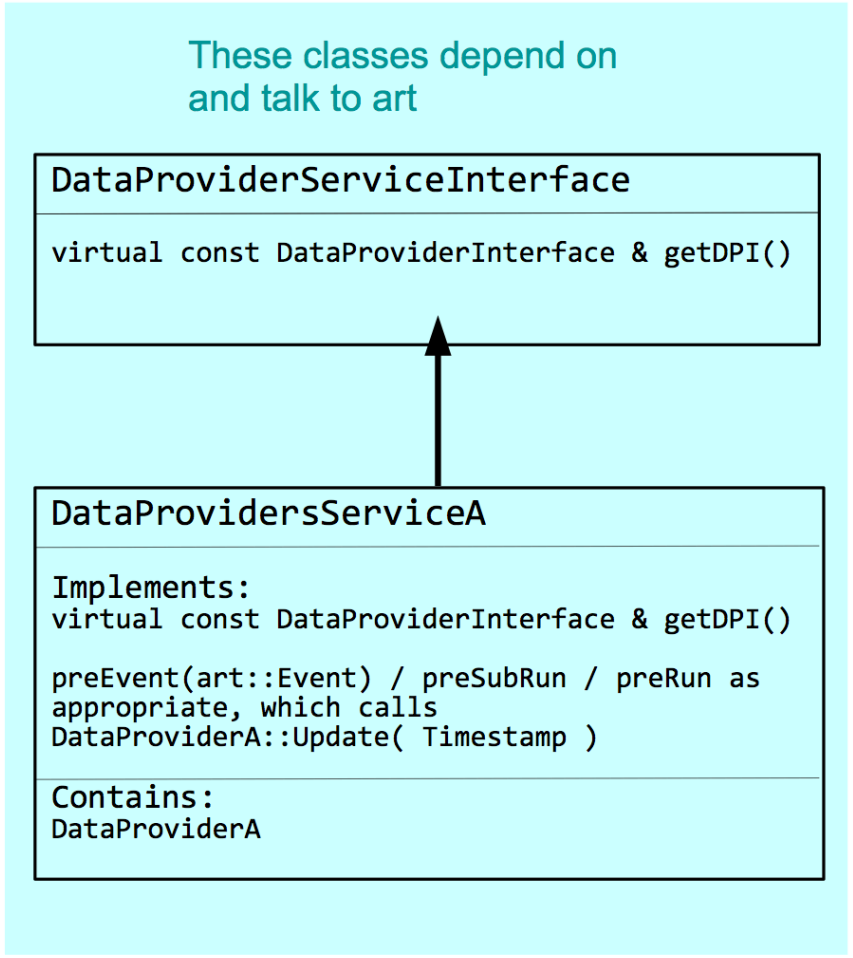


diagram by E. Snider

One service interface per database folder



Pedestal Provider Interface

This interface is for a class that is able to retrieve pedestals from a database and maintain a local cache of pedestal constants (larevt/CalibrationDBI/Interface/IDetPedestalProvider.h):

```
class IDetPedestalProvider {

  public:

    /// Retrieve pedestal information
    virtual float PedMean(std::uint64_t ch) const = 0;
    virtual float PedRms(std::uint64_t ch) const = 0;
    virtual float PedMeanErr(std::uint64_t ch) const = 0;
    virtual float PedRmsErr(std::uint64_t ch) const = 0;

    /// Update local state of implementation
    virtual bool Update(std::uint64_t ts) = 0;
};
```

The single-IOV pedestal retrieval class in larsoft has been converted to an implementation of this interface (larevt/CalibrationDBI/Providers/DetPedestalRetrievalAlg.h):

```
class DetPedestalRetrievalAlg : public DatabaseRetrievalAlg, public IDetPedestalProvider
```

Base class to be used by all single-IOV provider implementations

Pedestal Service Interface

This interface is for a service that maintains and gives access to a detector pedestal provider (larevt/CalibrationDBI/Interface/IDetPedestalProvider.h):

```
class IDetPedestalService {

  public:

    virtual ~IDetPedestalService() = default;

    //retrieve pedestal provider
    const IDetPedestalProvider& GetPedestalProvider() const {
      return this->DoGetPedestalProvider();
    }

  private:

    virtual const IDetPedestalProvider& DoGetPedestalProvider() const = 0;
};
```

Single-IOV Pedestal Service Impl

And larsoft now provides the single-IOV style service implementation (larevt/CalibrationDBI/Services/SIOVDetPedestalService_service.cc):

```
class SIOVDetPedestalService : public IDetPedestalService {

  public:

    SIOVDetPedestalService(fhicl::ParameterSet const& pset, art::ActivityRegistry& reg);
    ~SIOVDetPedestalService(){}

    void PreProcessEvent(const art::Event& evt) {
      // This is a temporary kludge to allow microboone to analyze early data
      // which did not have a proper timestamp in the daq header.
      // NOTE: it would be nice if there was a way to check that the art::Event
      // is microboone or not so that this kludge does not affect other experiments
      if (evt.isRealData() && evt.run() < 183) {
        std::uint64_t kludge_stamp = 1430000000000000000; //yes, there really needs to be 16 zeroes
        fProvider.Update(kludge_stamp);
      }
      else fProvider.Update(evt.time().value());
    }

  private:

    const IDetPedestalProvider& DoGetPedestalProvider() const override {
      return fProvider;
    }

    DetPedestalRetrievalAlg fProvider;
};
```

Using the Single-IOV implementation

- The previous code is on larevt feature branch eberly_dbinterface and is ready to be merged with develop

- lareventdisplay and uboonecode are both updated to use the single-IOV pedestal retrieval service implementation
  - both on feature branch eberly_dbinterface

- It is also possible to just use the pedestal providers outside of the service
  - WARNING: If you wrote code that uses DetPedestalRetrievalAlg, you will need to update your code (see next slide) and fcl files to use the service, or:

1) In CMakeLists.txt, change:

   CalibrationDBI_WebDBI → CalibrationDBI_Providers

2) In your code:

```
#include "CalibrationDBI/Providers/DetPedestalRetrievalAlg.h"
//....
fPedestalRetrievalAlg.Update( evt.time().value() );
```

Using the Single-IOV implementation

- Using DetSim/SimWireMicroBooNE_module.cc in uboonecode as an example:

- Include the interfaces:

```
#include "CalibrationDBI/Interface/IDetPedestalService.h"
#include "CalibrationDBI/Interface/IDetPedestalProvider.h"
```

- Then ask for the service handle:

```
//get pedestal conditions
const lariov::IDetPedestalProvider& pedestalRetrievalAlg =
      art::ServiceHandle<lariov::IDetPedestalService>()->GetPedestalProvider();
```

- And use it!

```
//Generate Noise:
//Pedestal determination and random gaussian variation
art::ServiceHandle<art::RandomNumberGenerator> rng;
CLHEP::HepRandomEngine &engine = rng->getEngine("pedestal");
CLHEP::RandGaussQ rGaussPed(engine, 0.0, pedestalRetrievalAlg.PedRms(chan));
float ped_mean = pedestalRetrievalAlg.PedMean(chan) + rGaussPed.fire();
```

Using the Single-IOV implementation

•fcl configuration: I'll assume that you understand the basics of configuring a service via fcl

•Just remember to specify which implementation of the service you wish to use!

•Example (uboonecode: services_microboone_simulation.fcl)

```
# Define microboone_detsim_dark_services ... (2)
microboone_detsim_dark_services:                          @local::microboone_basic_services
microboone_detsim_dark_services.LArFFT:                   @local::microboone_larfft
microboone_detsim_dark_services.SignalShapingServiceMicroBooNE: @local::microboone_signalshapingservice
microboone_detsim_dark_services.IDetPedestalService:      @local::microboone_detpedestalservice
```

•Example (larevt: database_standard.fcl)

```
standard_siov_detpedestal_service:
{
  service_provider: SIOVDetPedestalService
  DetPedestalRetrievalAlg: @local::standard_pedestalretrievalalg
}
```

More on FCL

- From evdservices_microboone.fcl in uboonecode

```
microboone_disp:
{
 ExptGeoHelperInterface:    @local::microboone_geometry_helper
 Geometry:                  @local::microboone_geo
 LArProperties:             @local::microboone_properties
 DetectorProperties:        @local::microboone_detproperties
 DatabaseUtil:              @local::microboone_database
 ColorDrawingOptions:       @local::microboone_colordrawingopt
 SimulationDrawingOptions:  @local::microboone_simdrawingopt
 RawDrawingOptions:         @local::microboone_rawdrawingopt
 RecoDrawingOptions:        @local::microboone_recodrawingopt
 AnalysisDrawingOptions:    @local::microboone_analysisdrawingopt
 EvdLayoutOptions:          @local::microboone_evdlayoutopt
 EventDisplay:              @local::microboone_evd
 ScanOptions:               @local::microboone_scanopt
 LArG4Parameters:           @local::microboone_largeantparameters
 LArVoxelCalculator:        @local::microboone_larvoxelcalculator
 InfoTransfer:              @local::microboone_infotransfer
 TimeService:               @local::microboone_timeservice
 #IDetPedestalService:       @local::microboone_detpedestalservice
}
microboone_disp.IDetPedestalService: @local::microboone_detpedestalservice
```

Library problem when this is uncommented

Other Changes

- You may have noticed other changes to the single-IOV pedestal provider:
  - Update() now takes a uint64_t timestamp
  - The uint64_t timestamp is converted to a DB-friendly timestamp ONLY after checking whether the database is being used
    - No more timestamp-related crashes for jobs that do not use the database
  - Conversion from uint64_t timestamp to DB-friendly timestamp is handled by the single-IOV implementation
    - All experiments using the larsoft single-IOV are forced to use this conversion
    - Not ideal, but can only be avoided by making the data provider art-dependent (implement timestamp conversion as a service)

Next

- Write single-IOV implementations of the larproperties and detectorproperties interfaces (waiting for Jon Paley to finish the interfaces)

- Organization: can larevt/CalibrationDBI be the home for all DB-related classes?
  - three directories: Interface, SingleIOV, MultiIOV

- I will work on other interfaces (e.g., PMTs) and single-IOV database write scripts as demanded by MicroBooNE
  - Maybe 35t will want some interfaces first?