Gaudi Introspection: a possible starting point for the LCG-Data-Dictionary

Alain Bazan
CERN/ATLAS
LAPP

Thierry Bouedo
CERN/ATLAS
LAPP

Pere Mato
CERN/EP-LBC

Stefan Roiser
CERN/EP-LBC
TU Vienna

Craig Tull
CERN/ATLAS
NERSC/LBNL

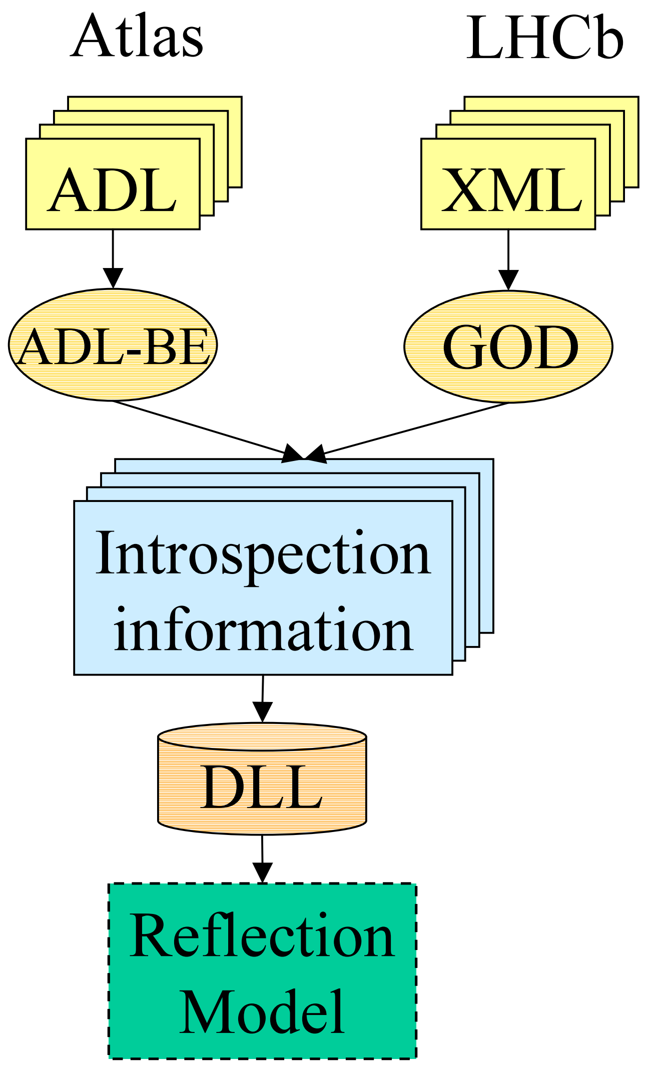
Content

- *General information*
- *Atlas & LHCb*
- *MetaModel*
- *Some examples*
- *Additional information*

General

- *Almost independent from Gaudi-framework*
  - *framework only needed for loading of libraries*
  - *self-contained model*
- *borrowed from Java-Reflection-API*
  - *robust and complete model*
  - *easy to handle and intuitive*
  - *well documented*

Atlas & LHCb

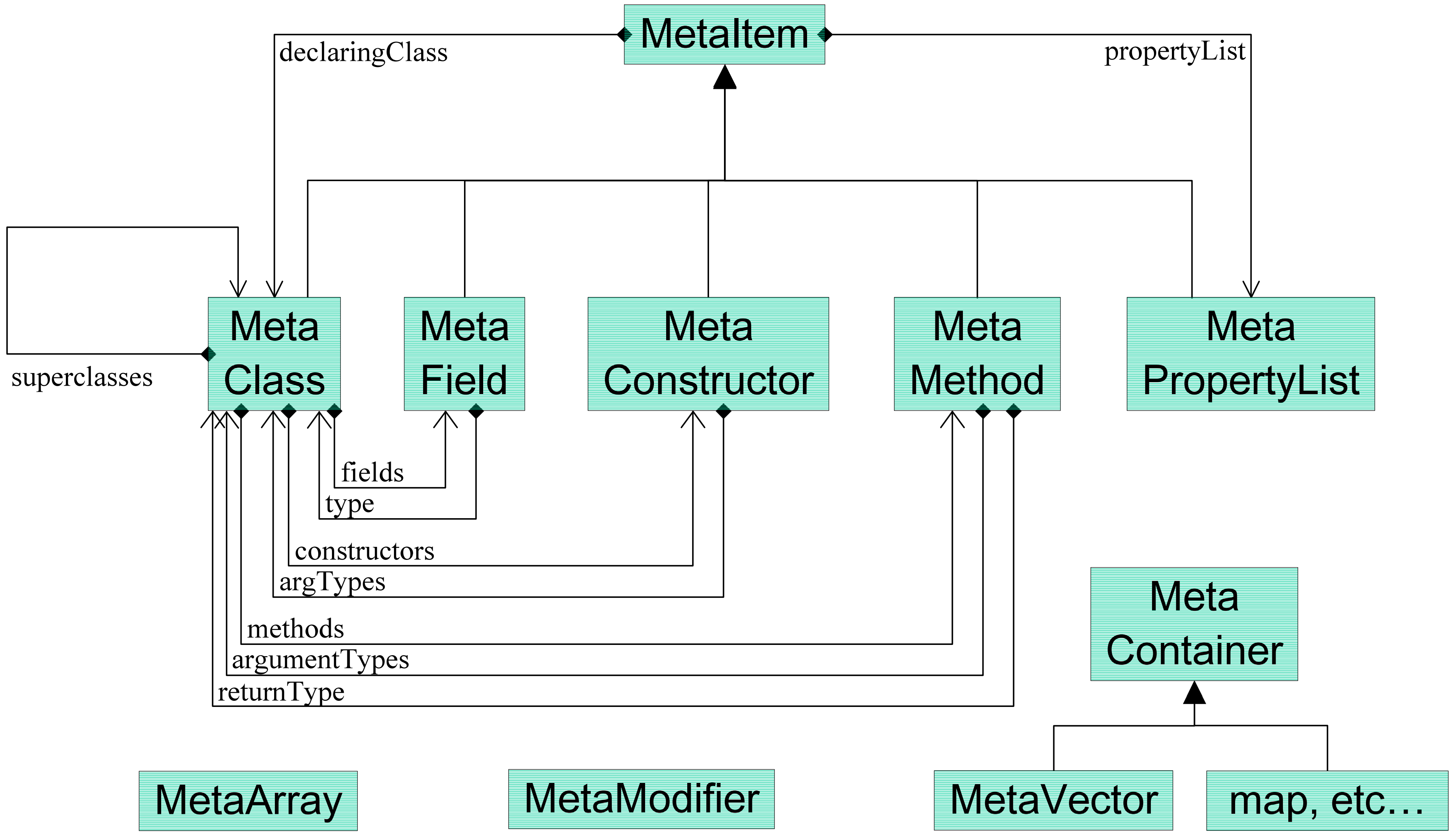


- *Common effort*
- *Automatic production of dictionary-information*
  - *by hand also possible*
- *Different additional info*
  - *classID, author*
  - *ADL-information*
- *Points to discuss*
  - *access to private data-members*

C++ concepts supported

- *Classes*
- *Inheritance*
  - *walk tree*
- *Constructors*
  - *create instances*
- *Methods*
  - *invocation*
- *Members*
  - *get/set values*
- *Pointers*
- *Limited template functionality*

The Model


MetaItem
declaringClass
propertyList
superclasses
Meta Class
Meta Field
Meta Constructor
Meta Method
Meta PropertyList
fields
type
constructors
argTypes
methods
argumentTypes
returnType
Meta Container
MetaArray
MetaModifier
MetaVector
map, etc…

Meta-Classes

Name	Description	Functions
MetaItem	*holds information common to all classes*	*name, description, declaringClass, propertyList*
MetaClass	*basic entity, distinguish pub/priv*	*fields, methods, constructors, forName, superClass*
MetaField	*info about members set and get values*	*get, set, type, offset*
MetaMethod	*info about methods invoke methods*	*invoke, returnType, argumentTypes,*

Meta-Classes cont'd

Name	Description	Functions
`MetaConstructor`	`create new instances`	`argumentTypes, instantiate`
`MetaModifier`	`static functions, check modifiers`	`isPrivate, isConst, isProtected, etc…`
`MetaArray`	`get, set values`	`get, set`
`MetaContainer`	`MetaVector, MetaList, etc…`	`size, set, get, …`
`MetaPropertyList`	`additional info, that doesn't fit in model`	`getProperty, getProperties`

How to fill the model

```
class MCParticle_dict {
public: MCParticle_dict(); }
```

```
static MCParticle_dict instance;
```

```
MCParticle_dict::MCParticle() {
MetaClass* metaC = new MetaClass(“MCParticle”,
   “The Monte Carlo particle kinematics information”,
   0);
metaC->addField(“helicity”,
   “double”,
   “Helicity”,
   &((MCParticle)0)->m_helicity,
   MetaModifier::setPrivate());
}
```

How to use the model

```
void* baseOfClass = new MCParticle;
MetaClass* mc = MetaClass::forName("MCParticle");
```

```
std::vector<MetaField*> mf = mc->fields();
std::cout << mf[0]->name()                    // 'helicity'
  << mf[0]->type()->name()                    // 'double'
  << mf[0]->declaringClass()->name()          // 'MCParticle'
  << mf[0]->get(baseOfClass,double());        //  some value
```

```
std::vector<MetaMethod*> mm = mc->methods();
std::cout  << mm[0]->name()                   // 'helicity'
  << mm[0]->returnType()->name()              // 'double'
  << mm->invoke(baseOfClass, double());       //  some value
```

```
MetaPropertyList* mp = mc->propertyList();
std::cout << mp->getProperty("ClassID"); // '210'
```

Use cases

- *Serialization of objects*
- *Data object description*
  - *produce dictionary-information with reflection-info*
- *Event data store browser*
  - *browse transient event store*
- *Interactive python-interface*

Improvements

- *access to private data members for foreign classes*
  - *for the time being '#define public private'*
- *namespace*
  - *should be easy to implement*
- *templates*
- *split into read/write interface*

Additional information

- *Data Dictionaries web pages*
  - *http://cern.ch/lhcb-comp/Frameworks/DataDictionary*
  - *http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/architecture/DataDictionary*
- *afs-area*
  - GaudiIntrospection
    - */afs/cern.ch/sw/Gaudi/releases/GaudiIntrospection*
  - Examples (Event-packages)
    - */afs/cern.ch/lhcb/software/NEW/Event*

Summary

- *The model*
  - *is well designed*
  - *is self-contained*
  - *is complete (except templates and namespaces)*
  - *is intuitive*
  - *is easy to feed*
  - *is familiar to people knowing the Java Refletion API*

- *Dictionary is more than persistency*
  - *e.g. event data store browser*
- *Common effort of Atlas and LHCb*
  - *agreement on this model*