Algorithms and Conditions in AthenaMT

Summary

Charles Leggett

June 10 2016

Glasgow TIM

THistSvc for AthenaMT

- default behaviour
  - each cloned Algorithm gets it's own copy of a Histogram
    - Hists in cloned Algorithms are made memory-resident
  - at `finalize()`, "primary" Algorithm asks THistSvc for merged Histogram (if merge is trivial)
    - if merge is non-trivial, Algorithm will get back all Hists with same id from THistSvc, and do merge itself
    - memory-resident clones are deleted, only primary Hist is written out
- new interface: `THistSvc::regSharedHist(TH1*& hist,...)`
  - all clones get same instance of hist
  - during `execute()` (or any other concurrent access), Algorithm must lock hist before writing to it
    - via some type of smartPtr or LockGuard
- THistSvc still needs to be made thread safe in general, as we can't assume that `regHist()` won't be called from `execute()`

VarHandleArrays

- yes
- I'll put something together next week

Update Handles

- How to add decorations to an object already in the store (const)?
  - and avoid implementing an UpdateVarHandle....

- Possible solution
  - new type `DecoratedHandleKey<foo>("SGKey","SGKey.decoration");`
    - adds a Read dependency on "SGKey", and Write dependency on "SGKey.decoration" to the Scheduler
  - when DecoratedHandle gets written, makes an alias in StoreGate with "SGKey.decoration"
    - done automatically during StoreGate::commit() phase at end of Algorithm::execute?
  - downstream Algs use ReadHandleKey("SGKey.decoration")

Data Sharing Between Clones

- no extensive discussion
- please add comments to accumulator proposal at https://gitlab.cern.ch/gaudi/Gaudi/merge_requests/176
- unanswered question:
  - how to handle shared data in re-entrant Algs, since execute() is const?

Conditions

- Now that we've successfully migrated a few clients, next step is to make new infrastructure work in serial Athena
  - need to add CondInputLoader to start of topSequence
  - other CondAlgs need to be added to topSequence (before regular Algs)
    - new main sequence CondSeq that's execute before topSequence?
  - ensure that CondAlgs return quickly if all their conditions data is still valid
    - could optimize this by putting all CondAlgs into a sequence, and triggering them externally, just like the IOVSvc does now with the callbacks
  - Vakho has a prototype that gets all the way to finalize before segfaulting!
- Walter suggested a few more simple clients in Calo that we can start migrating
- We can keep a lot of existing access methods by clients of conditions data by careful refactoring of code when the CondAlgs are written