Interval of Validity Service
IOVSvc

EDM Workshop

Detector Description Session

01/27/2003

Charles Leggett <CGLeggett@lbl.gov>

Interval of Validity Service

- Purpose:
  - associates valid time ranges with objects
  - triggers updates of data, and validity ranges when object enters a new validity range
  - allows use of call back functions

- Use cases:
  - alignment data
  - calibration objects
  - detector description information
  - anything which has a timed validity range associated with it

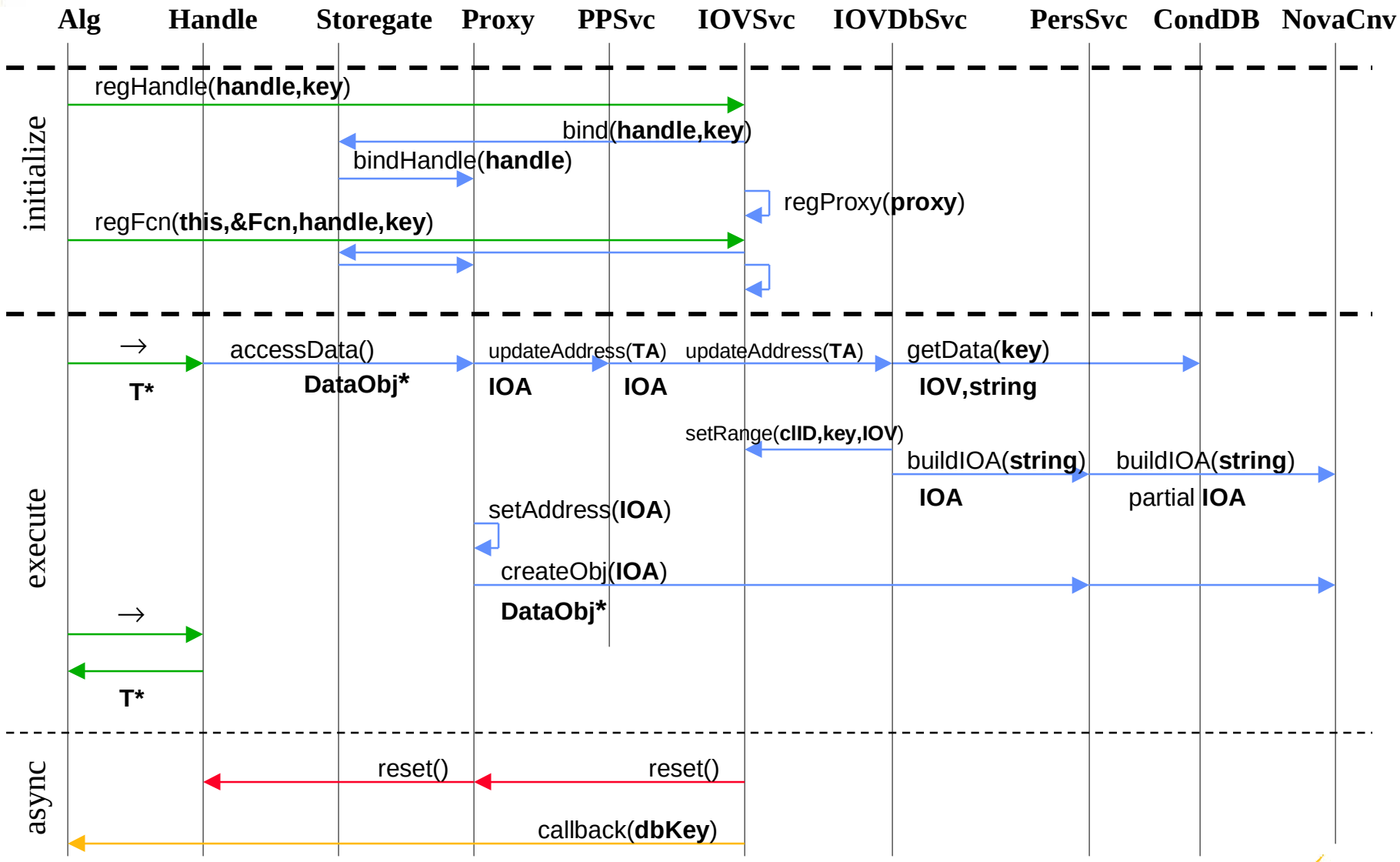
Access Patterns

- Two types of access/usage patterns:
  - pure data, held in dB, used as a data member inside a class. Only needs to have it's contents refreshed when it enters a new validity range.
  - Container class, eg GeoNode, which has no representation inside the dB, contains one or more IOV data objects from dB as data members. Needs to know when any of its constituent members enters a new validity range, at which point a callback method is triggered.
- No overlap between these patterns
  - will never have an IOV data object which needs a callback
  - will never have a container class which has data stored in dB

Class Example

```
class myIOVClass {
   public:
      StatusCode CallBackFcn(list<string> &dbKeys);

   private:
      const DataHandle<CalAlign> m_align1;
      const DataHandle<CalAlign> m_align2;

      myIOVAlg2* that;
};

myIOVClass::initialize() {

   p_IOVSvc->regHandle(m_align1, dBkey1);
   p_IOVSvc->regHandle(m_align2, dBkey2);

   p_IOVSvc->regFcn(this, &myIOVClass::CallBackFcn, m_align1);
   p_IOVSvc->regFcn(this, &myIOVClass::CallBackFcn, m_align2);

   p_IOVSvc->regFcn(this, &myIOVClass::CallBackFcn,
                    that, &myIOVClass2::OtherCallBackFcn);
}
```

Sequence Diagram



Sequence

- In `Initialize()`, alg registers `DataHandle<alignData>` with the IOVSvc using `IOVSvc->regHandle(handle,key)`, using the same dBkey as is used in the dB.
- Alternatively, the algorithm registers container class (eg GeoNode) with `IOVSvc->regFcn()`, supplying callback fcn and DataHandles as parameters.

- At first deref of the DataHandle, call is passed through to Proxy, which calls `getAddress(TransientAddress)` on the IOVDbSvc. IOVDbSvc talks to the CondDB to retrieve storage location of an element given a `(clID,dbKey)`pair. It gets back an IOV and a string which points to the actual location of the data from which an IOA can be built. It then calls `setRange()` on the IOVSvc, which adds the IOV to its list of registered intervals. The IOA is set in the TransientAddress, and then the DataProxy calls `createObj()` on the persistency converter for the actual data using this information, and gets back a DataObj*. The DataObj* is passed back through the chain of sequences to the Algorithm, eventually being converted to a T*.

Sequence (cont)

- The next time the DataHandle is derefed, a T* is automatically returned from the Proxy.
- At the start of each event, the IOVSvc parses its list of registered entries to see if any are out of range. When it finds one, it first resets the corresponding proxy, then triggers the registered callback function. The callback function gets a list of the keys of the items which have gone out of range as a parameter.
- The next time one of the DataHandles is derefed, the initial sequence is replayed, getting back a new IOV and data.

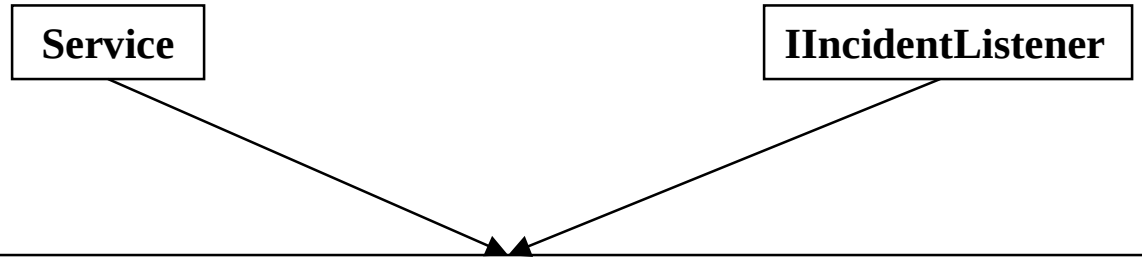
Things Needed

- IOVDbSvc
  - provided by Database people - see R.D.'s talk next.
  - Can use **AtlasTest/IOVASCIIDbSvc** instead, which keeps IOV Range info in a hierarchical ASCII file based format. See that package for details.
- A converter/conversion svc for your class.
  - an example can be found in AtlasTest/SiAlignTest

Changes to DataHandle *et al*

- DataHandle
  - inherits from **`IResetable`**, declares reset() method
- DataProxy
  - list of bound handles (`list<IResetable*>`)
  - `bindHandle(IResetable*), unbindHandle(IResetable*)`
  - `resetBoundHandles()`
- StoreGate
  - ProxyProvider
  - `bind()`
    - like retrieve, but also associates Handles with Proxy

IOVSvc




```
IOVSvc {
public:
   StatusCode regHandle(DataHandle<H> handle, string key);
   StatusCode regFcn(void (T::*fcn)(), T* obj, DataHandle<H> handle, string key);
   StatusCode regFcn(void (T1::*fcn1)(), T1* obj, void (T2::*fcn2)(), T2* obj);
private:
   StatusCode regProxy(DataProxy *);
   set<DataProxy*> m_proxies;
   map<DataProxy*, IOVEntries* > m_entries;
   multimap<DataProxy*, boost::function<StatusCode>* > m_proxyMap;
   multiset<IOVEntry, IOVEntryStartCriterion>  m_startSet;
   multiset<IOVEntry, IOVEntryStopCriterion> m_stopSet;
};
```

Associated Classes

IOVTime

```
IOVTime(int)
IOVTime(EventID)
unsigned long long m_time
```

IOVRange

```
IOVTime m_start
IOVTime m_stop
```

IOVEntry

```
DataProxy* m_proxy
IOVRange*  m_range
```

Internals of IOVSvc

- indexing is done via `DataProxy*`
- IOVs are held in `map<DataProxy*, IOVEntry*>`
- Callback functions are
  `multimap<DataProxy*, boost::function<StatusCode>* >`
- IOVs are ordered in two sets:

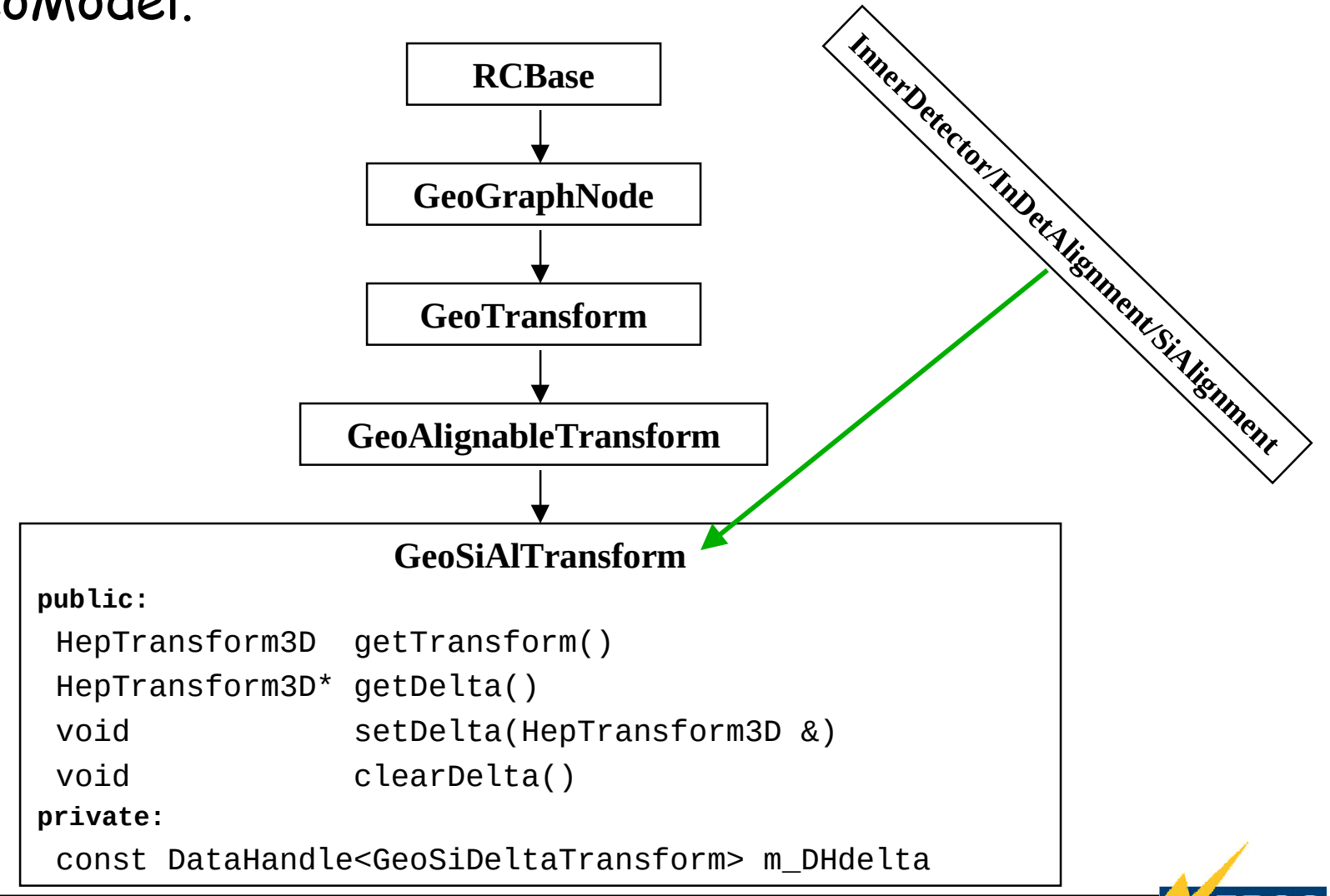
```
std::multiset<IOVEntry*,IOVEntryStartCritereon> m_startSet;
std::multiset<IOVEntry*,IOVEntryStopCritereon>  m_stopSet;
```

`m_startSet` is ordered by decreasing start time, `m_stopSet` is ordered by increasing end time. As a result of this ordering, when the sets are scanned at the beginning of each event for entries that need to be marked invalid, as soon as the first valid entry is found in each set, the scanning can be stopped as one is assured that all subsequent entries are valid.

- When invalid entries in these multisets are found, the associated DataProxy* is added to a list. At the end of the scan, the IOVs of these DataProxies are updated, the DataProxies are reset (and thus the associated DataHandles) and the associated callback functions are activated.

A Real Example: Pixel Detector

- Used to handle automatic alignments with the GeoModel.



How It Works

- The GeoSiAlTransform is associated with a GeoFullPhysVolume, which has a cache for the alignment data.
- When this alignment data is needed, `getDelta()` in GeoSiAlTransform is called, which in turn derefs the DataHandle `m_DHdelta`. This DataHandle has been registered with the IOVSvc, so the IOVSvc is able to manage its contents, updating it as necessary.
- The method `clearDelta()` has also been registered as the accompanying callback function for this DataHandle, and is called when a new alignment is read in. It invalidates the position caches for all elements in the detector hierarchy below it, so that the next time the position of any of them is needed, it is recalculated with the updated information.

Colours provided by the colourized MsgService, coming soon to a terminal near you!

Example Output

Alignment first checked at event [1,5]

db file opened, appropriate range found ([1,4] -> [1,8]), tag “tag2” read in

Position loaded from converter (ideal + delta), identified by “tag2”

Event [1,6] - start and stop sets show valid ranges for this detector element

Event [1,10] - detector element is out of range, has been removed from stop set

db file found and opened, next valid range found ([1,9]->[2,2]), tag “tag3” read in.

Position loaded from converter, is now different from before (the delta has changed)

Event [2,2] - start and stop sets show valid ranges for this detector element

```
:0.0 /afs/cern.ch/atlas/maxidisk/d90/5.2.0/TestRelease/TestRelease-00
IOVSvc              DEBUG Begin Event: [R/E] = [1,5]
GetSi1               INFO gettting position!!
GetSi1              DEBUG detector 2/1/0/2/5/0
IOVASCIIDbSvc     VERBOSE updateAddress: CLID: 2530  key: 2/1/0/2/5/0
IOVASCIIDbSvc       DEBUG found db file for 2_1_0_2_5_0
IOVASCIIDbSvc       DEBUG   tag: tag1  start: [0,0]  end: [1,3]  in: 0
IOVASCIIDbSvc       DEBUG   tag: tag2  start: [1,4]  end: [1,8]  in: 1 <---
IOVSvc              DEBUG setRange(): 2530 2/1/0/2/5/0  start: [1,4] end: [1,8]
GetSi1              DEBUG Absolute position 96.8102  75.6722  0

IOVSvc              DEBUG Begin Event: [R/E] = [1,6]
IOVSvc              DEBUG start set:
  DataHandle<GeoSiDeltaTransform,>[2/1/0/2/5/0] (0x8322d40) [1,4]
IOVSvc              DEBUG stop set:
  DataHandle<GeoSiDeltaTransform,>[2/1/0/2/5/0] (0x8322d40) [1,8]

IOVSvc              DEBUG Begin Event: [R/E] = [1,10]
IOVSvc              DEBUG start set:
  DataHandle<GeoSiDeltaTransform,>[2/1/0/2/5/0] (0x8322d40) [1,4]
IOVSvc              DEBUG stop set:
  -- empty --
GetSi1               INFO gettting position!!
GetSi1              DEBUG detector 2/1/0/2/5/0
IOVASCIIDbSvc     VERBOSE updateAddress: CLID: 2530  key: 2/1/0/2/5/0
IOVASCIIDbSvc       DEBUG found db file for 2_1_0_2_5_0
IOVASCIIDbSvc       DEBUG   tag: tag1  start: [0,0]  end: [1,3]  in: 0
IOVASCIIDbSvc       DEBUG   tag: tag2  start: [1,4]  end: [1,8]  in: 1
IOVASCIIDbSvc       DEBUG   tag: tag3  start: [1,9]  end: [2,2]  in: 2 <---
IOVSvc              DEBUG setRange(): 2530 2/1/0/2/5/0  start: [1,9] end: [2,2]
GetSi1              DEBUG Absolute position 95.8596  75.3619  1

IOVSvc              DEBUG Begin Event: [R/E] = [2,2]
IOVSvc              DEBUG start set:
  DataHandle<GeoSiDeltaTransform,>[2/1/0/2/5/0] (0x8322d40) [1,9]
IOVSvc              DEBUG stop set:
  DataHandle<GeoSiDeltaTransform,>[2/1/0/2/5/0] (0x8322d40) [2,2]
```

example taken from running package **AtlasTest/SiAlignTest**

Caveats

- The callback function must be virtual.
- All DataHandles to elements that the IOVSvc will manage must be const, or you'll get a runtime error.
- Don't try to deref the DataHandle before the first event is loaded, eg in the initialize phase. The proxy hasn't been filled, and you'll get a pointer to a null object.
- Whenever you call `regFcn(...)`, you will get a warning that looks like this:

```
...lots of crap ... instantiated by ... instantiated by ...
../../../Control/IOVSvc/IOVSvc-00-00-07/IOVSvc/CallBackID.icc:24:
warning: unsigned int format, different type arg (arg 3)
```

- This is OK, and can be safely ignored

What's Next

- This is still a prototype, but has been demonstrated to work with the Pixel alignments in the Geo model
- Has been in the release since 5.1.0, but has been evolving, along with StoreGate
- Plan to have it in stable form in 6.0.0

Other Stuff

- Full online documentation:
  - **http://annwm.lbl.gov/~leggett/Atlas/IOVSvc**
- Thanks for giving this talk Paolo, and sparing me having to get up and be coherent at 3AM!