The GammaWare package

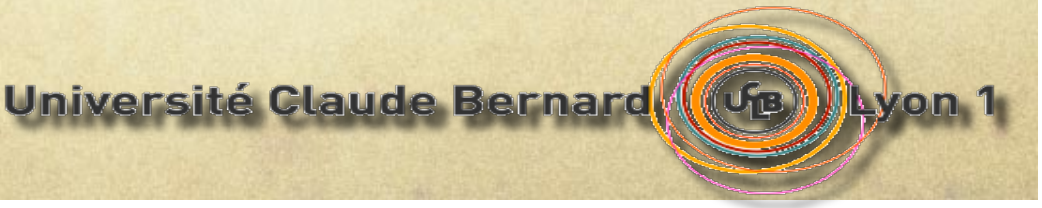
(including ADF)

Olivier Stézowski

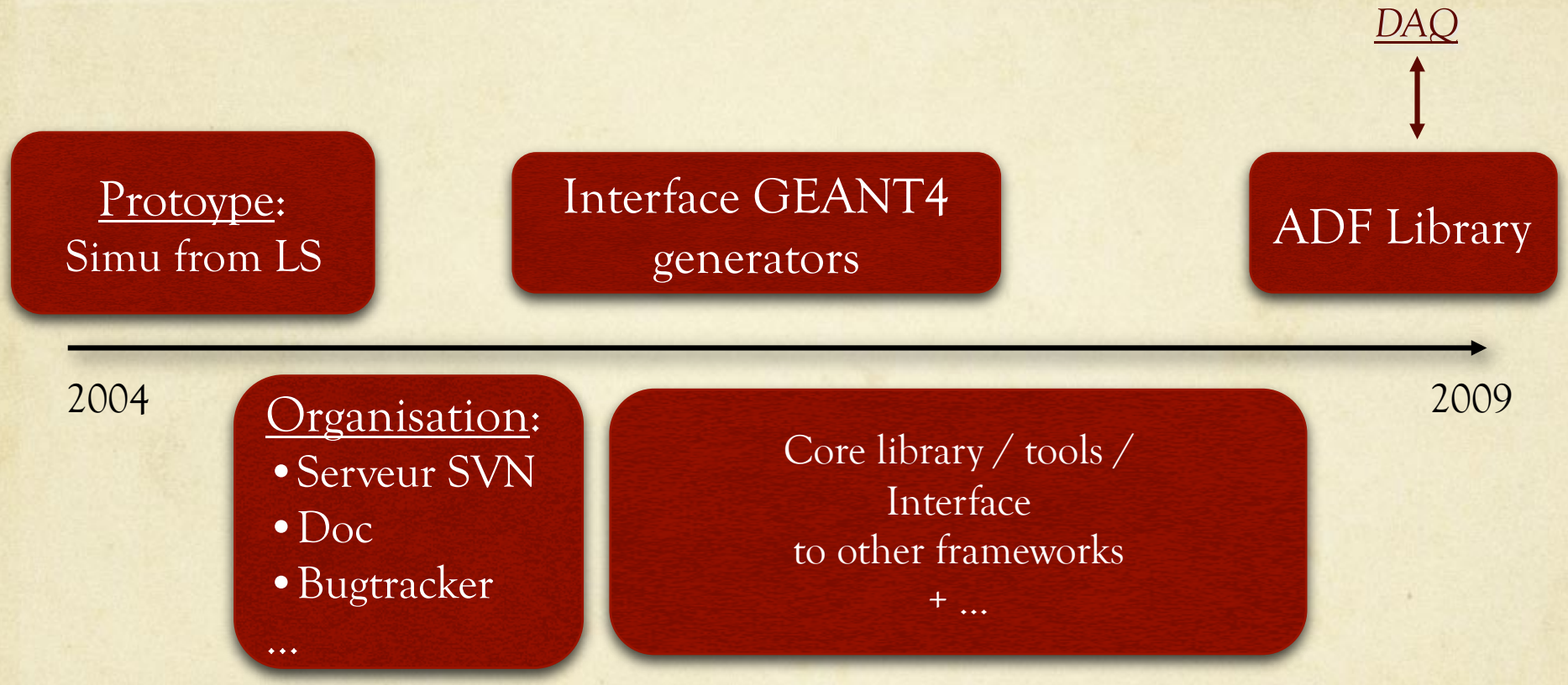
(on behalf of the Data Analysis Team)

Q-T Doan, E. Farnea, J. Ljungvall, A. Lopez-Martens, J. Nyberg, F. Recchia, B. Rossé, G.Suliman, Ch. Theisen, Ch Finck ...

AGATA week, Legnaro, January 20-22 2010



Evolution of the package



« Collaborative development » !!

Framework : collaborative work

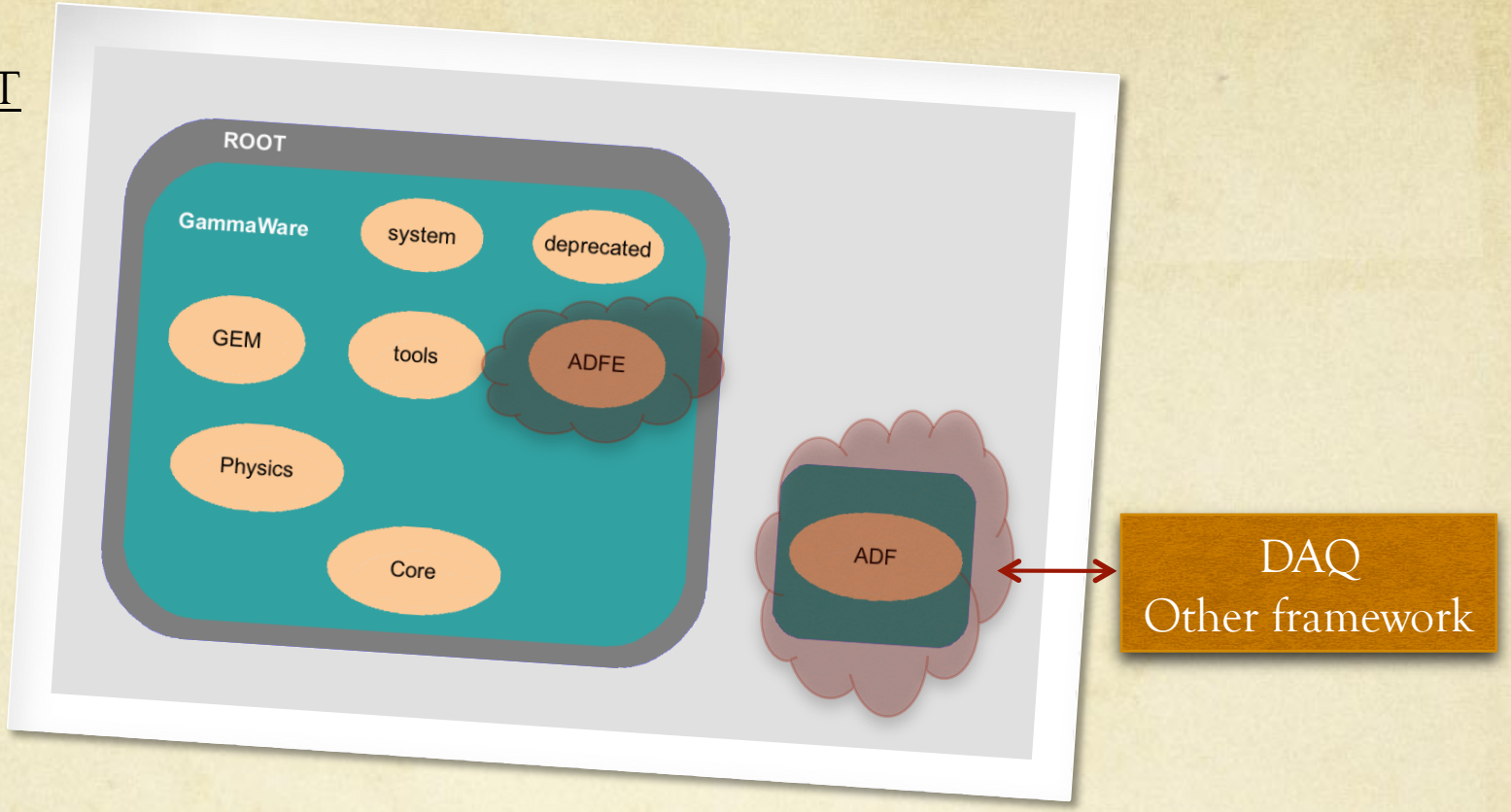
New(*)

- Work on the same files (subversion)
- Developers / users – tags, releases
- Documentation !
- Interaction (bug tracker, ticket system)
- Design proposals : design before writing
- …

➔ Not a huge success … so far …

GwUserGuide.(odt.pdf) - https://cvs.in2p3.fr/agata/docs/trunk/
(Agata, AgataSoftware)

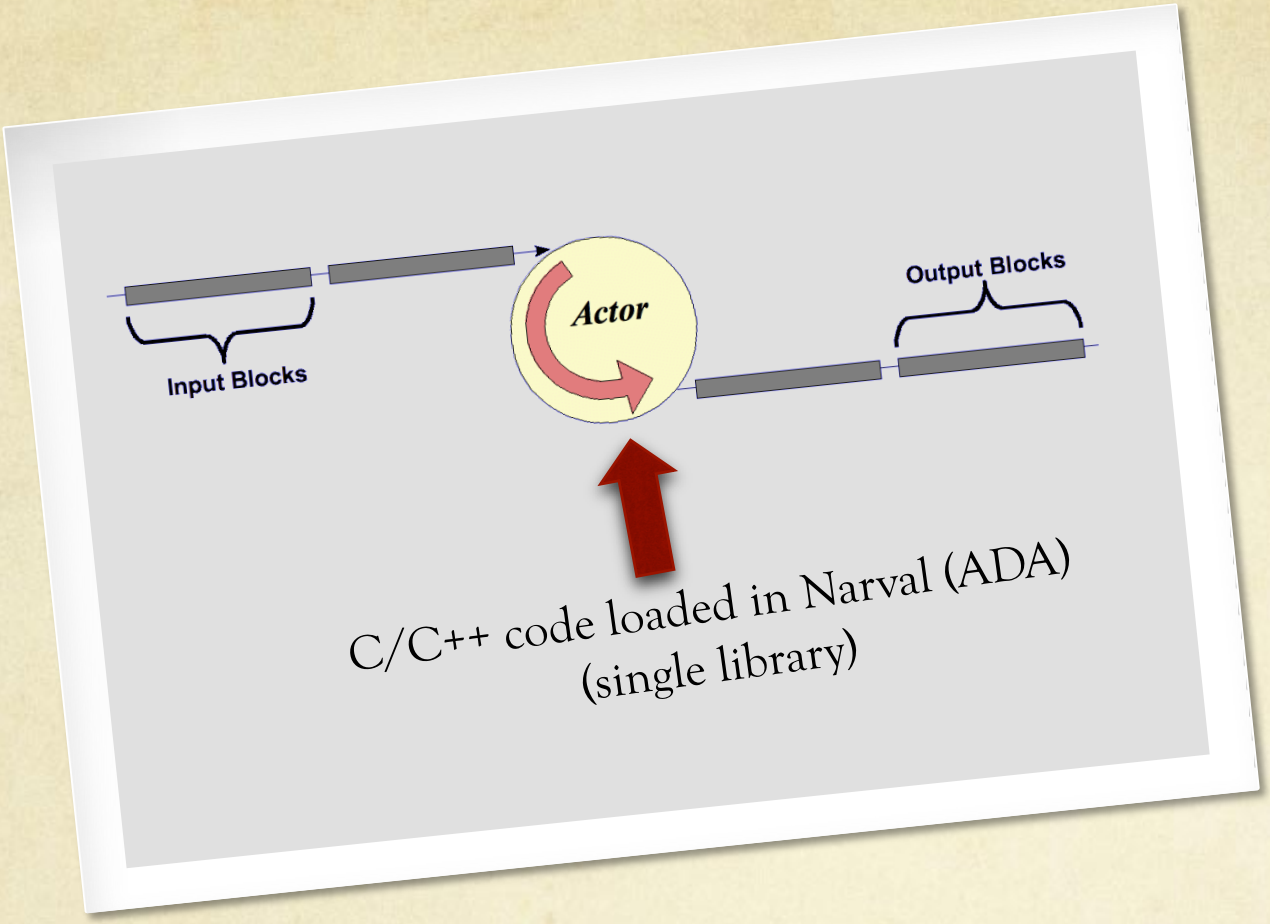
Add-ons for ROOT

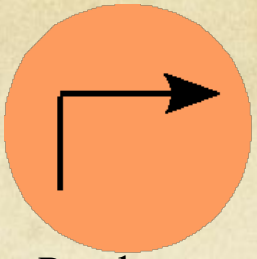


Design : avoid (circular) dependancies between libraries

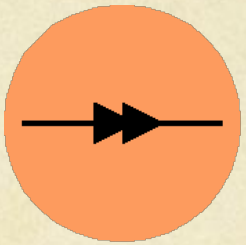
GammaWare$^{0.9}$ current situation

(PSA, Tracking, ancillaries not in but could be plugged)

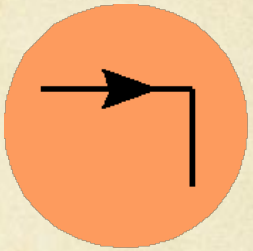




Producer



Filter



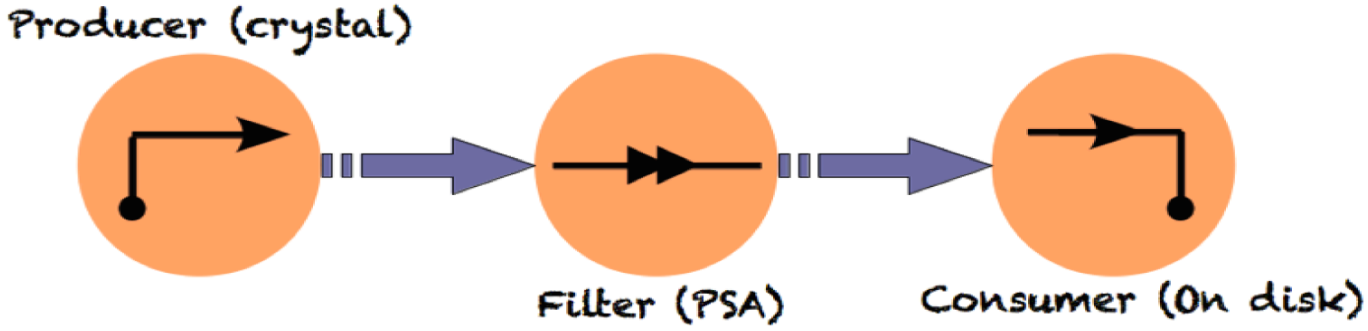
Consumer

« Narval terminology »

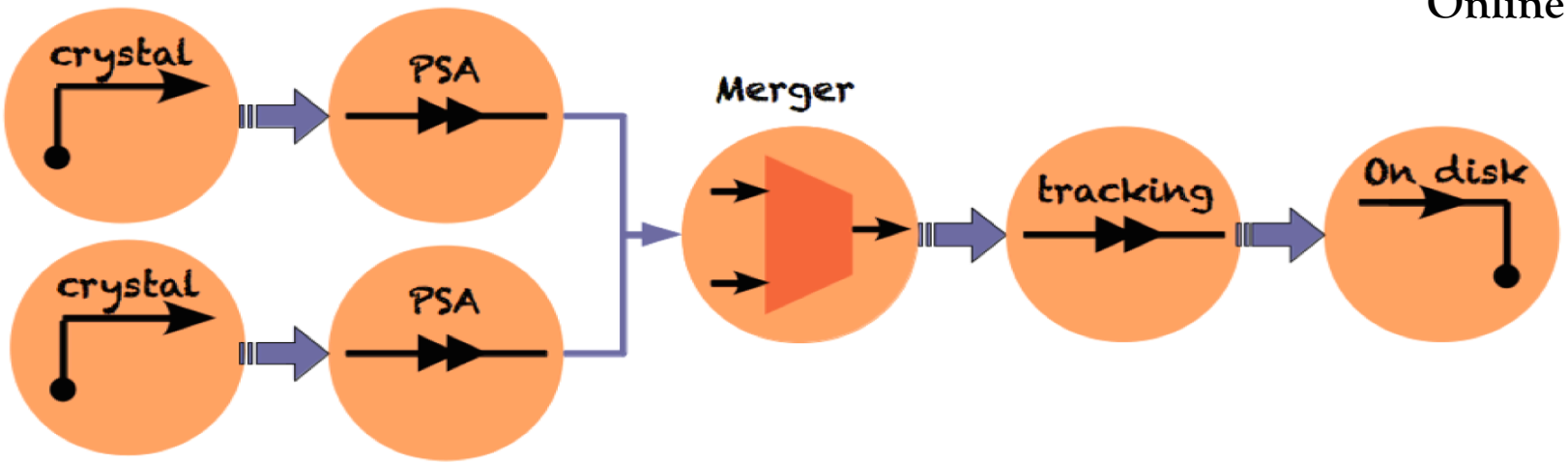
Distributed system : many computers connected by the network

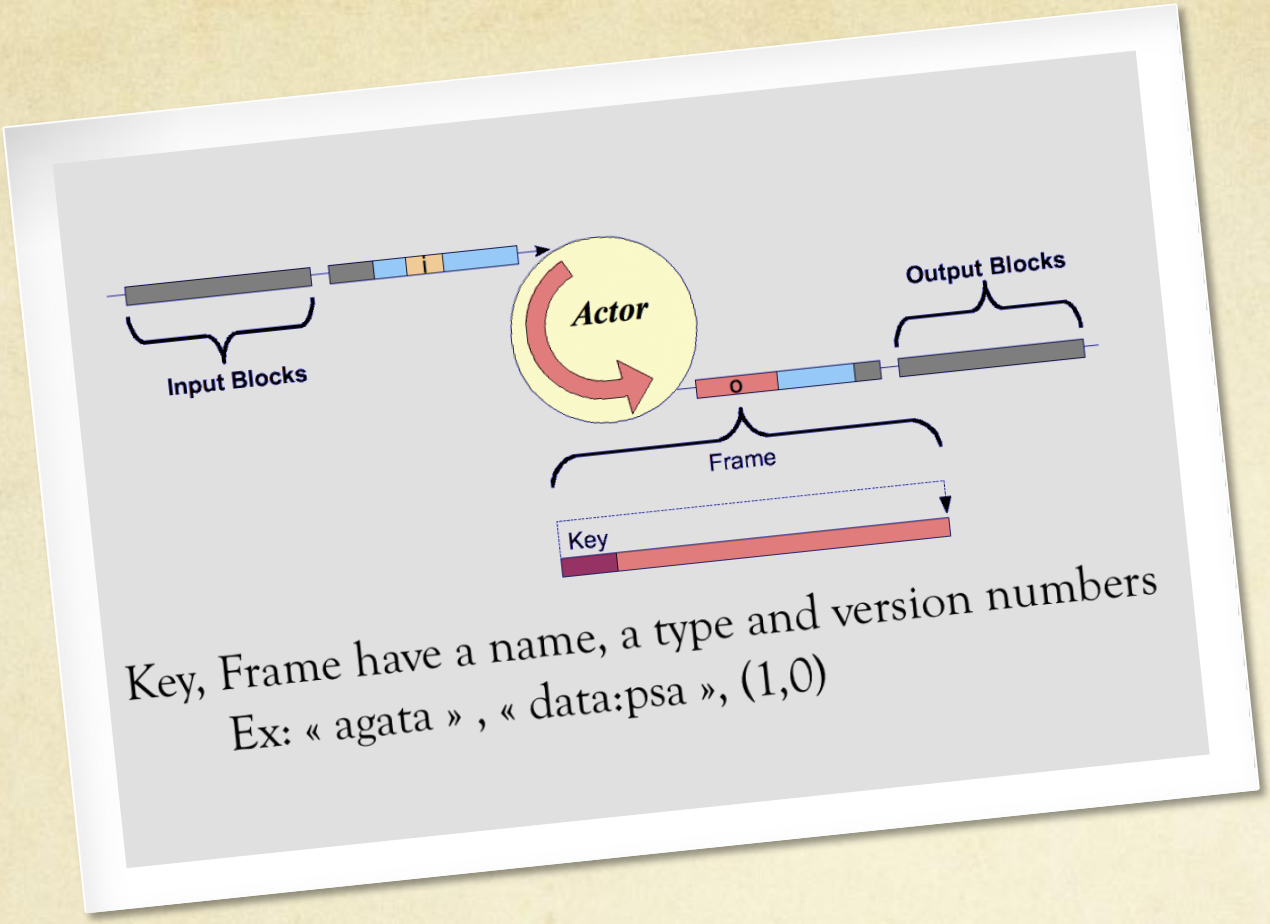
A topology : chain of analysis

Online / offline



Online





Strongly processed data flow ➔ likely to change over the years (version !!)

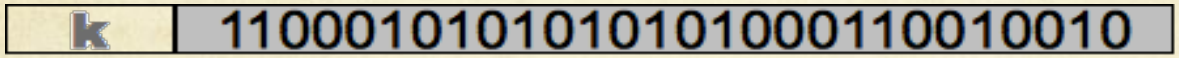
« ADF view » : Blocks of Frames

Agata Keys

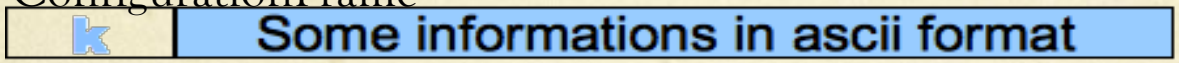
Key Version	Key content (what is really written in the data flow)				Key length
(0,0)	Frame Length UInt_t	Message type	Event Number	Timestamp	4 bytes
(1,0)	Frame Length UInt_t	Message type UInt_t	Event Number	Timestamp	8 bytes
(2,0)	Frame Length UInt_t	Message type UInt_t	Event Number UInt_t	Timestamp	12 bytes
(4,0)	Frame Length UInt_t	Message type UInt_t	Event Number UInt_t	Timestamp ULong64_t	20 bytes

Frames

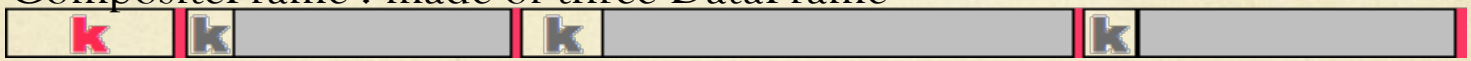
DataFrame



ConfigurationFrame



CompositeFrame : made of three DataFrame



An .adf data flow or file

Full definition of the DataFlow !!

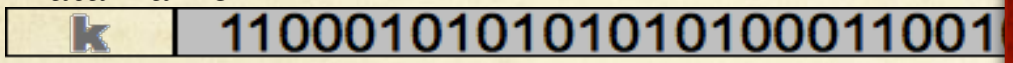
Agata Keys

Key Version	Key content (what is really written in the data flow)			
(0,0)	Frame Length UInt_t	Message type	Event Number	
(1,0)	Frame Length UInt_t	Message type UInt_t	Event Number	
(2,0)	Frame Length UInt_t	Message type UInt_t	Event Number UInt_t	
(4,0)	Frame Length UInt_t	Message type UInt_t	Event Number UInt_t	

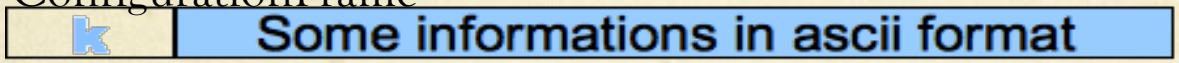
agata	0xFA000000
conf	0xFA000200
conf:global	0xFA000200
data	0xFA000100
data:crystal	0xFA010100
data:ccrystal	0xFA010111
data:psa	0xFA010102
event:data:psa	0xFA010103
event:data	0xFA010104
data:tracked	0xFA010105
data:ranc0	0xFA0101A0
meta	0xFA001100
meta:vertex	0xFA011100

Frames

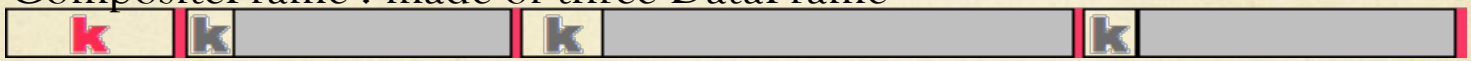
DataFrame



ConfigurationFrame



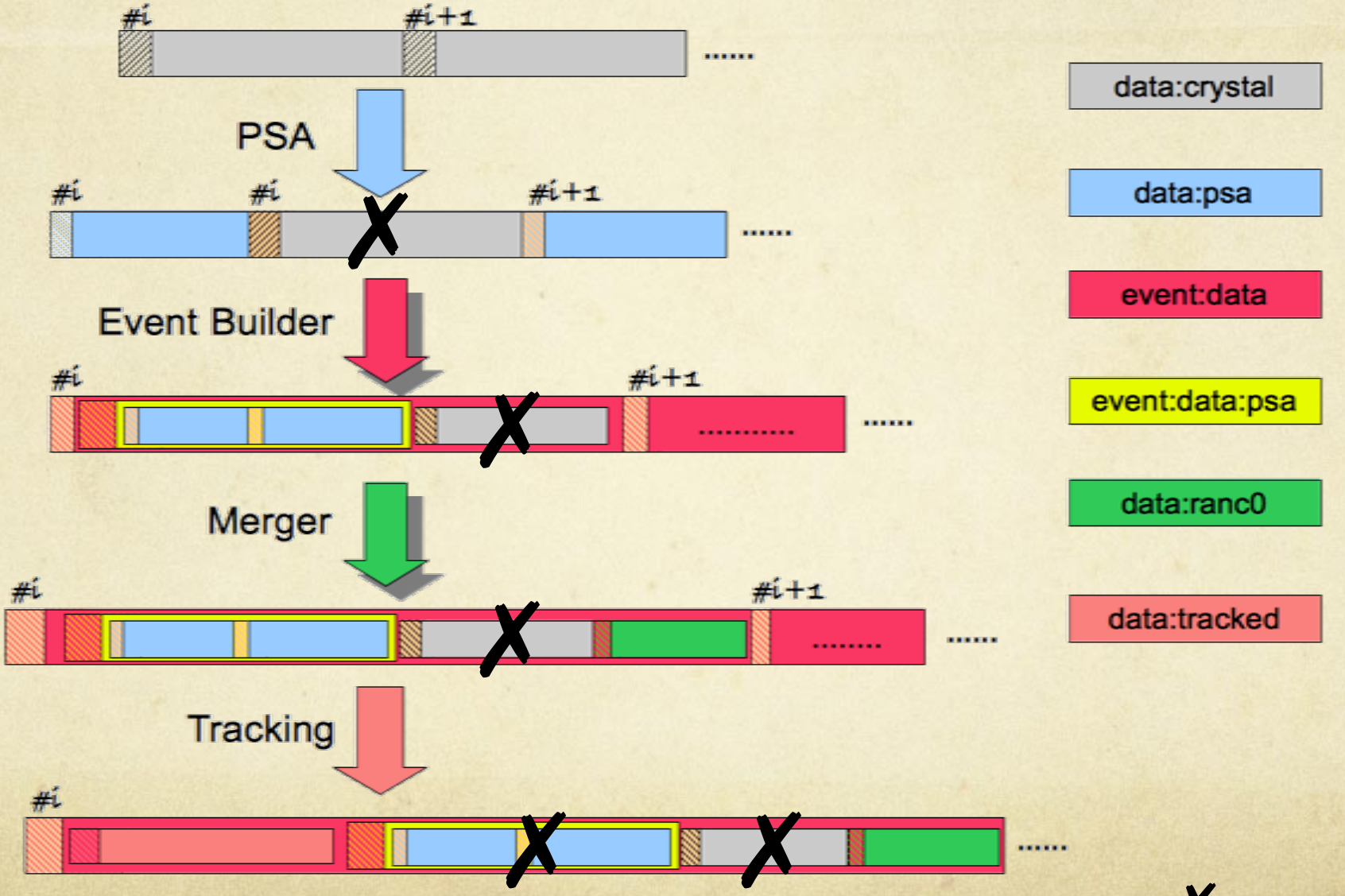
CompositeFrame : made of three DataFrame



An .adf data flow or file

Full definition of the DataFlow !!

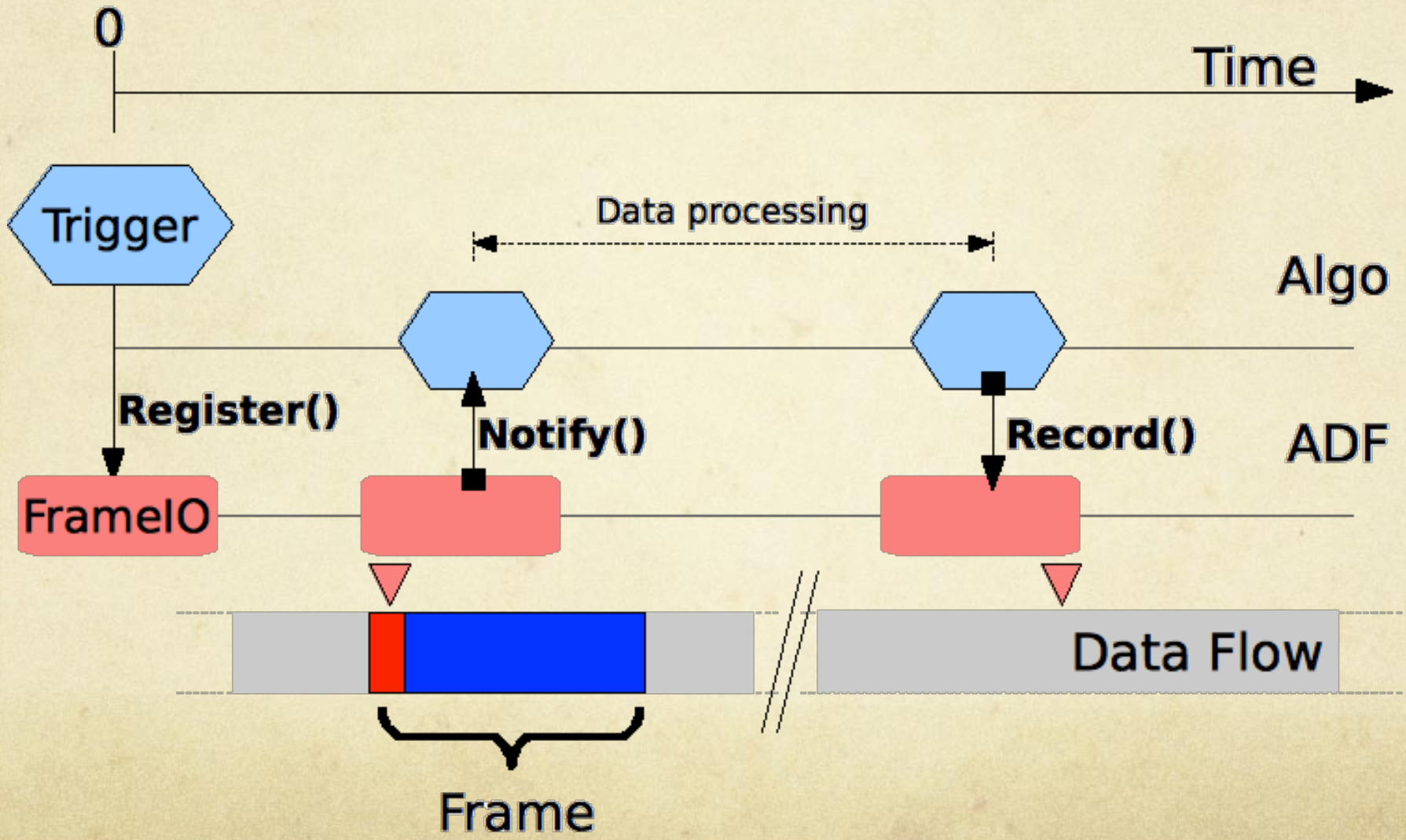
Agata Data Flow @ different stages



Actions on an Agata Data Flow

Complex Data Flow **BUT** an algorithm just needs the Frames it could processed !!

➔ DFTrigger (« list of Frames »)



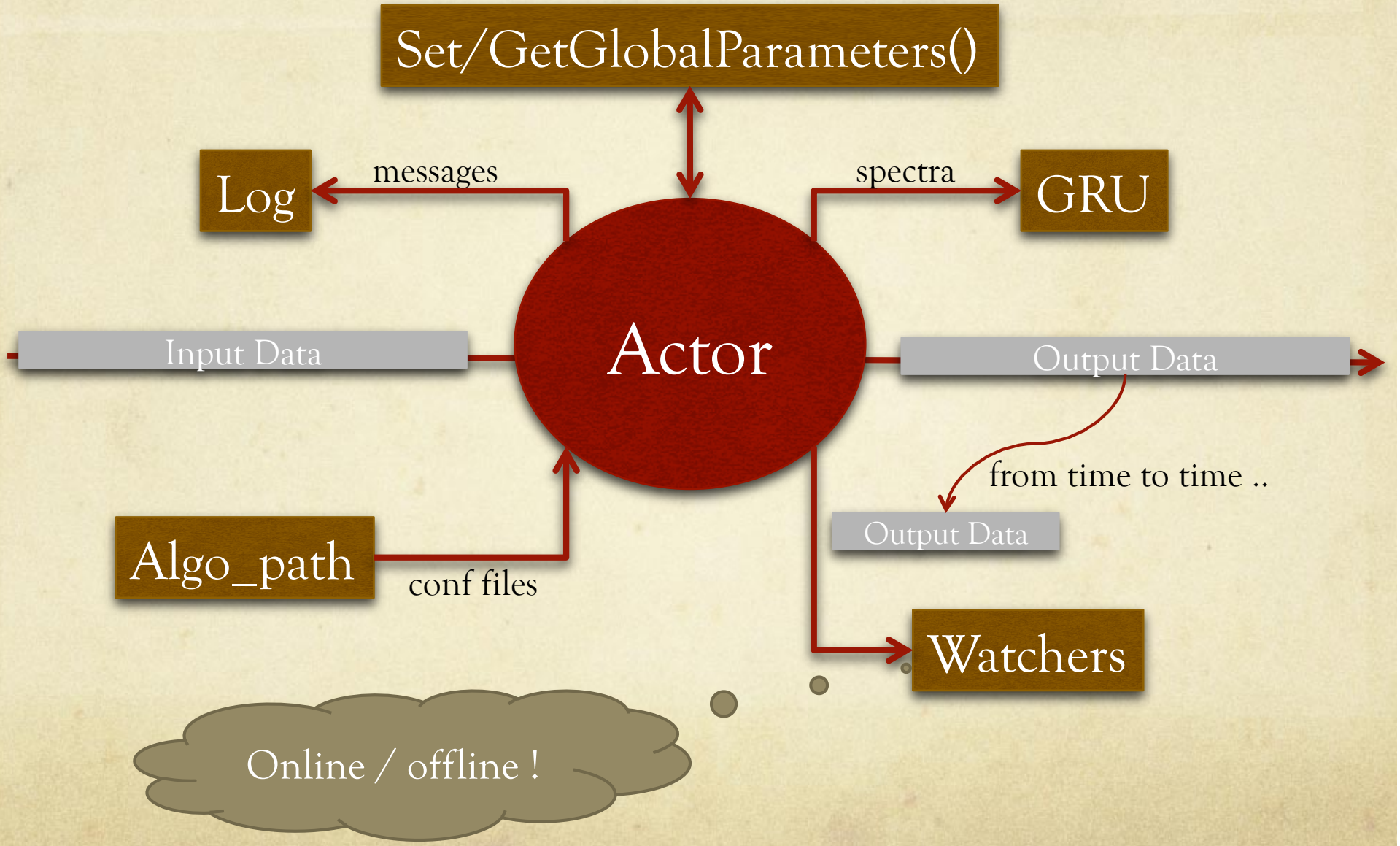
Actions on an Agata Data Flow

Definition of a DFTrigger

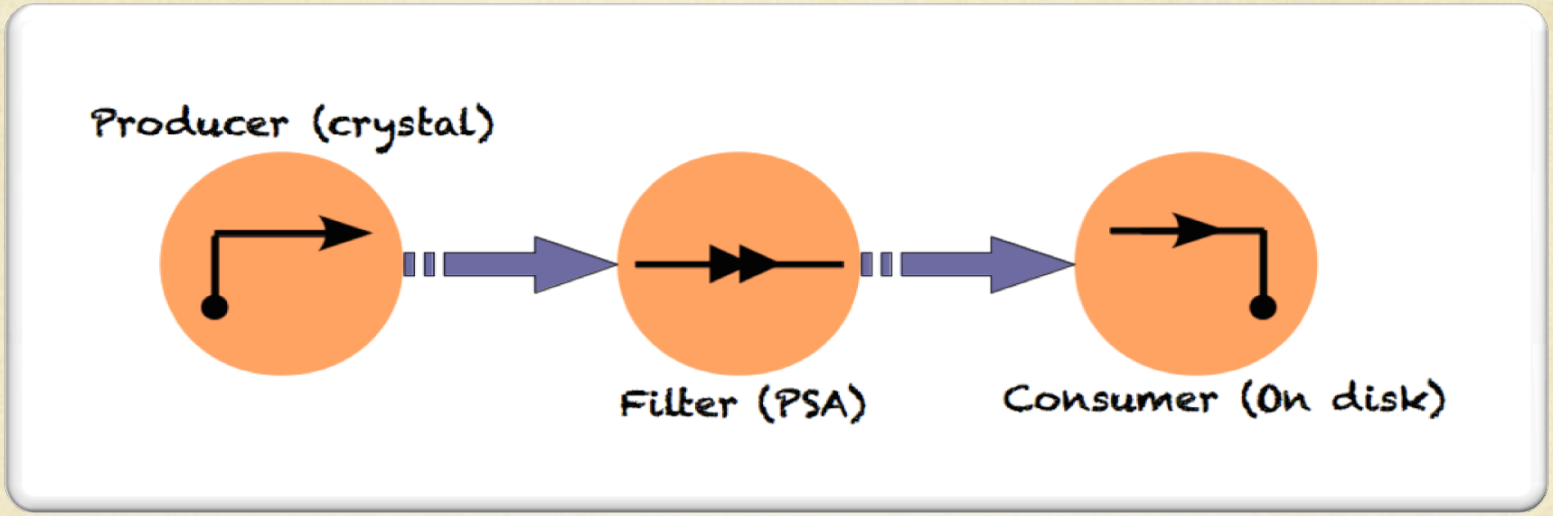

```
AgataFrameTrigger *t = new AgataFrameTrigger("OnTrackAndAncillary");

    t->Add("event:data" ..., IsConsumed , ... );
      t->Add("data:tracked", ..., IsConsumed , ... );
      t->Add("data:ranc0", ..., IsConsumed , ... );

    t->SetOutputFrame( ... );
```

event:data with inside a tracked and an ancillary frame

Communications Narval - ADF



Emulators



Simple topology could be emulated <u>without</u> Narval (see in gammaware) :

src/adf/dev/test0.1.c
src/adf/dev/test0.2.c
} c programs

demos/adf/Emulator.C ⟶ ROOT macros

\+ ... others (Joa)

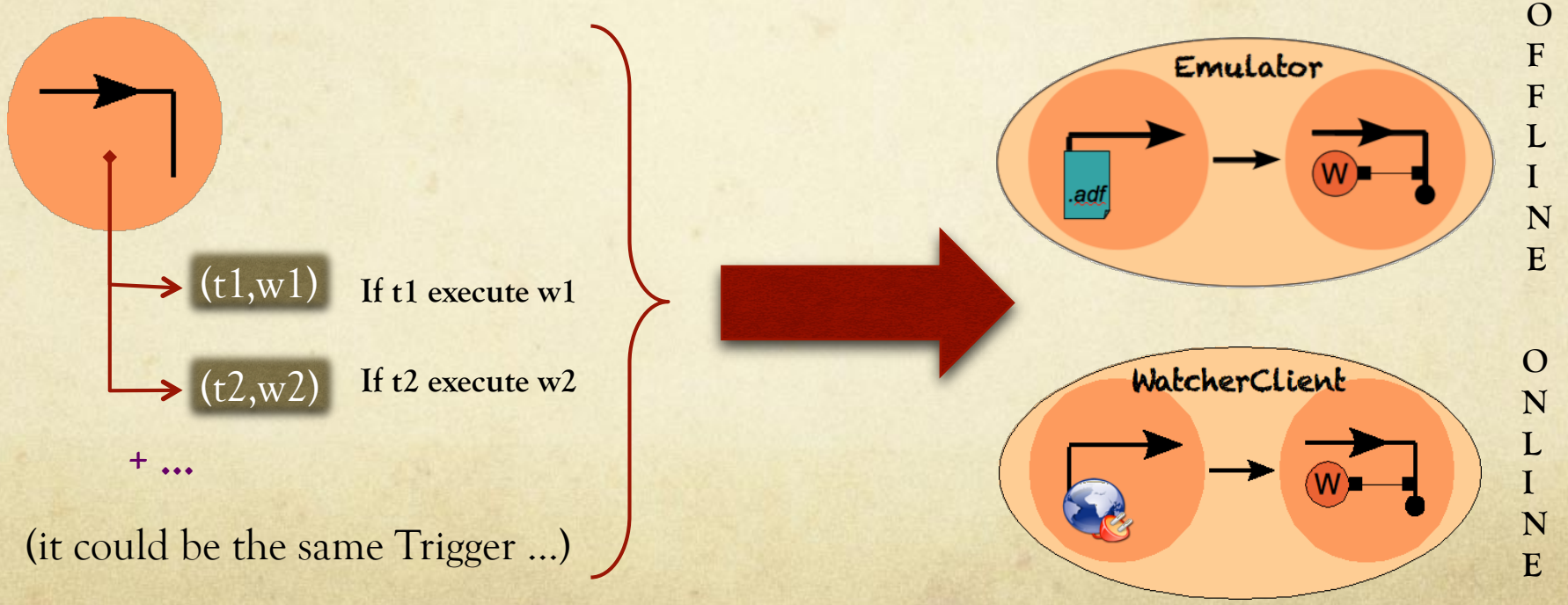
Watchers

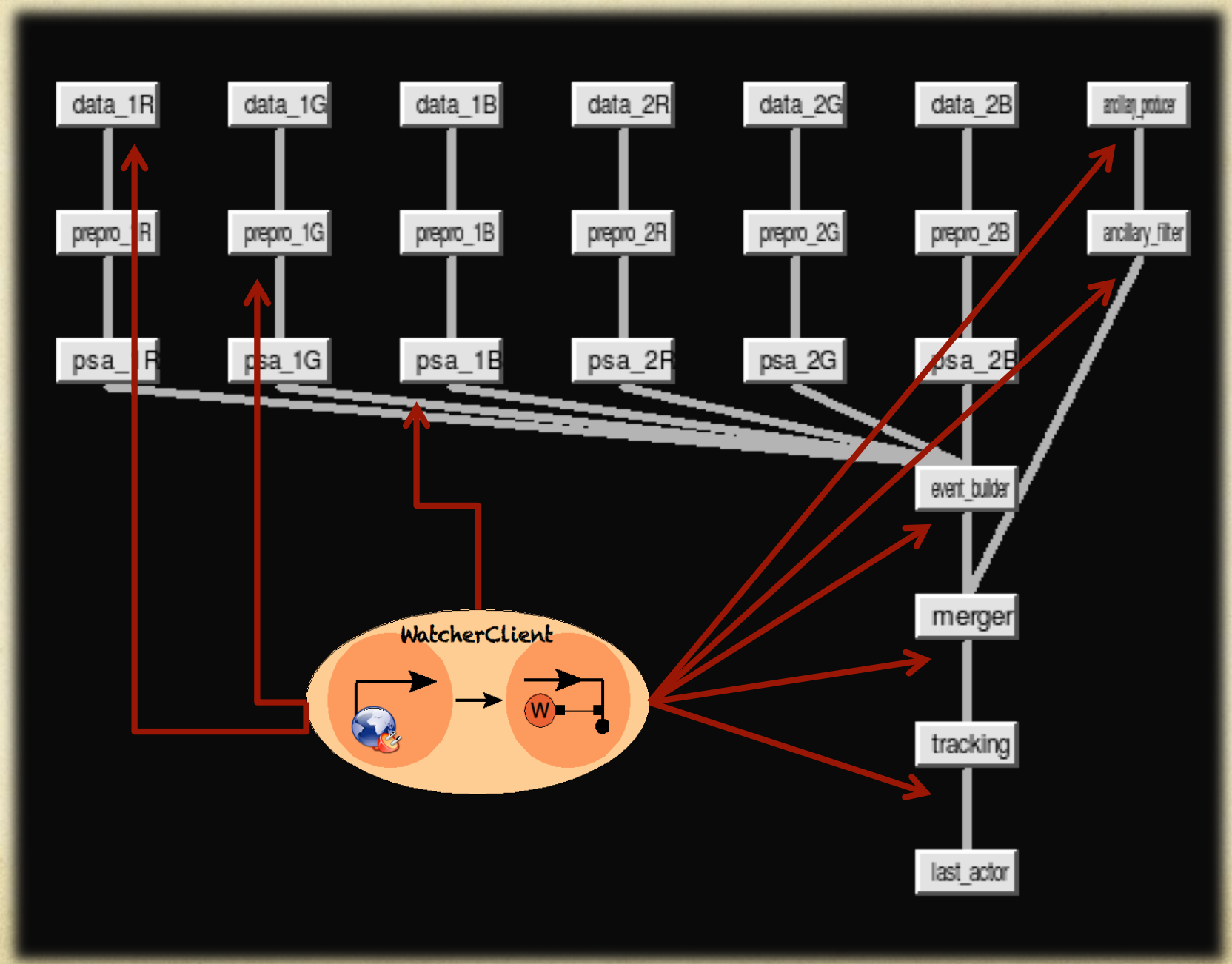
A Watcher is a task performed once a given condition (Trigger) is fullfilled

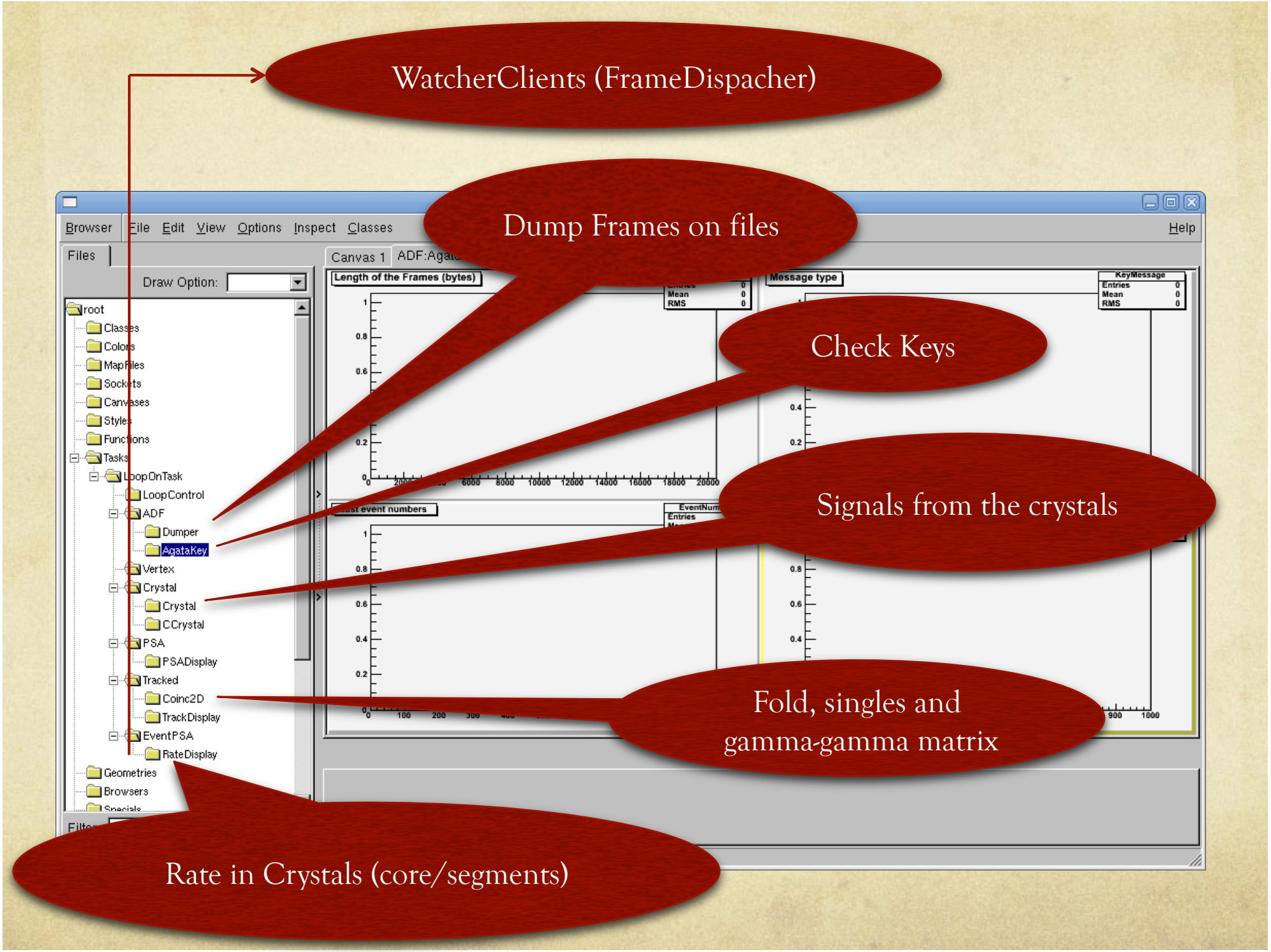
Ex :

RateDisplay watcher on event:data:psa shows counting rates in crystals
Coinc2D watcher on data:tracked makes (Fold, Singles, γ-γ matrix)

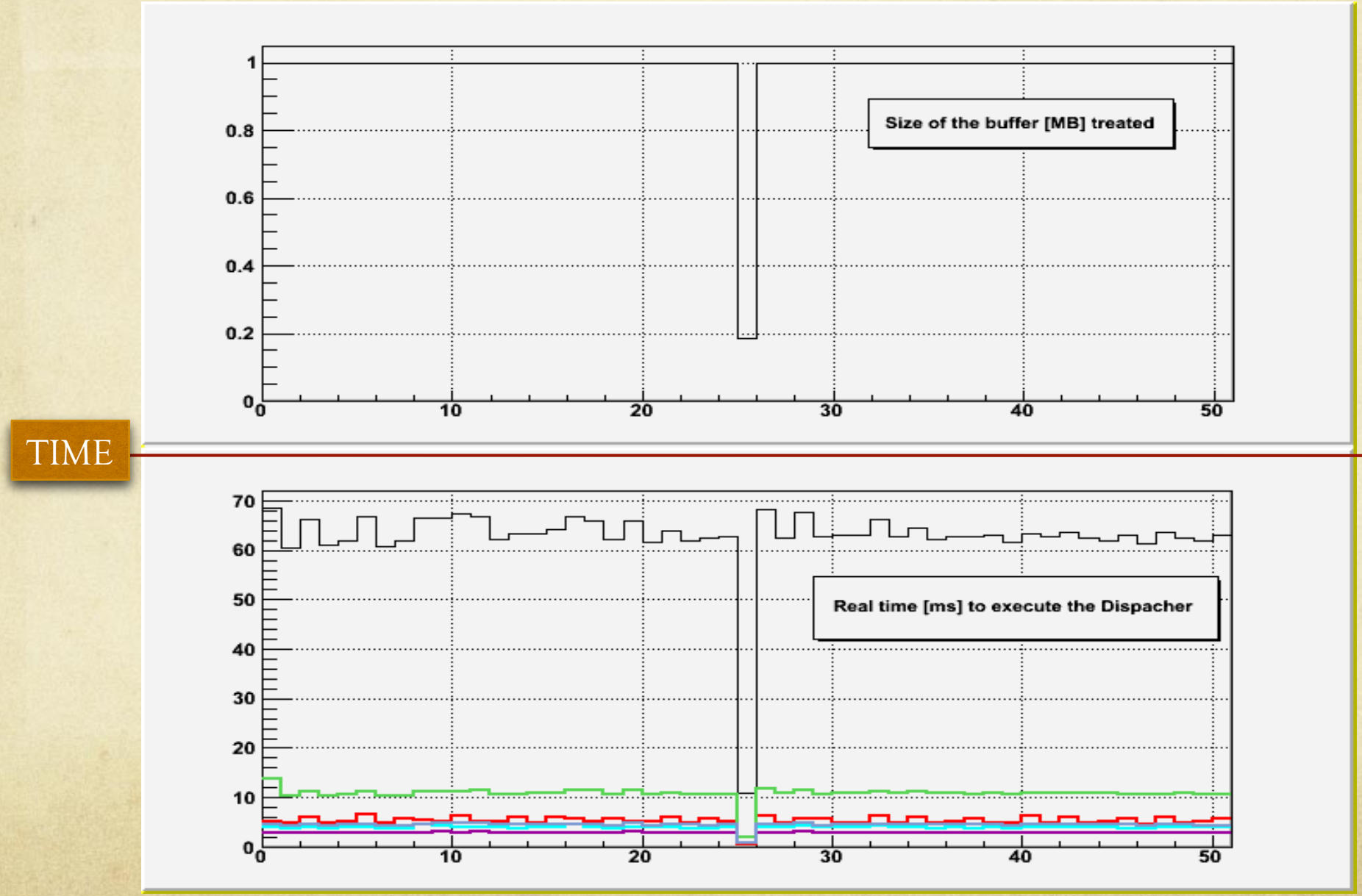
A FrameDispacher is a consumer to which is attached (several) Watchers



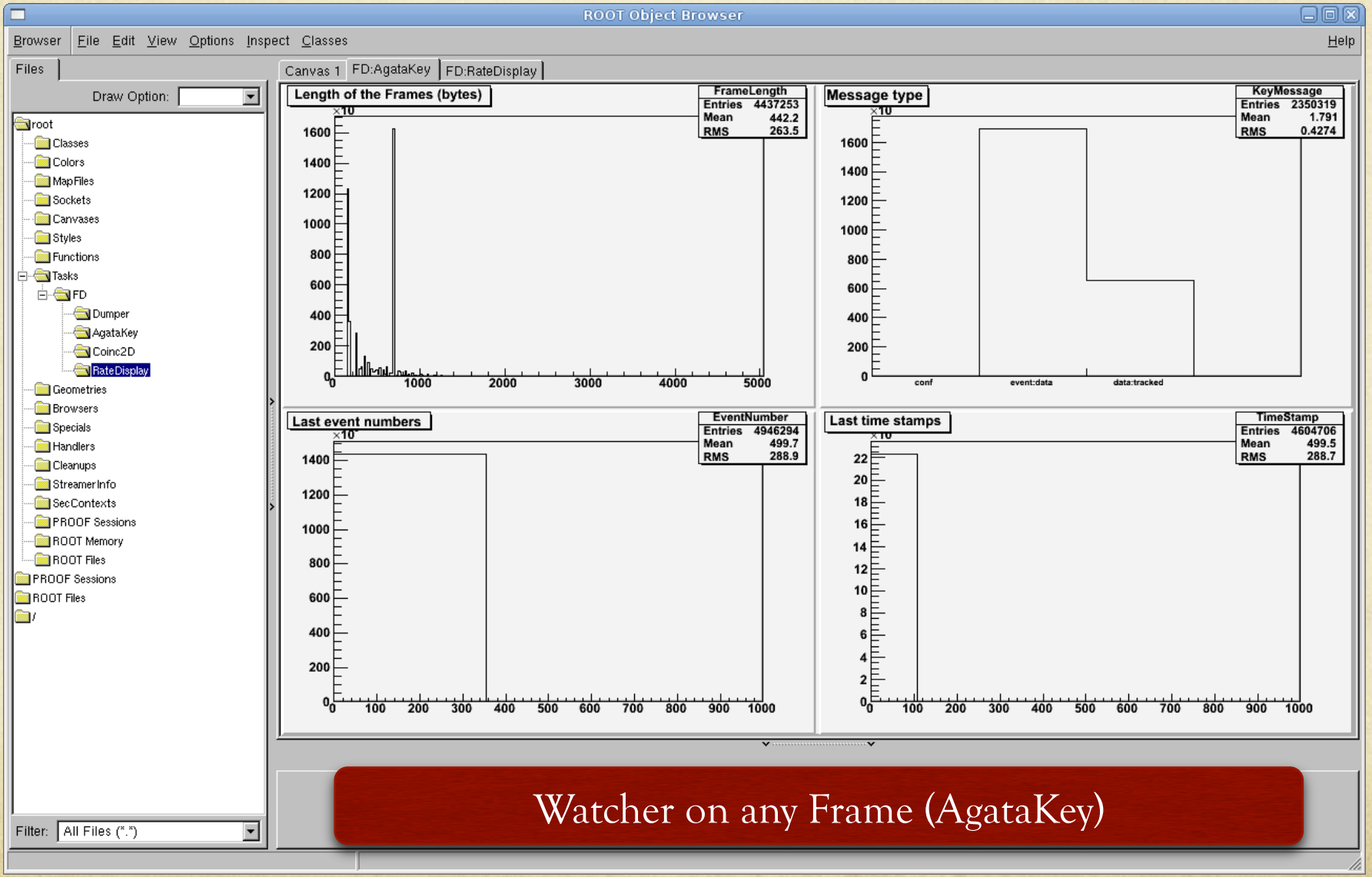

data_1R
data_1G
data_1B
data_2R
data_2G
data_2B
ancillary_producer
prepro_1R
prepro_1G
prepro_1B
prepro_2R
prepro_2G
prepro_2B
ancillary_filter
psa_1R
psa_1G
psa_1B
psa_2R
psa_2G
psa_2B
event_builder
merger
WatcherClient
W
tracking
last_actor


WatcherClients (FrameDispacher)
Dump Frames on files
Check Keys
Signals from the crystals
Fold, singles and
gamma-gamma matrix
Rate in Crystals (core/segments)
Browser
File
Edit
View
Options
Inspect
Classes
Help
Files
Draw Option:
root
Classes
Colors
MapFiles
Sockets
Canvases
Styles
Functions
Tasks
LoopOnTask
LoopControl
ADF
Dumper
AgataKey
Vertex
Crystal
Crystal
CCrystal
PSA
PSADisplay
Tracked
Coinc2D
TrackDisplay
EventPSA
RateDisplay
Geometries
Browsers
Canvas 1
Length of the Frames (bytes)
Message type
KeyMessage
Entries
Mean
RMS

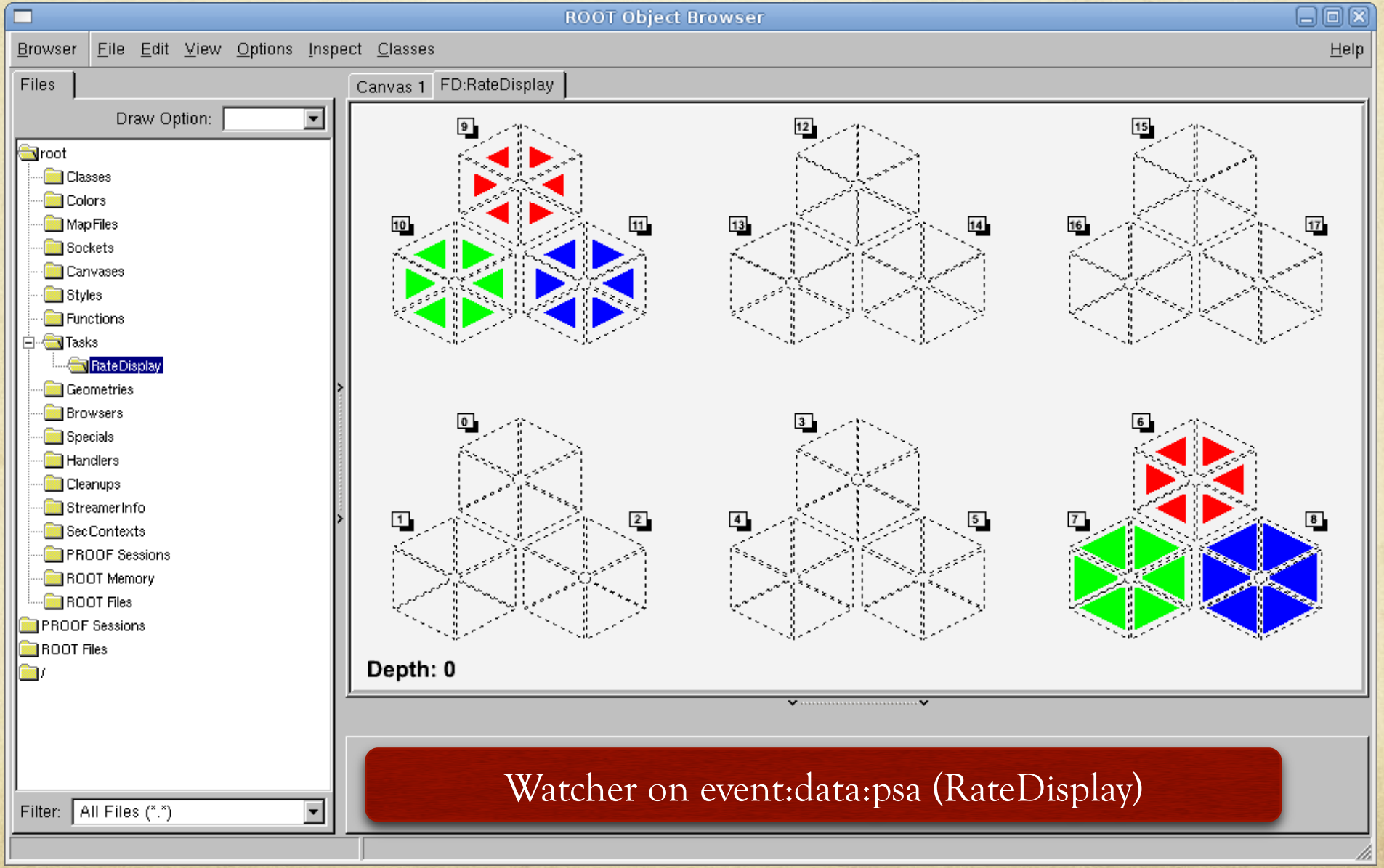
Watchers : running time



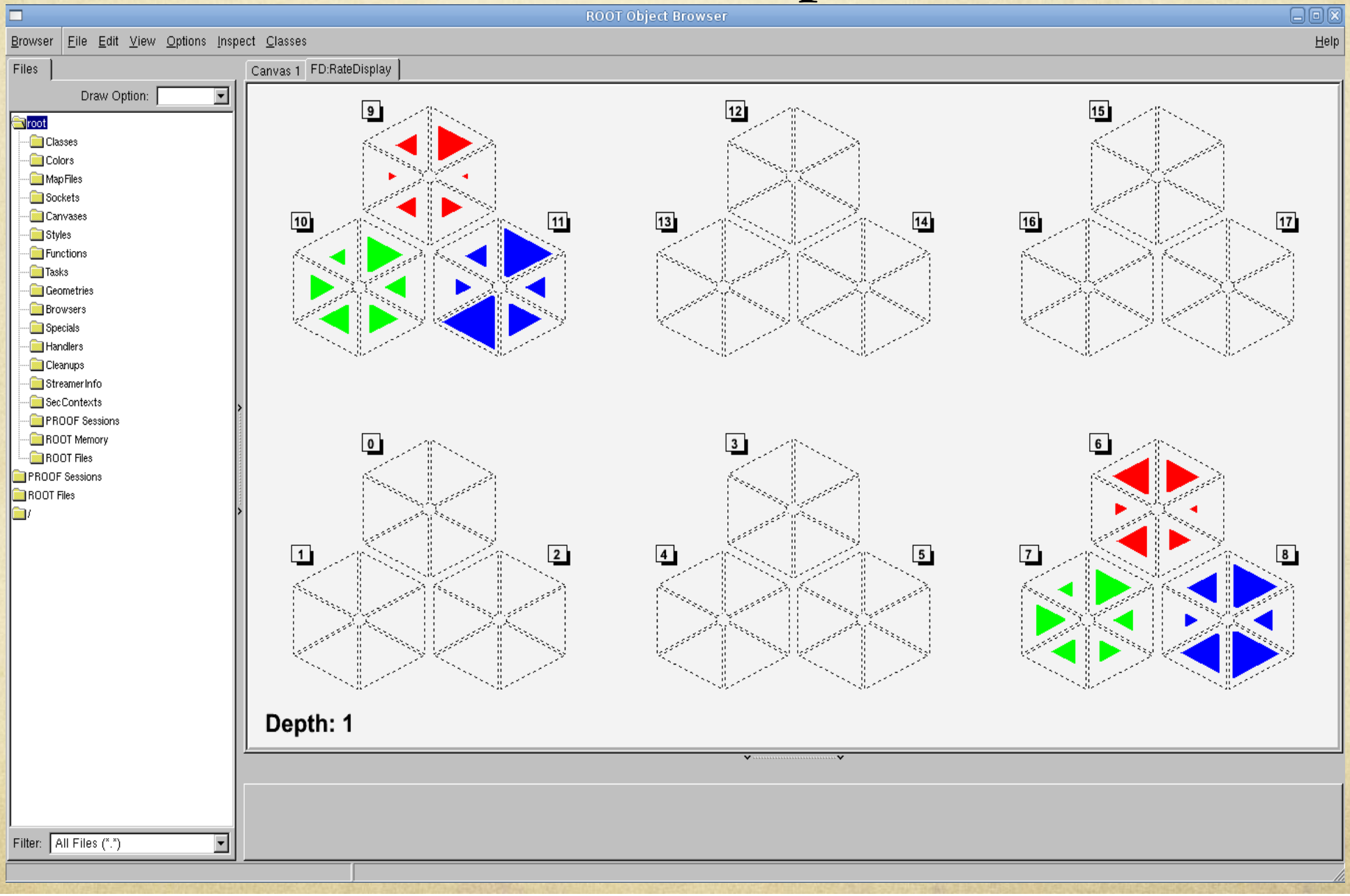
From the last experiment



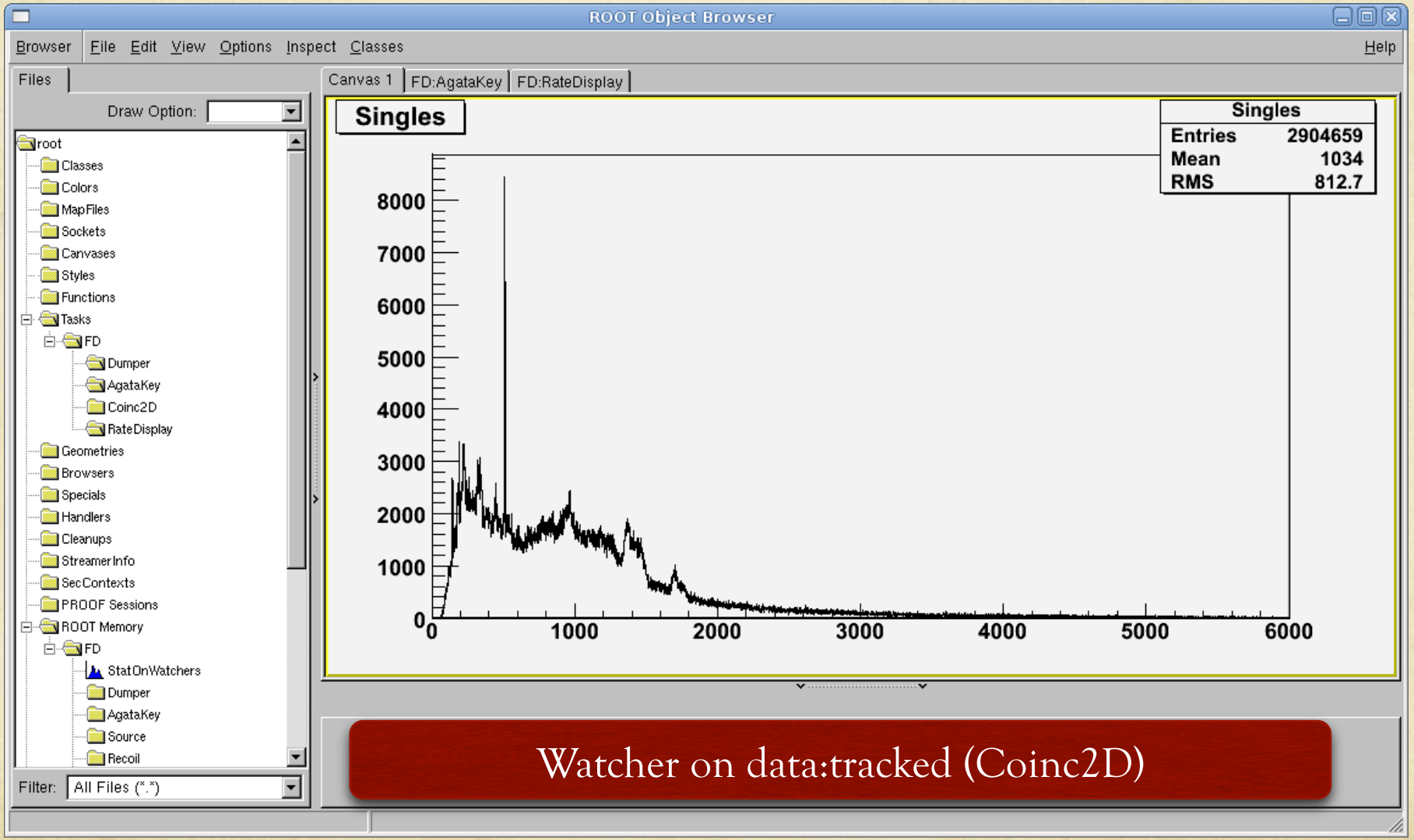
From the last experiment



From the last experiment

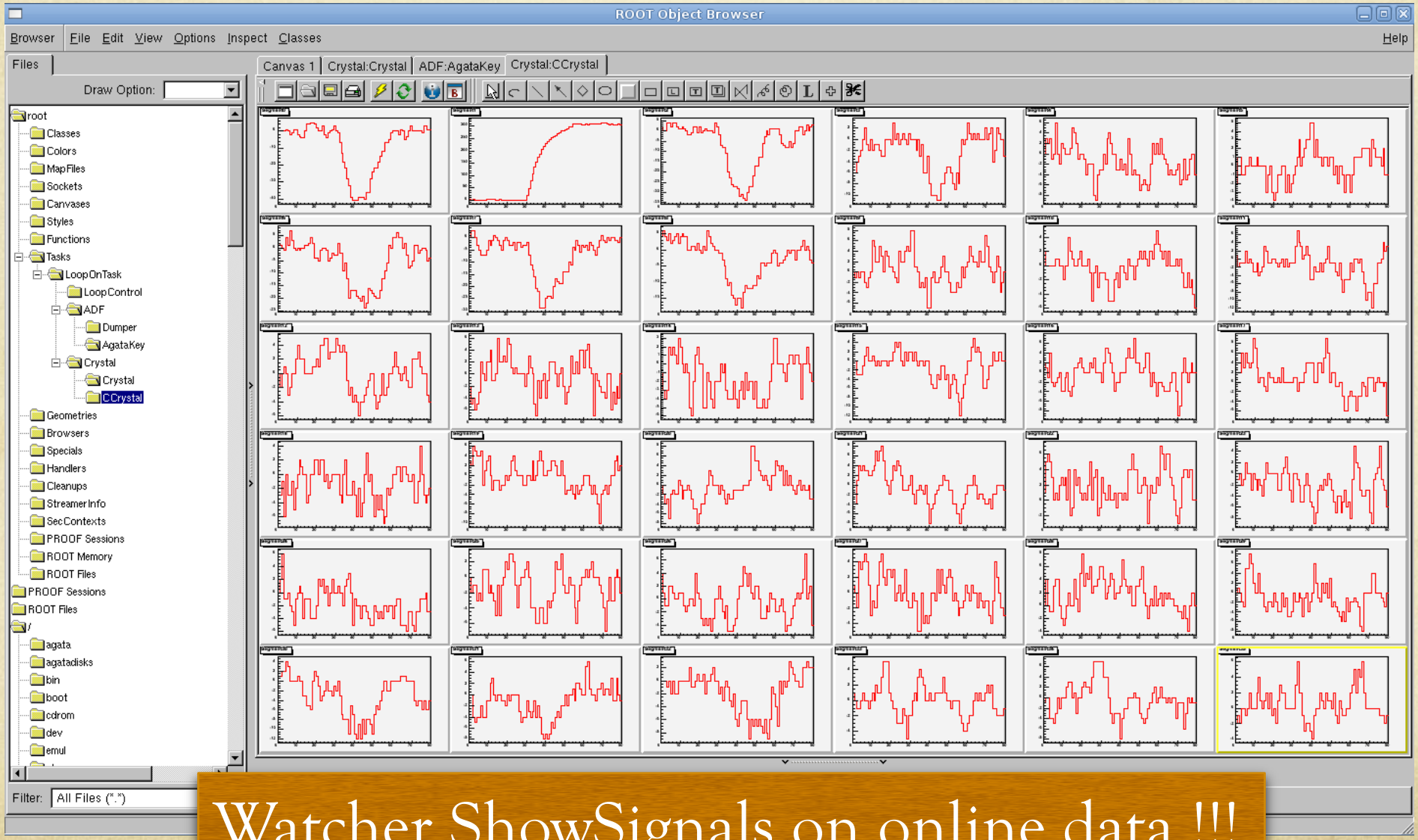


From the last experiment



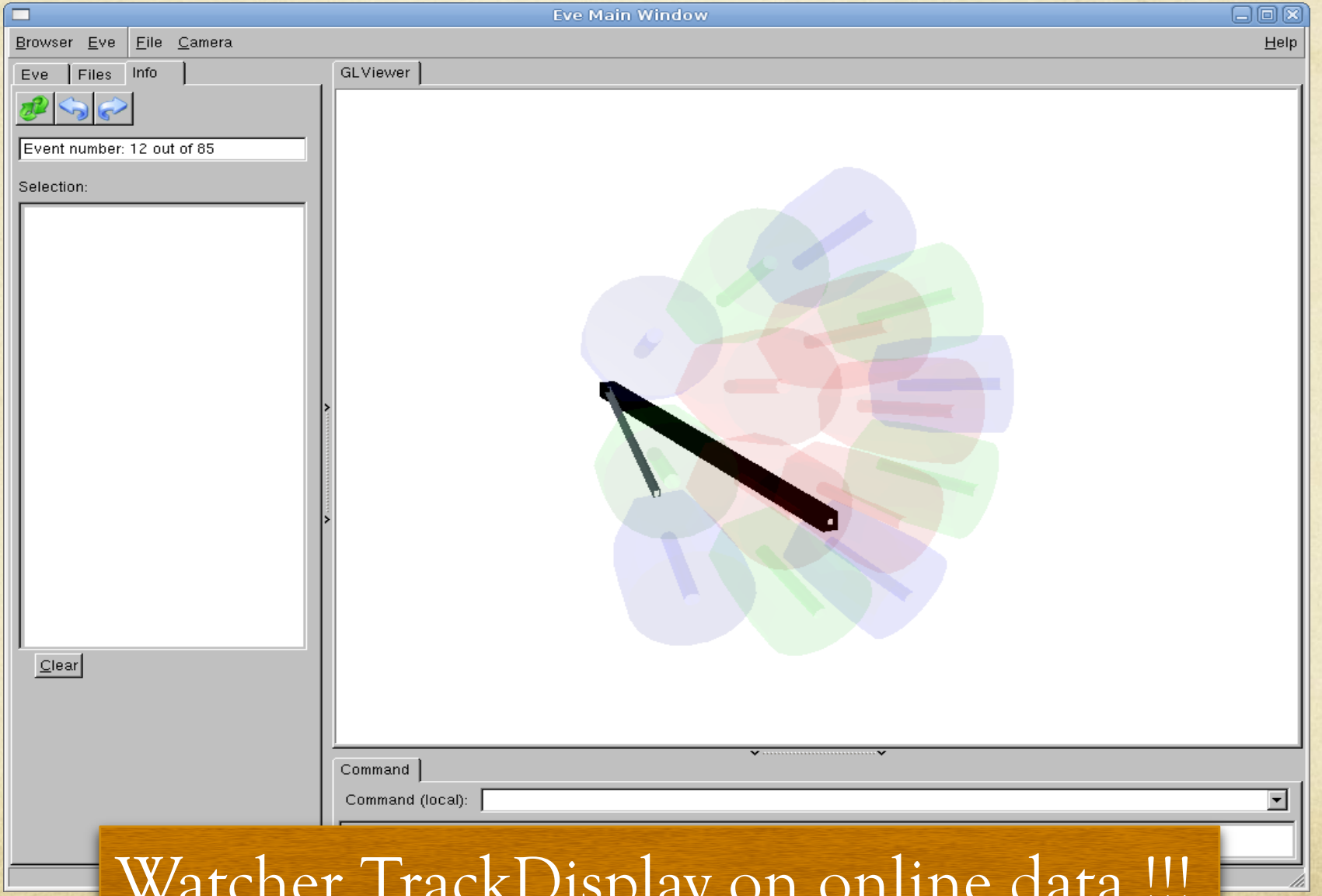
Doppler correction available (VertexFrame) .... to be implemented ...

From the last experiment



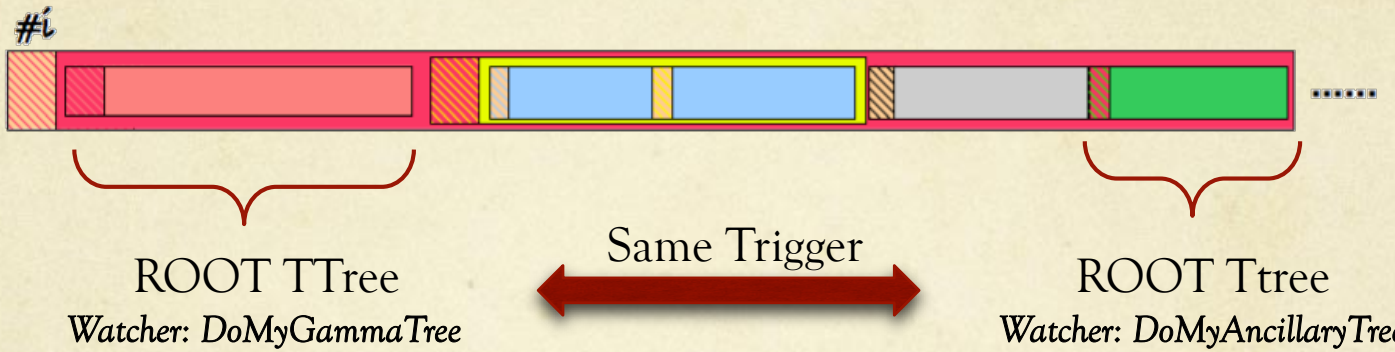
Watcher ShowSignals on online data !!!

From the last experiment



Watcher TrackDisplay on online data !!!

Since the DA workshop in december



Trigger on event:data with inside a tracked and an ancillary frame

see :

PrismaAndDanteWatchers.h and PrismaAndDanteWatchers.C

With Prisma !

Other parts of the GammaWare

- SpectrumPlayer
- LevelSchemePlayer
- Graphical Fit
- Spectra converters
- All ROOT facilities for free !

Conclusions & perspectives

- More stabilities
- More tests/benchmarks
- More documentations
- New « fast » features (« anti-trigger »)
- LevelShemePlayer
  + interactions with matrix, cube like ... to come
- New watchers ?

- What else ?

```
Coinc2D::Coinc2D() :
      TrackedWatcher("Coinc2D","Check singles and build a gamma-gamma matrix"),
      fSingles(0x0),
      fSinglesDoppler(0x0),
      fFold(0x0),
      fGxG(0x0)
{
      fSingles = new TH1F("Singles","Singles",6000,0,6000);
      AddToPool(fSingles);
      fSinglesDoppler = new TH1F("SinglesDoppler","Singles with Doppler correction",6000,0,6000);
      TagOn(fSinglesDoppler);
      AddToPool(fSinglesDoppler);
      fFold = new TH1F("Fold","Fold distributions",10,0,10);
      TagOn(fFold);
      AddToPool(fFold);
      fGxG     = new TH2F("GxG","Gamma Gamma matrix",2000,0,1000,2000,0,1000);
      AddToPool(fGxG);
}
void Coinc2D::Exec(Option_t */*option*/)
{
      const GammaTrackedInterface *data = GetCstDataPointer<GammaTrackedInterface>(fFrame);

      // now fill histograms
      UInt_t fold = data->GetNbGamma();
      fFold->Fill( fold );
      for (UShort_t i = 0u; i < fold; i++) {
            const TrackedHit *gamma1 = data->GetGamma(i);
            Double_t edc1 = VertexWatcher::theCurrentVertex()->DopplerCorrection(gamma1);
            fSingles->Fill(gamma1->GetE());
            fSinglesDoppler->Fill(edc1);
            for (UShort_t j = i+1u; j < fold; j++) {
                  const TrackedHit *gamma2 = data->GetGamma(j);
                  Double_t edc2 = VertexWatcher::theCurrentVertex()->DopplerCorrection(gamma2);
                  fGxG->Fill(edc1,edc2); fGxG->Fill(edc2,edc1);
            }
      }
}
```

ConfAgent and ADF.conf

```
#////ADF::ConfAgent_beg//// Agata 1 Init
# `
Endian: kLittle
#
KeyFactory: Default
KeyFactory: Agata
#
FrameFactory: Agata
#
PrimaryKey: Default FS 0 1
#
AutoConf: Agata conf:global 2 0 Agata conf:global 0 0
#
#////ADF::ConfAgent_end////
```

One in the distribution
One in the algo path

\+

```
# Current Frames going through the data flow
#
#
#Frame: Agata agata 2 0 Agata agata 0 0
Frame: Agata data:tracked 2 0 Agata data:tracked 65000 1
Frame: Agata data:crystal 2 0 Agata data:crystal 65000 0
```

```
#////ADF::AgataKeyFactory_beg//// Init
Agata            0xFA000000   0xFF000000
Conf             0xFA0002000 xFF00FF00
conf:global      0xFA000200   0xFFFFFFFF
conf:crysta      l0xFA010201  0xFFFFFFFF
conf:psa                 0xFA010202   0xFFFFFFFF
Data             0xFA010000   0xFFFF0000
data:crystal     0xFA010101   0xFFFFFFFF
data:psa                 0xFA010102   0xFFFFFFFF
event:data:psa 0xFA010103   0xFFFFFFFF
event:data       0xFA010104   0xFFFFFFFF
data:tracked     0xFA010105   0xFFFFFFFF
#
#////ADF::AgataKeyFactory_end////
#
```

\+

Full definition of the Keys
(for the Keyfactory)