Teaching Database Services to be Lazy

Larsoft Coordination Meeting
Nov. 1, 2018

H. Greenlee

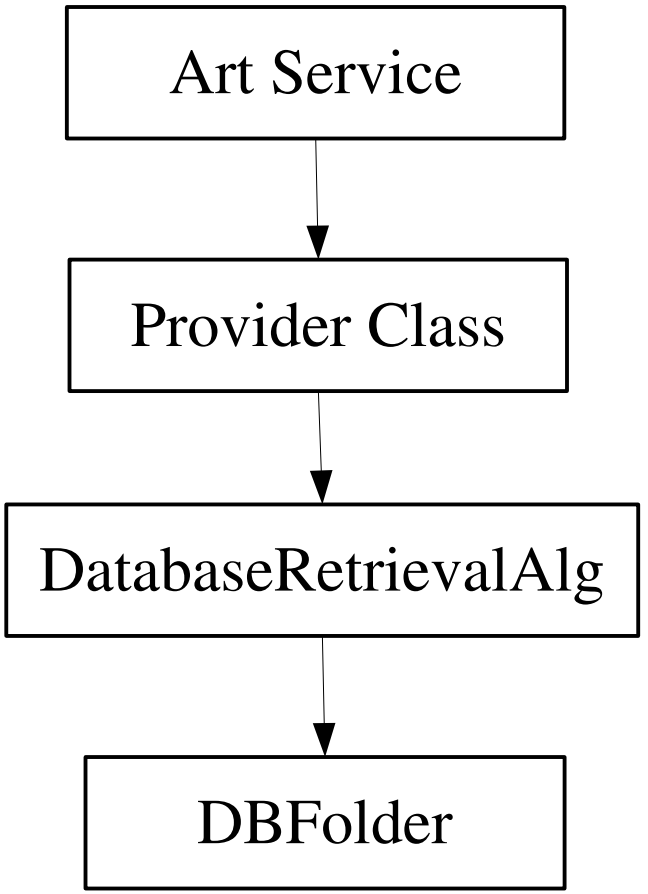
Outline

- Background.
- How IOV database services work currently.
- Proposed changes to IOV database services.

Background

- There was recently an incident where a MicroBooNE database server became overloaded, such that queries started failing.
  - A user submitted a large number of I/O intensive jobs that started hitting the database server at the same time, and the database server couldn't handle the load.
  - Not the user's fault. The user was running a standard fcl that configured a number of database services that weren't actually needed or used.
    - Unneeded services accessed the database anyway.
- Problem could have been avoided by using a fcl file that configured only needed services (difficult in practice).
- Better solution is to teach services to not do expensive tasks, such as database access, if they don't need to.

How Database Services Work Currently



How Database Services Work Currently (In More Detail)

- Art Service.
  - Owns instance of provider.
  - PreProcessEvent callback calls provider function Update(timestamp).
- Provider class.
  - Caches data that depends on DB contents.
  - Derives from base class DatabaseRetrievalAlg.
  - Update function calls DatabaseRetrievalAlg::UpdateFolder(timestamp).
- DatabaseRetrievalAlg.
  - Owns DBFolder instance.
  - Understands IOVs.
  - UpdateFolder function calls DBFolder::UpdateData(timestamp).
- DBFolder.
  - Does actual DB query via DB server.

Proposed Changes to Providers

- New functions and data members.
  - Add data member fCurrentTimeStamp – Time stamp of cached data.
  - Add data member fEventTimeSTamp – Time stamp of most recent event.
  - Add function UpdateTimeStamp(timestamp)
    - Updates fEventTimeStamp.
  - Add private const DoUpdate(timestamp) function.
    - Does actual update and updates fCurrentTimeStamp (if necessary).

Proposed Changes to Providers (cont.)

- Changes to existing functions and data members.
  - Update(timestamp) function (public, non-const) calls DoUpdate(timestamp) (private, const).
  - Make all data members modified by DoUpdate(timestamp) mutable.
  - Call DoUpdate(fEventTimeStamp) function from all other functions (accessors) that depend on mutable data members.

Proposed Changes to Services

- Modify PreProcessEvent callback to call provider function UpdateTimeStamp(timestamp) instead of Update(timestamp).

Commentary on Proposed Changes

- Nonbreaking interface change.
  - Only visible change is addition of new fuction UpdateTimeStamp.
    - Functions UpdateTimeStamp and Update do the same thing. The former is lazy about database access, but the latter is not.
      - Quiz question: why do there still need to be two functions?
- Art service PreProcessEvent callback becomes inexpensive because it can never trigger a database read.
  - Database access triggered by accessor function call.
- Existing callers of provider function Update(timestamp) might benefit by calling UpdateTimeStamp(timestamp) instead, but they won't break if they don't make this change.

Commentary on Proposed Changes (cont.)

- Risks.
  - During initial conversion, someone might forget to add a call to DoUpdate(timestamp) in a provider function that requires it.
  - Under maintenance, someone might adds a provider function and not add a required call to DoUpdate(timestamp).

Services and Providers (Owned by larsoft)

- In larevt/larevt/CalibrationDBI.
  - SIOVChannelStatusService / SIOVChannelStatusProvider.
  - SIOVDetPedestalService / DetPedestalRetrievalAlg.
  - SIOVElectronicsCalibService / SIOVElectronicsCalibProvider.
  - SIOVPmtGainService / SIOVPmtGainProvider.

Services and Providers (Owned by uboone suite)

- In uboonecode/uboone/Database or ubevt/ubevt/Database.
  - UbooneChannelStatusService / SIOVChannelStatusProvider.
  - UbooneDetPedestalService / DetPedestalRetrievalAlg.
  - UbooneElectronLifetimeService / UbooneElectronLifetimeProvider.
  - UbooneElectronicsCalibService / UbooneElectronicsCalibProvider.
  - UboonePmtGainService / UboonePmtGainProvider.
  - UbooneTPCEnergyCalibService / UbooneTPCEnergyCalibProvider.
  - WireCellChannelStatusService / WireCellChannelStatusProvider.

Status

- Ideas presented in this talk have been test by me using service / provider TPCEnergyCalib.
  - On uboonecode branch feature/greenlee_lazy_db (one commit).
  - This is the service that hits the database the hardest (and the service that triggered the database server overload).
- I propose that the four IOV services/providers owned by larsoft (as well as all seven IOV services/providers owned by uboone) be modified to be lazy.
  - Other experiments could possibly benefit from making similar changes to their services/providers.