GSI
Online
Offline
Object
Oriented

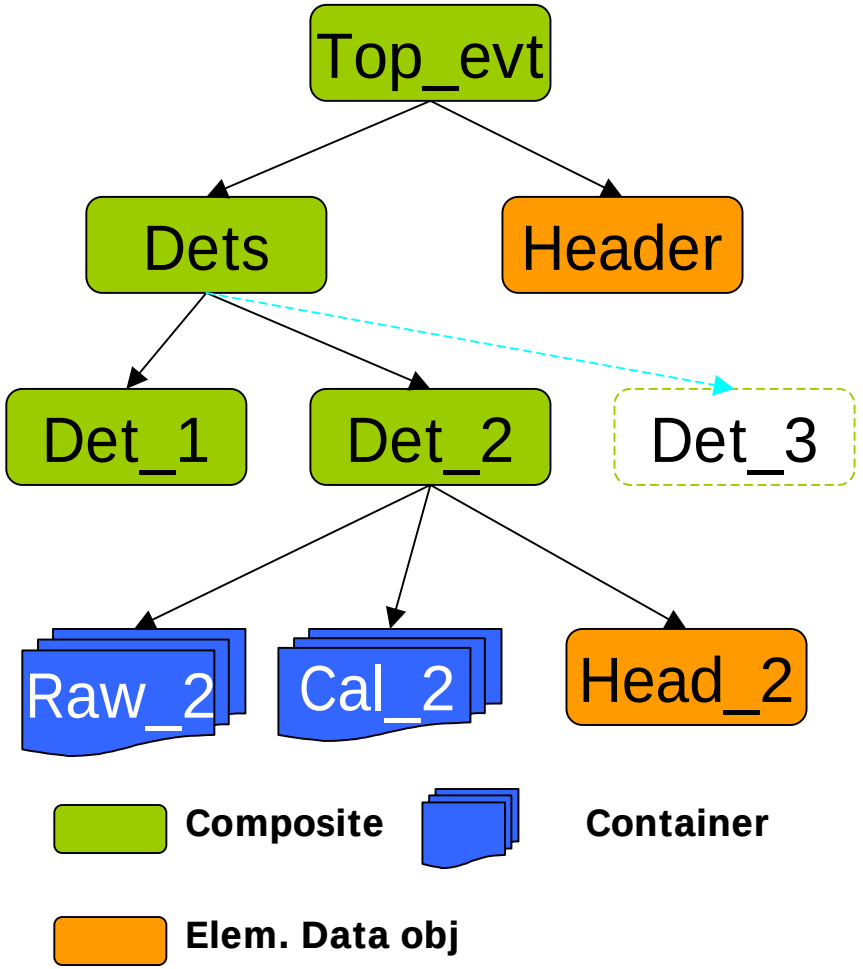
The Go4 Composite Event

J. Adamczewski, M. Al-Turany, D. Bertini, H.G.Essel, S.Linev

ROOT 2002

The Event Structure

- Structural organization of data record after primary interaction.
- Should represents the natural experimental setup (geometry, detectors, …)
- Fast direct access of components (indexation needed)
- Full or Partial IO: need to map the event object into a ROOT TTree
- General design to fit to different needs



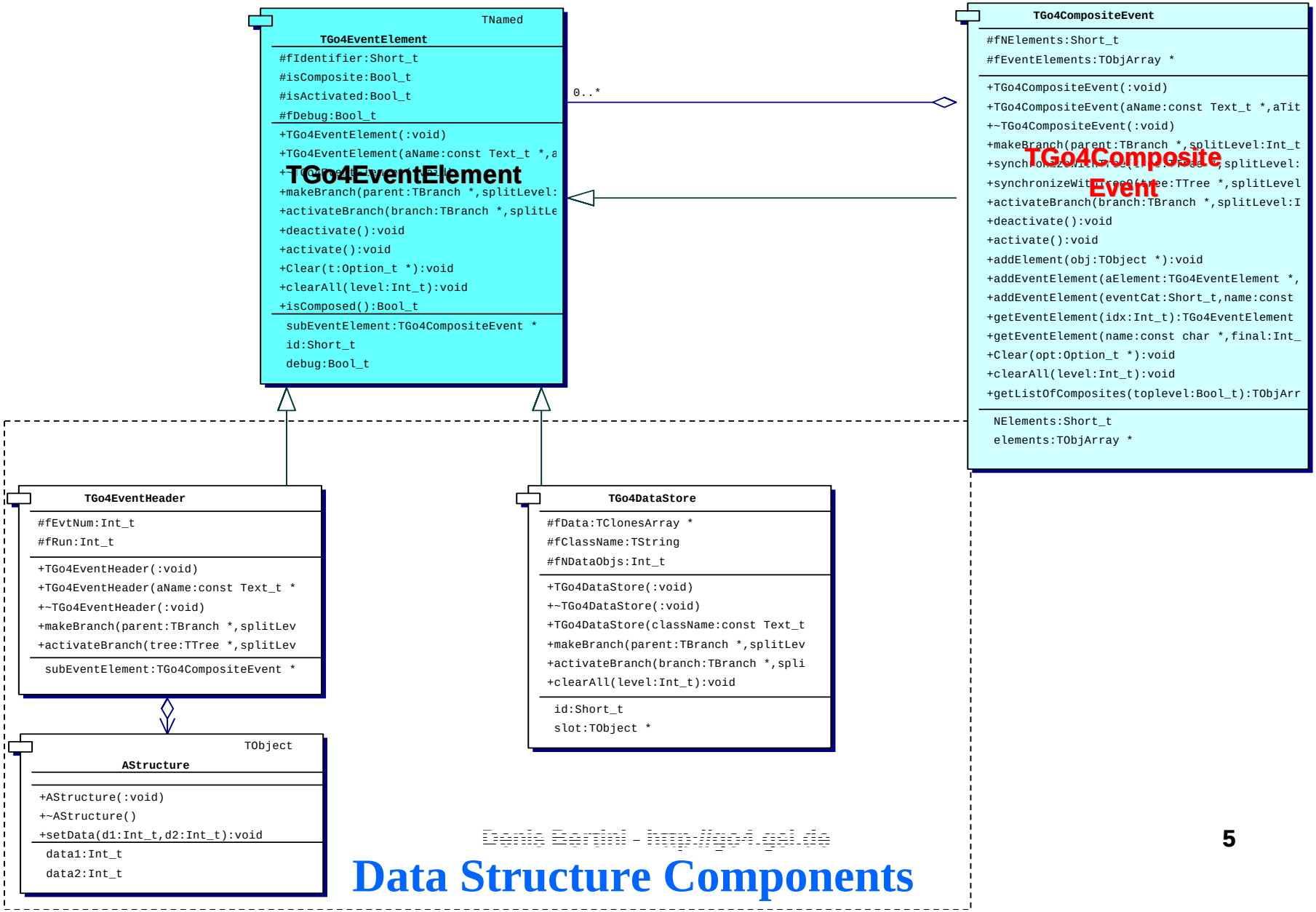
Goals

- Let user build complex Event Structure out of simple Data Object components
- Treats single Data Objects and composition of Data Objects uniformly
- Easy data retrieval from file

Requirements

- Use pure ROOT class for IO mechanism
  - No changes in TTree or TBranchElement
- Recursive method to synchronize the composite structure with TTree
  - To retrieve user Data, no need for :
    - TTree::MakeCode()
    - TTree::MakeClass()

OO Composite Model



Synchronizing With TTree

Use of recursive algorithms

(composite OO model)

- Create branches in TTree:

TGo4EventElement::makeBranch()

TGo4CompositeEvent::makeBranch()

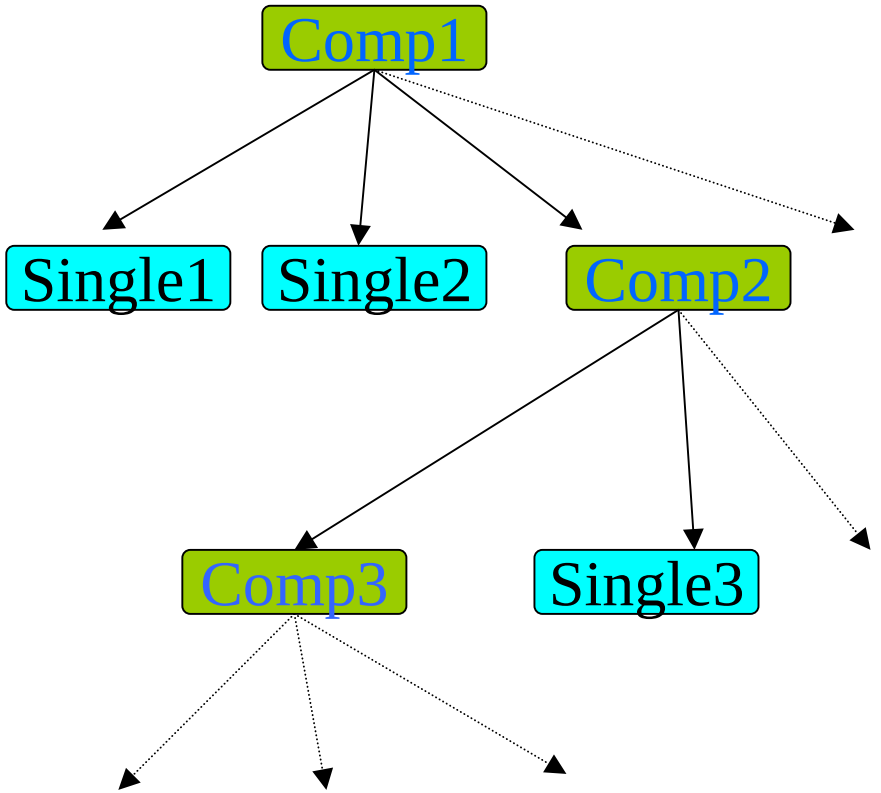
- Synchronize objects with TTree:

TGo4CompositeEvent::Synchronize(TTree*T)

- Partial IO:

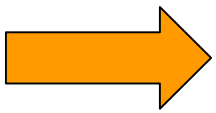
TGo4CompositeEvent::activate();

TGo4CompositeEvent::deactivate();

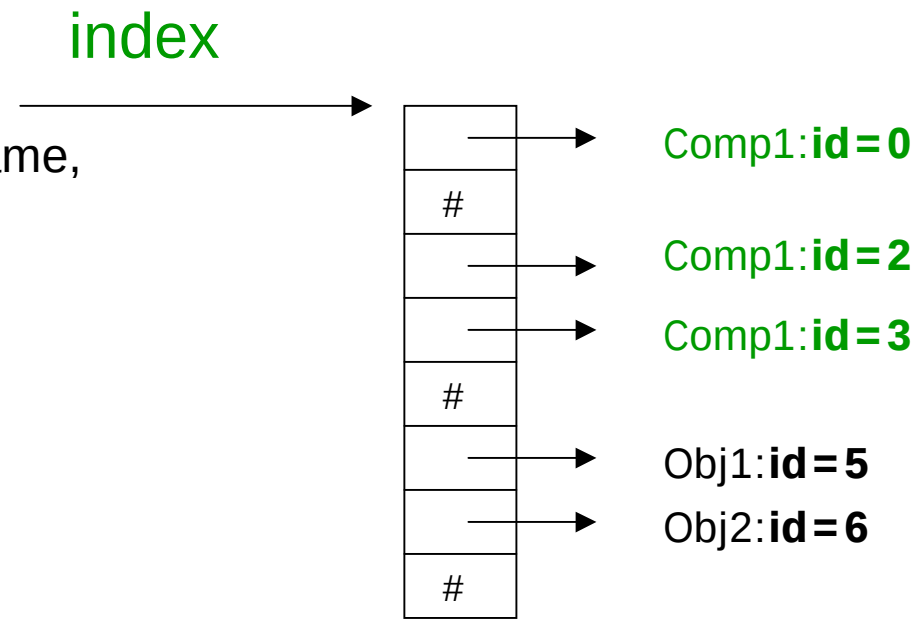


Creating Components

Class **TGo4CompositeEvent**: public **TGo4EventElement**{
private:
TObjArray *components;
public:
TGo4CompositeEvent(const char* name,
const char*title, **Int_t Id**);
…
};



unique identifier :Id per components in ctor

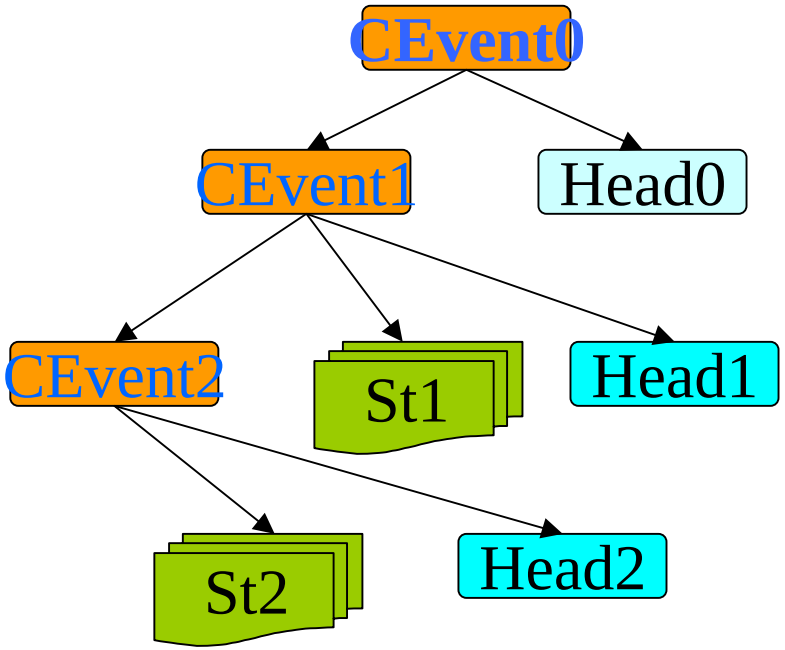


Creating Components

```
{ //user interface example
TTree *tree = new TTree("T","test");
// create some composites
TGo4CompositeEvent* event0,*event1,*event2;
CEvent0= new TGo4CompositeEvent("cEvent0","top",id1);
CEvent1= ...
// create some containers for data storage
TGo4ClonesElement* st0,*st1,*st2;
st0 = new TGo4ClonesElement("Htrack",100,"Det1",idc1);
...
// create some headers
TGo4EventHeader *head0,*head1,*head2;
head0 = new TGo4EventHeader("header0","header for det0",idh1);
...
}
```

Modeling an Event

```
{//Event structure modeling
// 2th composite
  CEvent2->addElement(head2);
  CEvent2->addElement(st2);
//first composite
  CEvent1->addElement(head1);
  CEvent1->addElement(st1);
  CEvent1->addElement(CEvent2);
// add in top composite event
  CEvent0->addElement(head0);
  CEvent0->addElement(CEvent1);
// set top level branch
  TBranch*b = T->Branch("top",
                ”CompositeEvent”,&CEvent0);
//compose tree structure
  CEvent0->makeBranch(b,split);
}
```

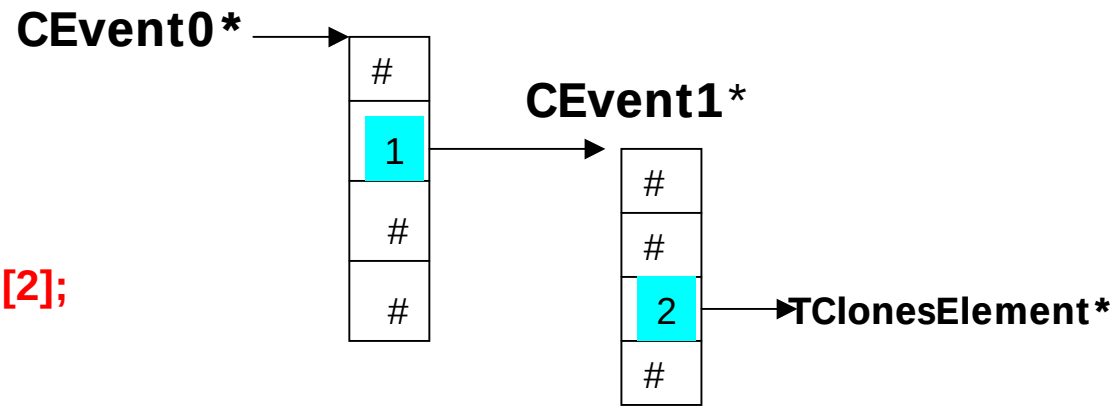


Data Retrieval (1)

```
{
    TFile *f = new TFile("combo.root");
    TTree *tree= (TTree*) f->Get("T");
// create an instance of composite event
    TGo4CompositeEvent* cEvent0 = new TGo4CompositeEvent()
// Use of a recursive algorithm to recreate
// the composite event in memory from the
// branches in TTree
    CEvent0->synchronizeWithTree(T,split);
//Retrieving sub-elements
    TGo4CompositeEvent* c1 = CEvent0->getElement("cEvent1");
    TGo4Element *e2 = c1->getElement("head1");
    CEvent0->getElement("cEvent2")->deactivate();
// then do event loop…
}
```

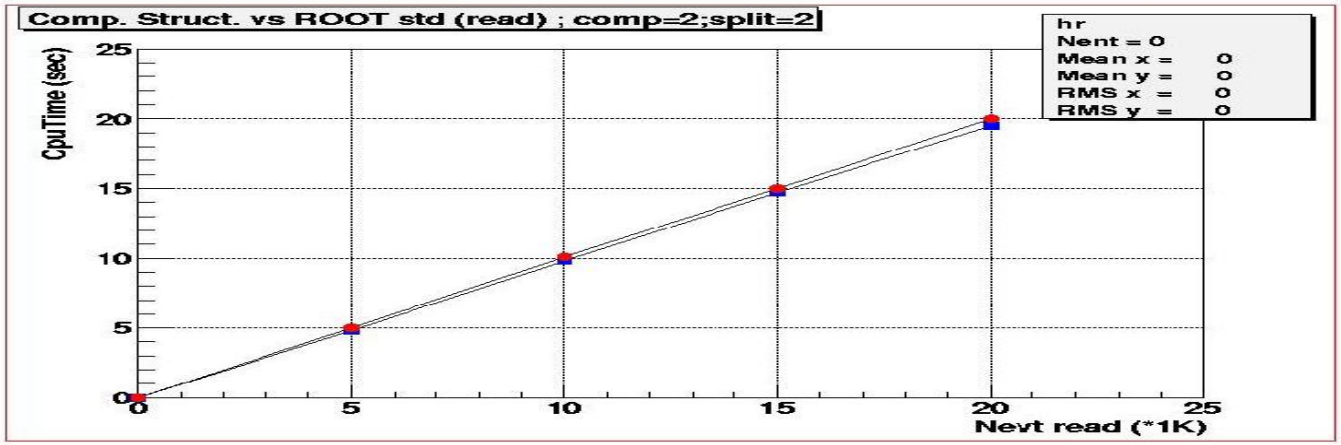
Data Retrieval (2)

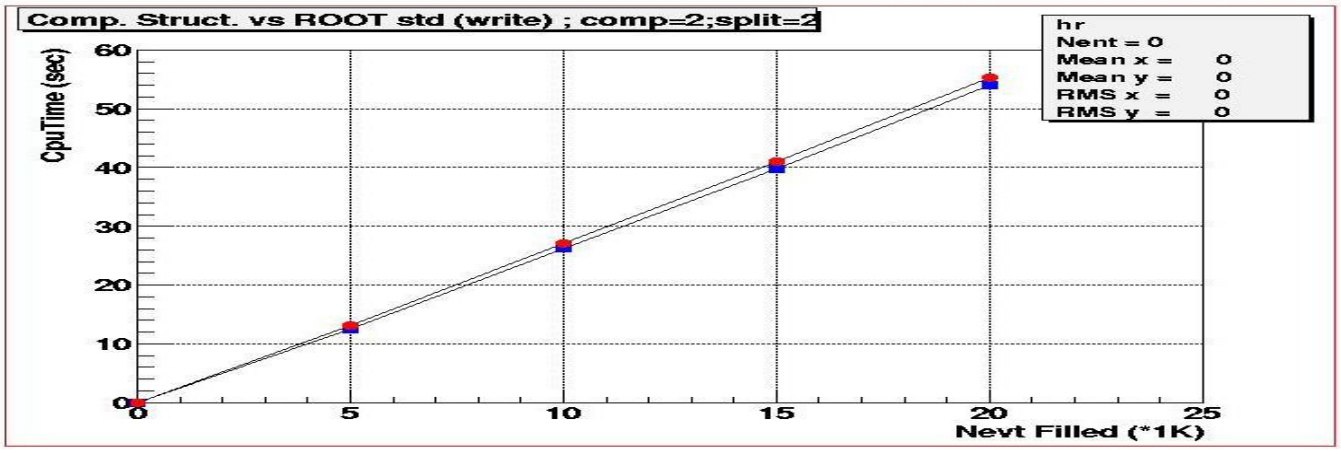
```
{
//Retrieving component by name
TGo4CompositeEvent* c1 = CEvent0->getElement("cEvent1");
TGo4Element *e2 = c1->getElement("head1");
//deactive IO for this component
CEvent0->getElement("cEvent2")->deactivate();
```

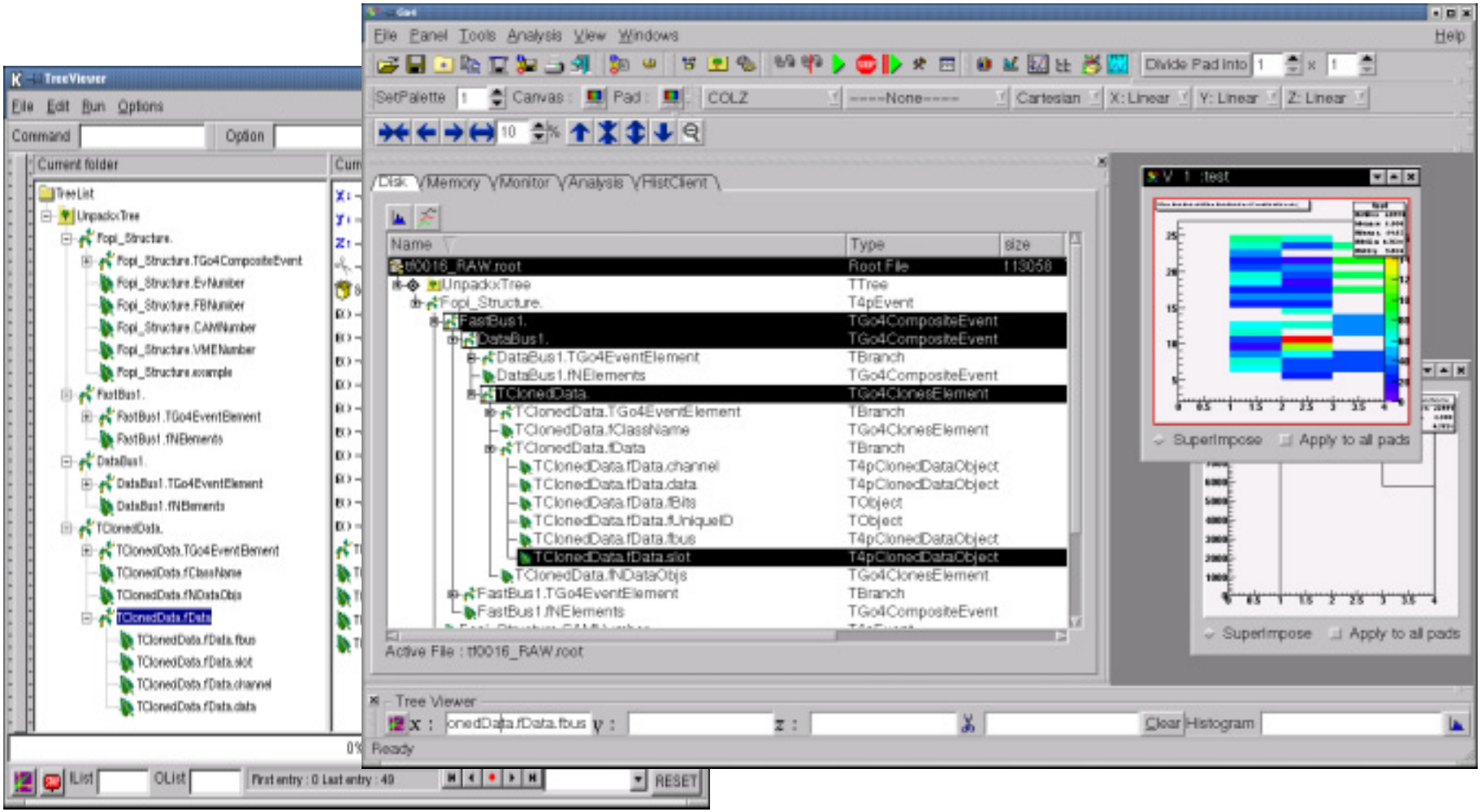



```
//Direct access by indexes
TClonesElement* cl =
(TClonesElement*) &(*CEvent0)[1][2];
}
```

Benchmarks


Comp. Struct. vs ROOT std (read) ; comp=2;split=2
CpuTime (sec)
Nevt read (*1K)
hr
Nent = 0
Mean x = 0
Mean y = 0
RMS x = 0
RMS y = 0


Comp. Struct. vs ROOT std (write) ; comp=2;split=2
CpuTime (sec)
Nevt Filled (*1K)
hr
Nent = 0
Mean x = 0
Mean y = 0
RMS x = 0
RMS y = 0


TreeViewer
File Edit Run Options
Command
Option
Current folder
TreeList
UnpackxTree
Fopi_Structure.
FastBus1.
DataBus1.
TClonedData.
IList
OList
First entry : 0 Last entry : 49
RESET
File Panel Tools Analysis View Windows
Help
Divide Pad into
SetPalette
Canvas :
Pad :
COLZ
-----None-----
Cartesian
X: Linear
Y: Linear
Z: Linear
Disk
Memory
Monitor
Analysis
HistClient
Name
Type
size
tf0016_RAW.root
Root File
119058
UnpackxTree
TTree
Fopi_Structure.
T4pEvent
FastBus1.
TGo4CompositeEvent
DataBus1.
TGo4CompositeEvent
DataBus1.TGo4EventElement
TBranch
DataBus1.fNElements
TGo4CompositeEvent
TClonedData.
TGo4ClonesElement
TClonedData.TGo4EventElement
TBranch
TClonedData.fClassName
TGo4ClonesElement
TClonedData.fData
TBranch
TClonedData.fData.channel
T4pClonedDataObject
TClonedData.fData.data
T4pClonedDataObject
TClonedData.fData.fBits
TObject
TClonedData.fData.fUniqueID
TObject
TClonedData.fData.fbus
T4pClonedDataObject
TClonedData.fData.slot
T4pClonedDataObject
TClonedData.fNDataObjs
TGo4ClonesElement
FastBus1.TGo4EventElement
TBranch
FastBus1.fNElements
TGo4CompositeEvent
Active File : tf0016_RAW.root
V 1 :test
SuperImpose
Apply to all pads
Tree Viewer
x :
y :
z :
Clear
Histogram
Ready

Conclusion

- A composite model for event structure has been developed and tested
- No sub-classing of ROOT IO classes needed
- Easy retrieval of data via recursive algorithm
- No limitation in the composition of data objects