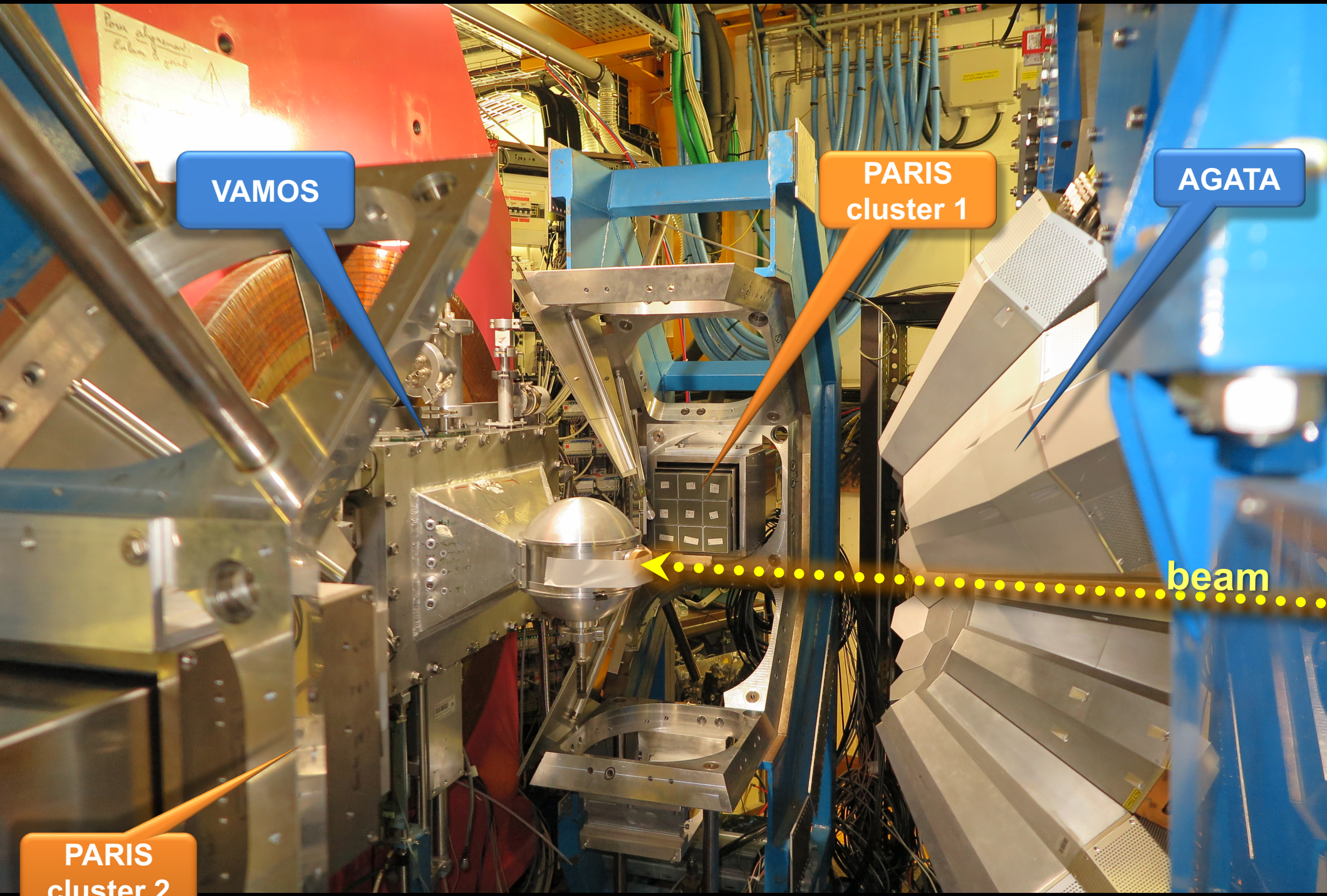
Jerzy Gręboszz
Kraków, Poland

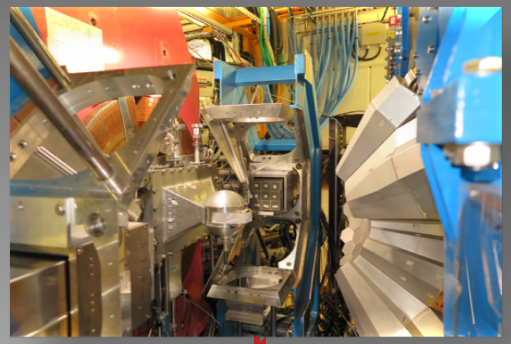
Is it possible to have a „complete” on-line analysis

for „AGATA + VAMOS +...” experiments ?


VAMOS
PARIS
cluster 1
AGATA
beam
PARIS
cluster 2

Idea of analysis (or simple monitoring)

experiment



DAQ

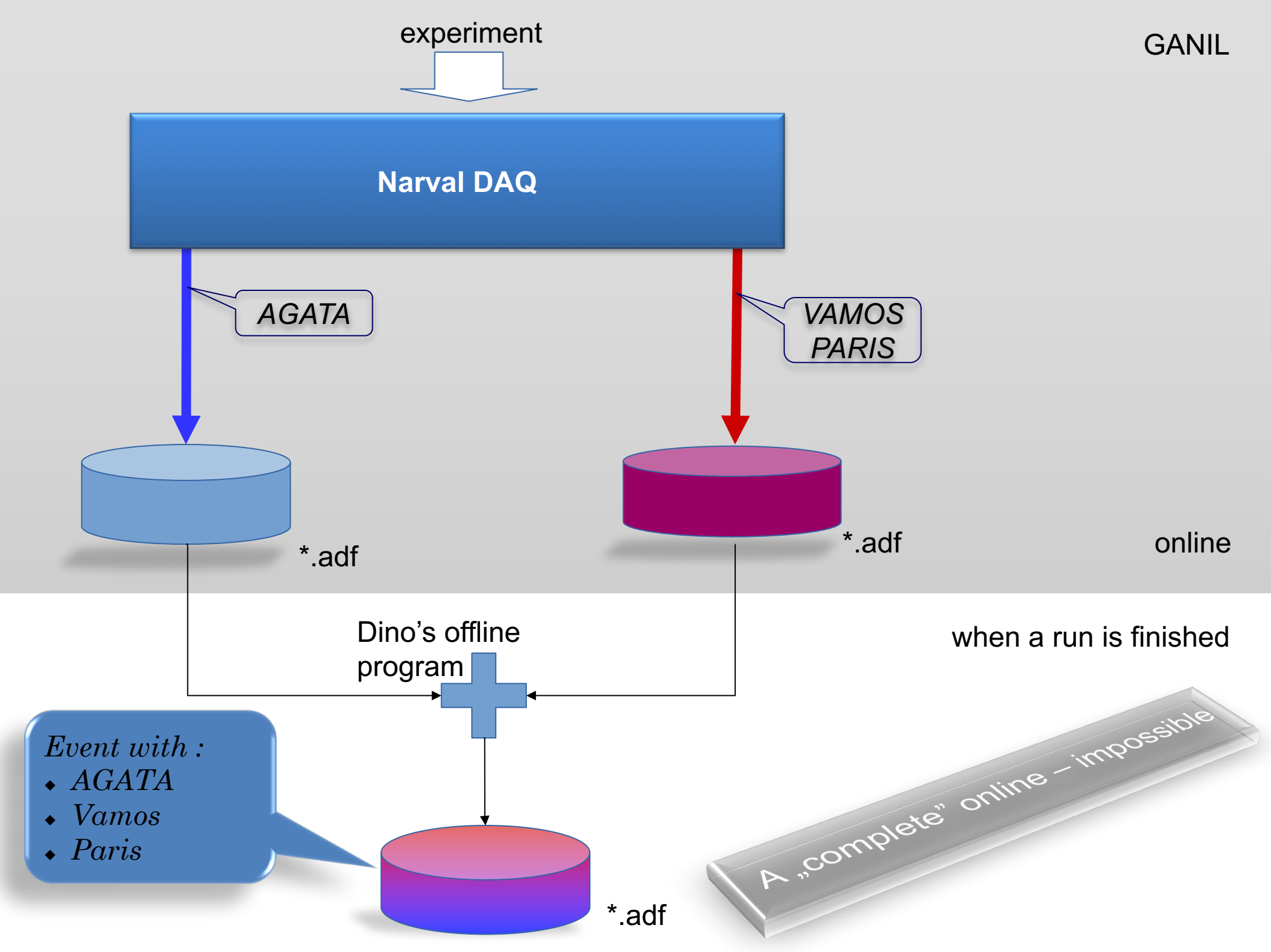
online

Data store

near-line

Greware: Spy + GUI

for example: Galileo


experiment
GANIL
Narval DAQ
AGATA
VAMOS
PARIS
*.adf
*.adf
online
Dino's offline
program
when a run is finished
Event with :
AGATA
Vamos
Paris
*.adf
A „complete" online – impossible

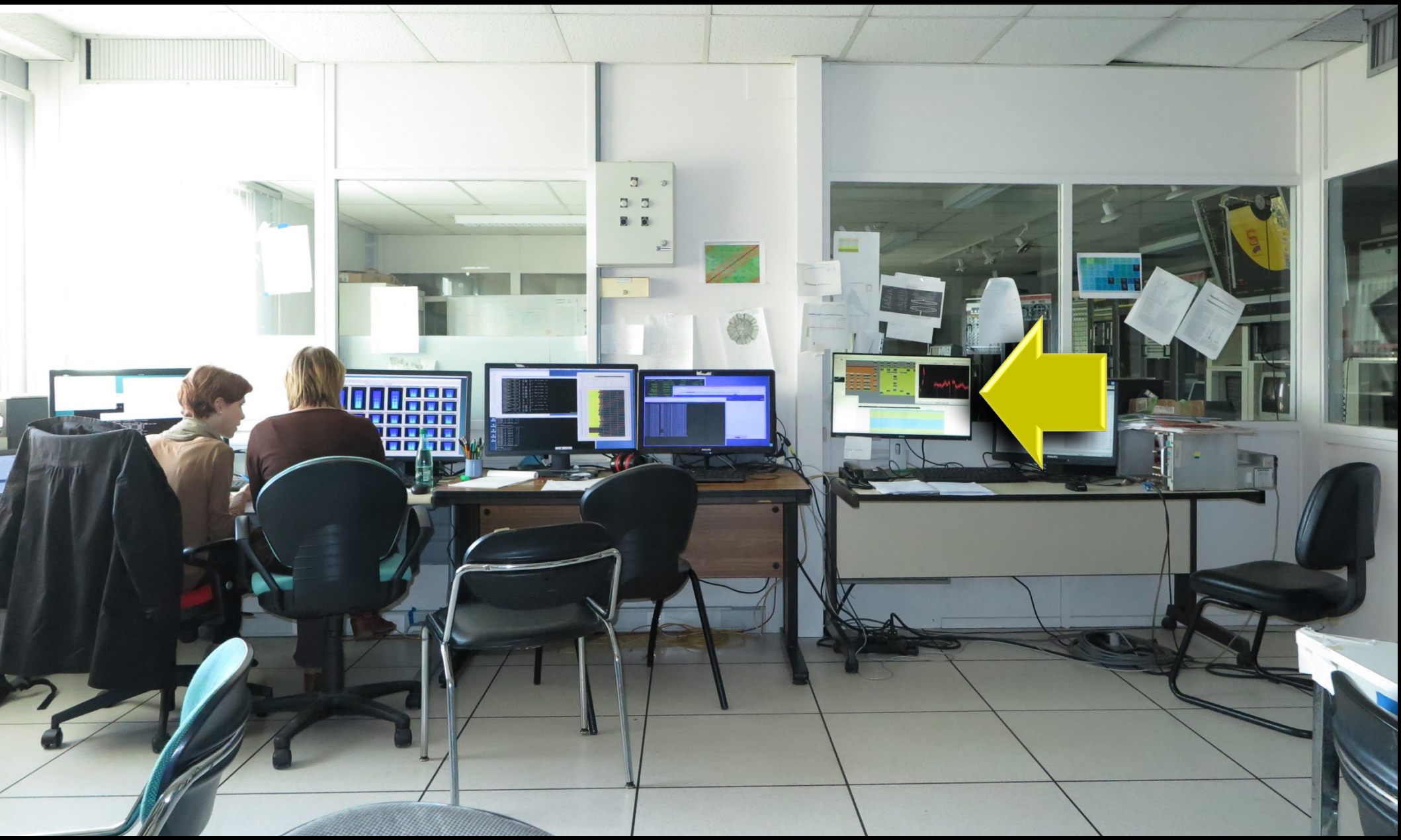
What we need for our experiment?

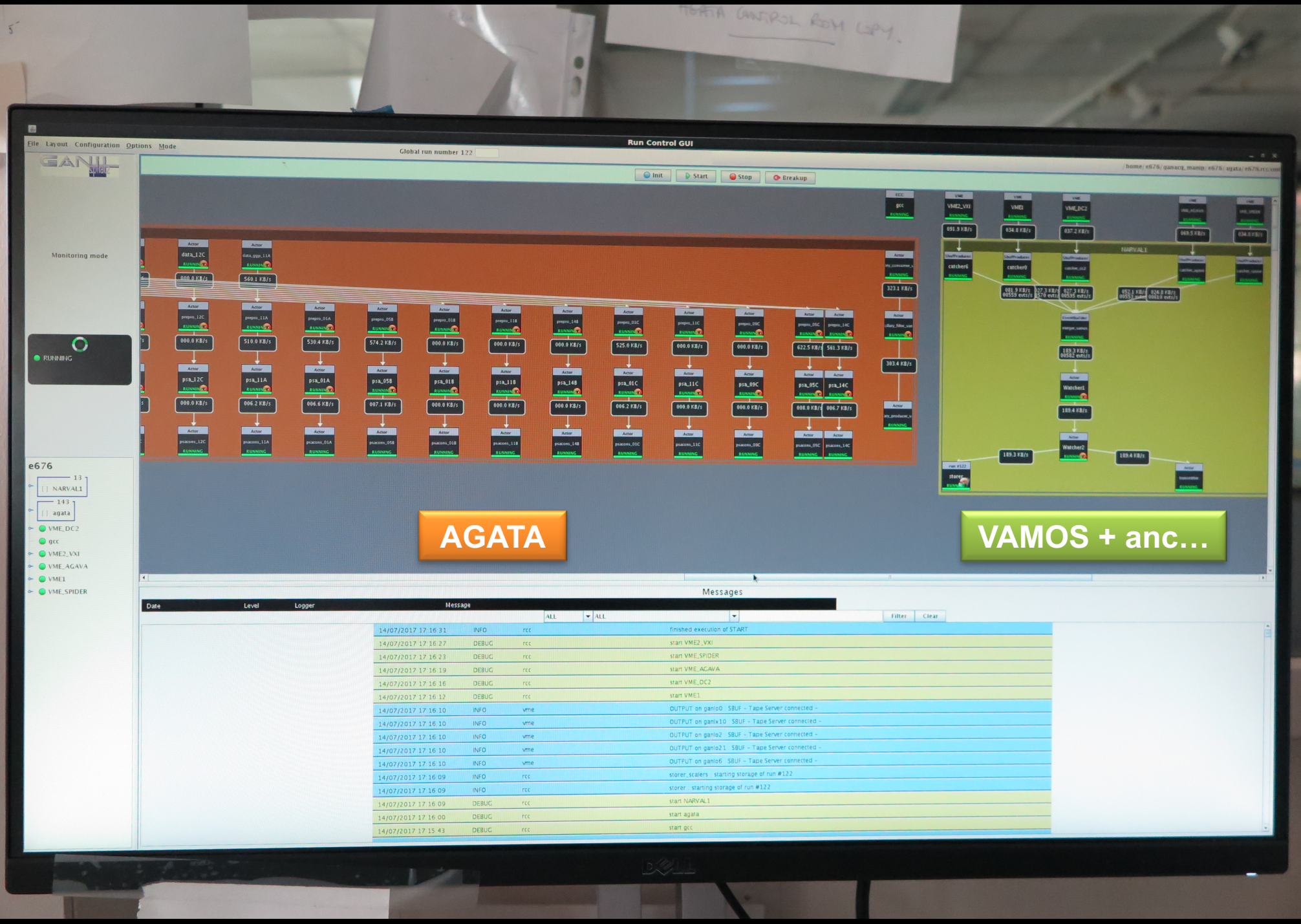
Adam Maj (the chief of PARIS detector), being at GANIL some months ago asked me to prepare an online analysis for the coming PARIS experiment. He said:

We need to have Paris + AGATA **coincidence** data
on a screen - online

without necessity to wait for making so called replay, making root trees, etc.

…




Run Control GUI
File Layout Configuration Options Mode
Global run number 122
Init
Start
Stop
Breakup
Monitoring mode
RUNNING
e676
NARVAL1
agata
VME_DC2
gcc
VME2_VXI
VME_AGAVA
VME1
VME_SPIDER
AGATA
VAMOS + anc…
Messages
Date
Level
Logger
Message
Filter
Clear
14/07/2017 17:16:31 INFO rcc finished execution of START
14/07/2017 17:16:27 DEBUG rcc start VME2_VXI
14/07/2017 17:16:23 DEBUG rcc start VME_SPIDER
14/07/2017 17:16:19 DEBUG rcc start VME_AGAVA
14/07/2017 17:16:16 DEBUG rcc start VME_DC2
14/07/2017 17:16:12 DEBUG rcc start VME1
14/07/2017 17:16:10 INFO vme OUTPUT on ganio0 SBUF - Tape Server connected -
14/07/2017 17:16:10 INFO vme OUTPUT on ganix10 SBUF - Tape Server connected -
14/07/2017 17:16:10 INFO vme OUTPUT on ganio2 SBUF - Tape Server connected -
14/07/2017 17:16:10 INFO vme OUTPUT on ganio21 SBUF - Tape Server connected -
14/07/2017 17:16:10 INFO vme OUTPUT on ganio6 SBUF - Tape Server connected -
14/07/2017 17:16:09 INFO rcc storer_scalers starting storage of run #122
14/07/2017 17:16:09 INFO rcc storer starting storage of run #122
14/07/2017 17:16:09 DEBUG rcc start NARVAL1
14/07/2017 17:16:00 DEBUG rcc start agata
14/07/2017 17:15:43 DEBUG rcc start gcc


Actor
data_ggp_01A
RUNNING
800.3 KB/s
Actor
prepro_10A
41.8 KB/s
Actor
psa_10A
0.1 KB/s
Actor
psacons_10A
Actor
data_ggp_05B
000.0 KB/s
Actor
prepro_04A
876.9 KB/s
Actor
psa_04A
011.0 KB/s
Actor
psacons_04A
Actor
data_03A
000.0 KB/s
Actor
prepro_03A
000.0 KB/s
Actor
psa_03A
000.0 KB/s
Actor
psacons_03A
Actor
data_13A
000.0 KB/s
Actor
prepro_13A
000.0 KB/s
Actor
psa_13A
000.0 KB/s
Actor
psacons_13A
Actor
data_ggp_14B
971.9 KB/s
Actor
prepro_02A
001.0 MB/s
Actor
psa_02A
013.2 KB/s
Actor
psacons_02A
Actor
data_ggp_01C
971.5 KB/s
Actor
prepro_12A
001.0 MB/s
Actor
psa_12A
012.9 KB/s
Actor
psacons_12A
Socket
IP + port
Socket
IP + port
Socket
IP + port
Socket
IP + port
Socket
IP + port

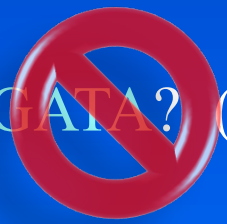
It is still before event builder

Event has to be build according to timestampes of every subevent

GSI → GER, FRS, DGF, HEC,

(4 types of subevents to be matched into one event)

GANIL → Vamos + AGATA? (2 types of subevents?)



…not so easy

GANIL → Vamos + ~28 Agata crystals

all (29) of them we should take from sockets…

Opening a socket with **proper** parameters

```
int Tsocket_for_data::open_socket (string hostname, int port)
{
    struct hostent *he;
     struct sockaddr_in their_addr;
     struct sockaddr_in l_addr;

       if ( ( sockfd = socket ( PF_INET, SOCK_STREAM, 0 ) ) == -1 )
       {  /*…*/  }

        l_addr.sin_family = PF_INET;
        l_addr.sin_port = htons ( 0 );
        l_addr.sin_addr.s_addr = htonl ( INADDR_ANY );
        memset ( & ( l_addr.sin_zero ), '\0', 8 );

       if ( setsockopt ( sockfd, SOL_SOCKET, SO_REUSEADDR, &sock_opt, sizeof ( int ) ) == -1 )
       {  /*…*/  }

       if ( bind ( sockfd, ( struct sockaddr * ) &l_addr, sizeof ( struct sockaddr ) ) == -1 )
       {  /*…*/  }

        if ( ( he=gethostbyname ( hostname.c_str() ) ) == NULL )
         {  /*…*/  }

        their_addr.sin_family = PF_INET;
        their_addr.sin_port = htons ( port );
        their_addr.sin_addr = * ( ( struct in_addr * ) he->h_addr );
        memset ( & ( their_addr.sin_zero ), '\0', 8 );

        // cout << "Trying to connect... "  << endl;
       if ( connect ( sockfd, ( struct sockaddr * ) &their_addr, sizeof ( struct sockaddr ) ) == -1 )
        {
            perror ( (description + " ---> connect error: ").c_str()  );
        }
        cout << description << ": Succes with opening host "<< my_host << " port nr ---> " << port  << endl;
        return 1;
}
```

Reading
a block of bytes
from the socket

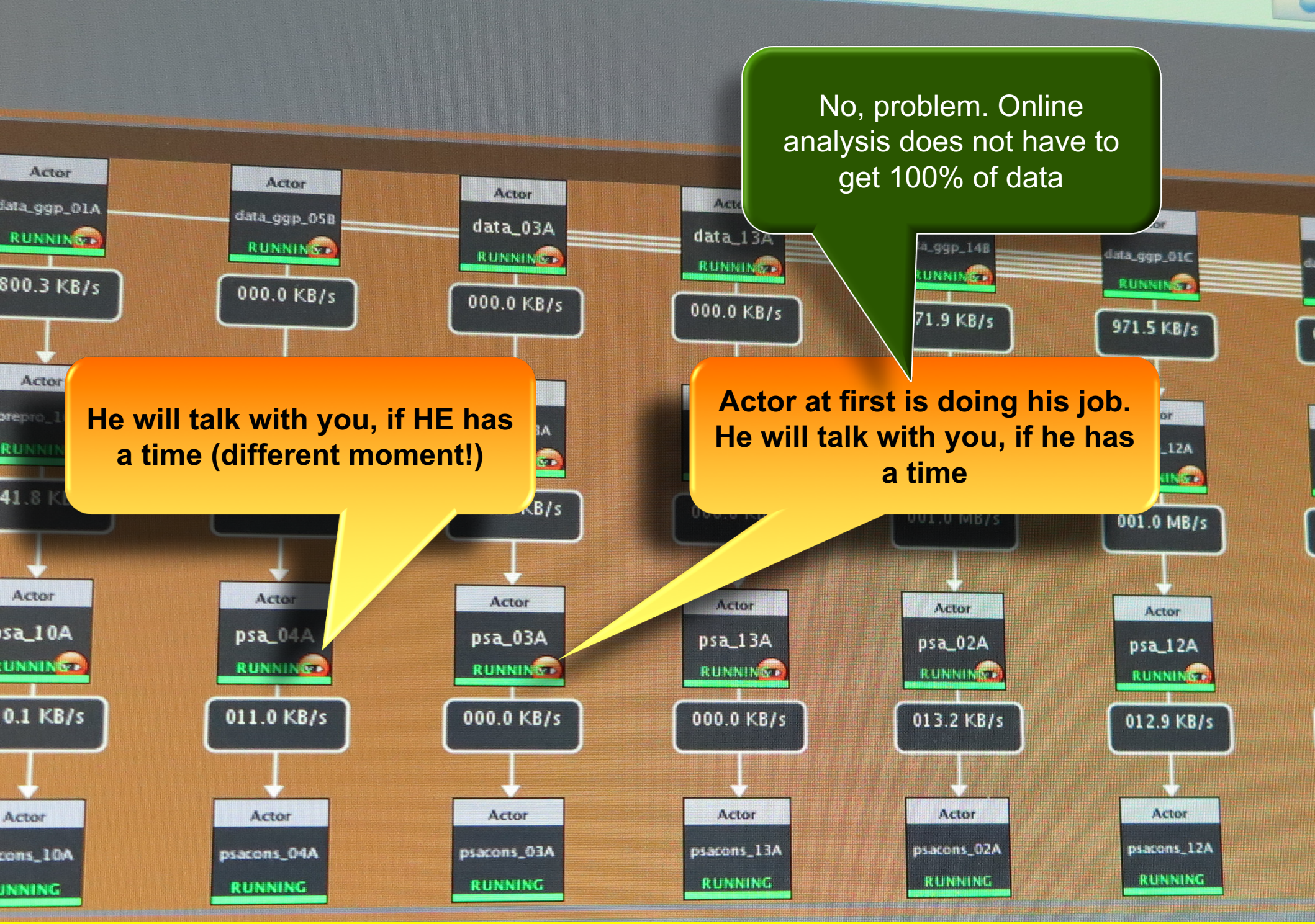
```
//...
    nread = recv ( sd, buf, nleft, MSG_DONTWAIT );        //    nread = recv ( sd, buf, nleft, 0 );

    if (nread < 0)
    {
       if(errno == EAGAIN)
          cout << "Error called: EAGAIN" << endl;
       else if(errno == EWOULDBLOCK)
          cout << "Error called: EWOULDBLOCK" << endl;

       throw Tsocket_recv_error{ " Can not receive data"};
    }
    else if (nread == 0)
    {
       if(errno == EAGAIN)
          cout << "Error called: EAGAIN" << endl;
       else if(errno == EWOULDBLOCK)
          cout << "Error called: EWOULDBLOCK" << endl;

        throw Tsocket_recv_no_data_now{ " No data available\n"};
    }
     // >0  success
    //  cout << "Tsocket_fo_data::reads     recv:  nread = " << nread << endl;
```

What am I doing wrong?


No, problem. Online analysis does not have to get 100% of data
He will talk with you, if HE has a time (different moment!)
Actor at first is doing his job. He will talk with you, if he has a time
No guarantee, that the subevents belonging to the same event.

end of a dream

the socket connection with PSA actors – is unreliable (for me).

So I changed the tactics:

PSA actors write their data on disk (every minute), so…

DAQ

Greware
Spy + GUI

100% of the
statistics

Data from DAQ

written (in bulk)
every several seconds

Current run data file

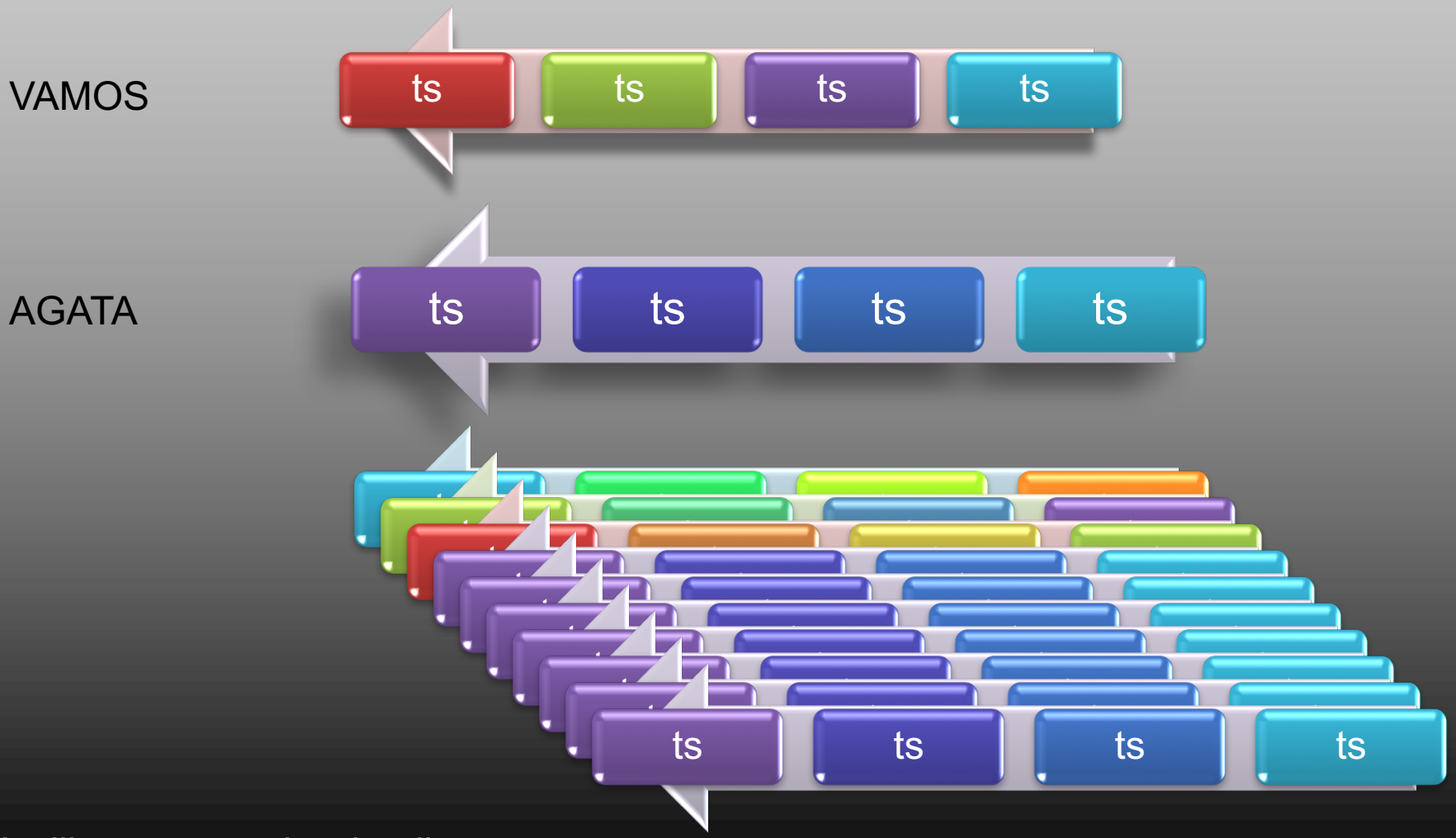
×29

„on-line” is when…

prise is: 30 - 60 s of delay

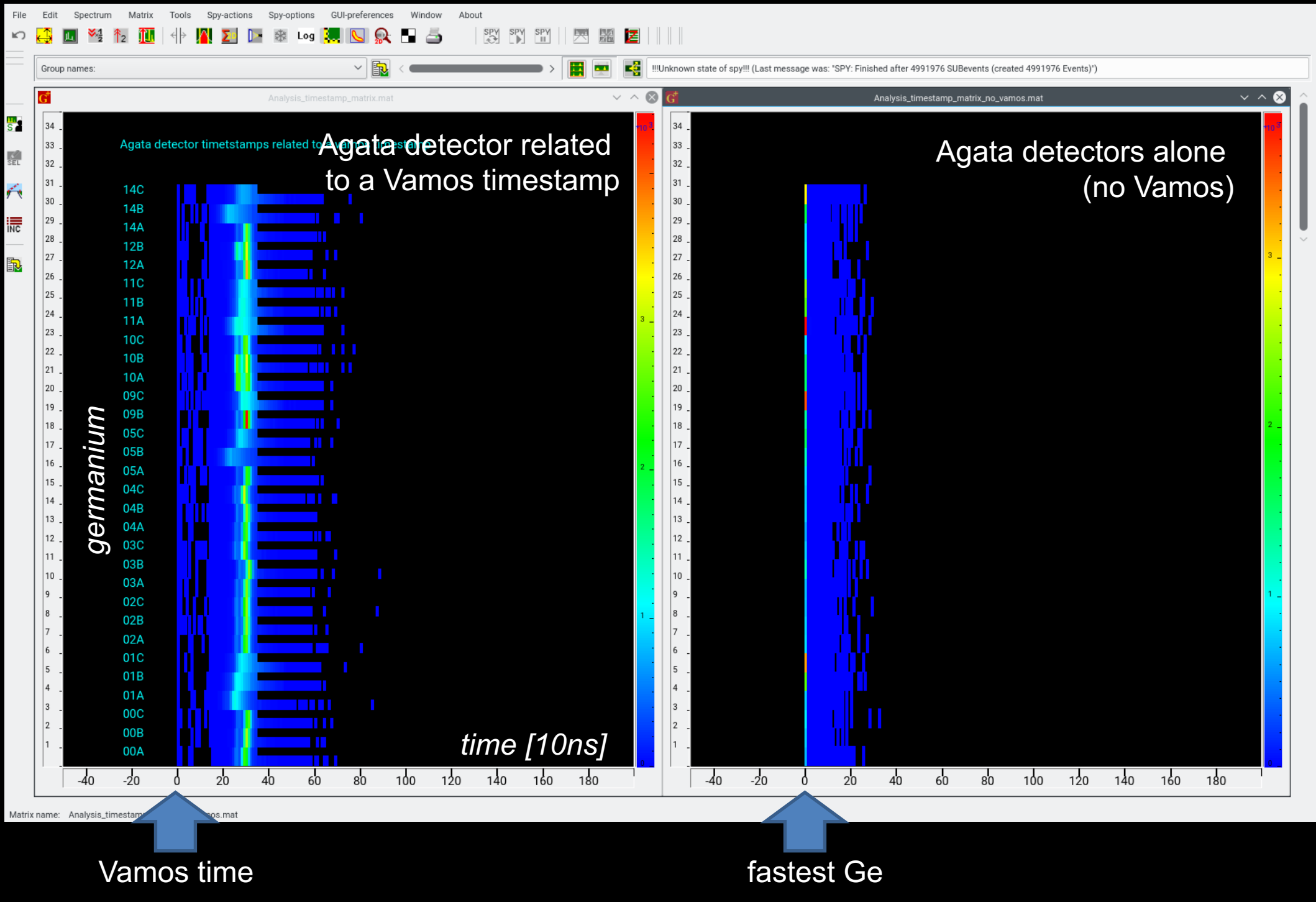
29 types of subevents may create on event

Depending on their timestamps (ts)



I will spare you the details…

Just look at this.

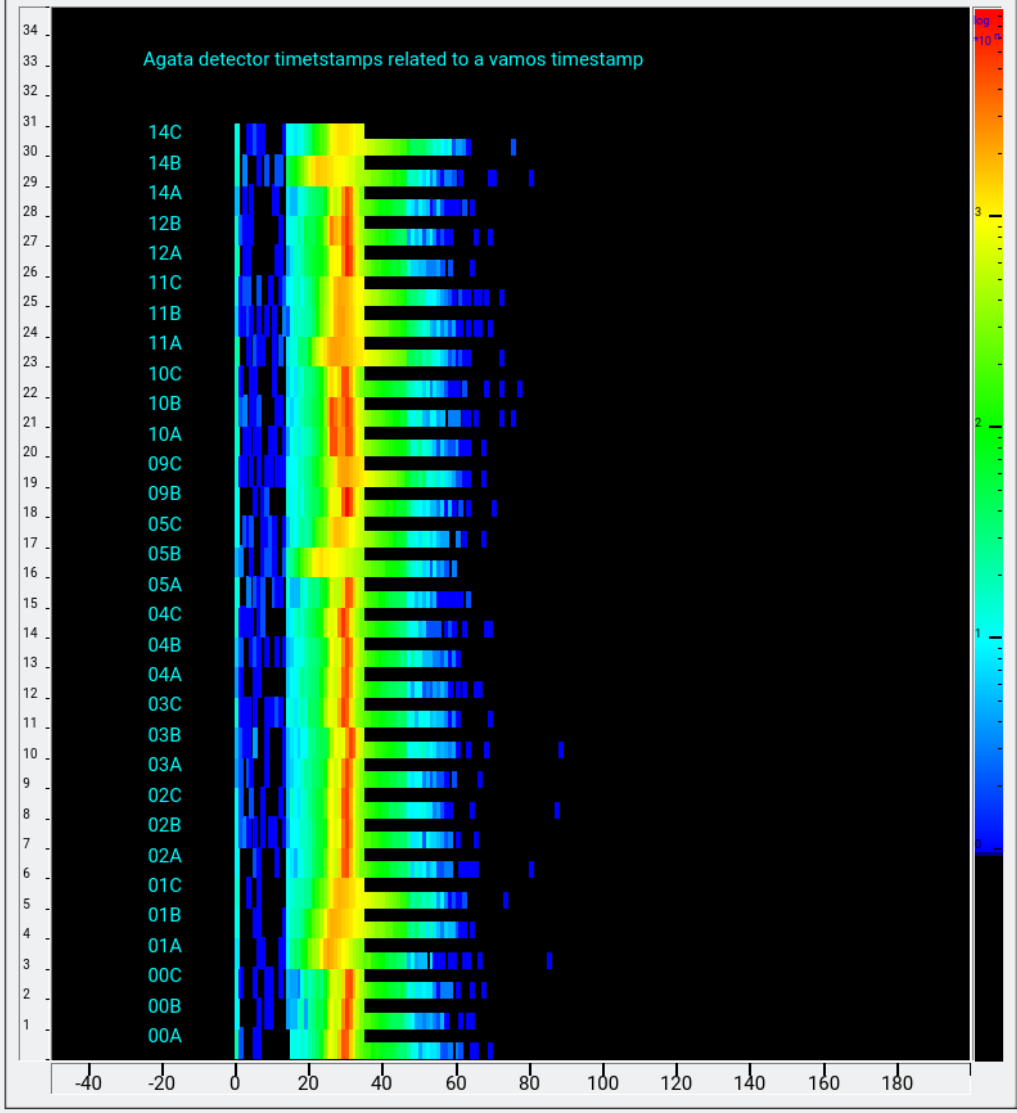

File
Edit
Spectrum
Matrix
Tools
Spy-actions
Spy-options
GUI-preferences
Window
About
Group names:
!!!Unknown state of spy!!! (Last message was: "SPY: Finished after 4991976 SUBevents (created 4991976 Events)")
Analysis_timestamp_matrix.mat
Analysis_timestamp_matrix_no_vamos.mat
Agata detector timetstamps related to a Vamos timestamp
Agata detector related to a Vamos timestamp
Agata detectors alone (no Vamos)
germanium
14C
14B
14A
12B
12A
11C
11B
11A
10C
10B
10A
09C
09B
05C
05B
05A
04C
04B
04A
03C
03B
03A
02C
02B
02A
01C
01B
01A
00C
00B
00A
time [10ns]
-40 -20 0 20 40 60 80 100 120 140 160 180
Matrix name:
Vamos time
fastest Ge

File Edit Spectrum Matrix Tools Spy-actions Spy-options GUI-preferences Window About

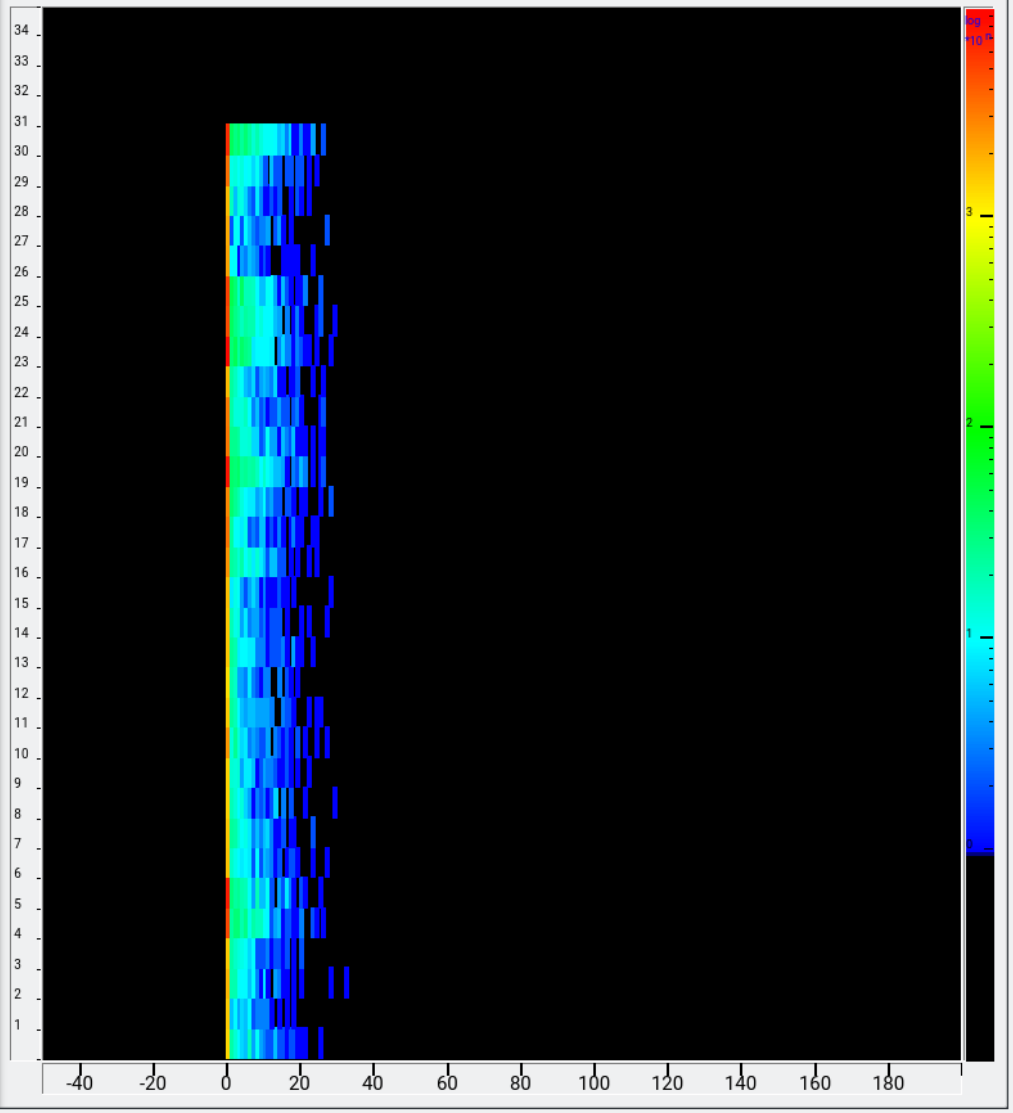
Group names:

!!!Unknown state of spy!!! (Last message was: "SPY: Finished after 4991976 SUBevents (created 4991976 Events)")

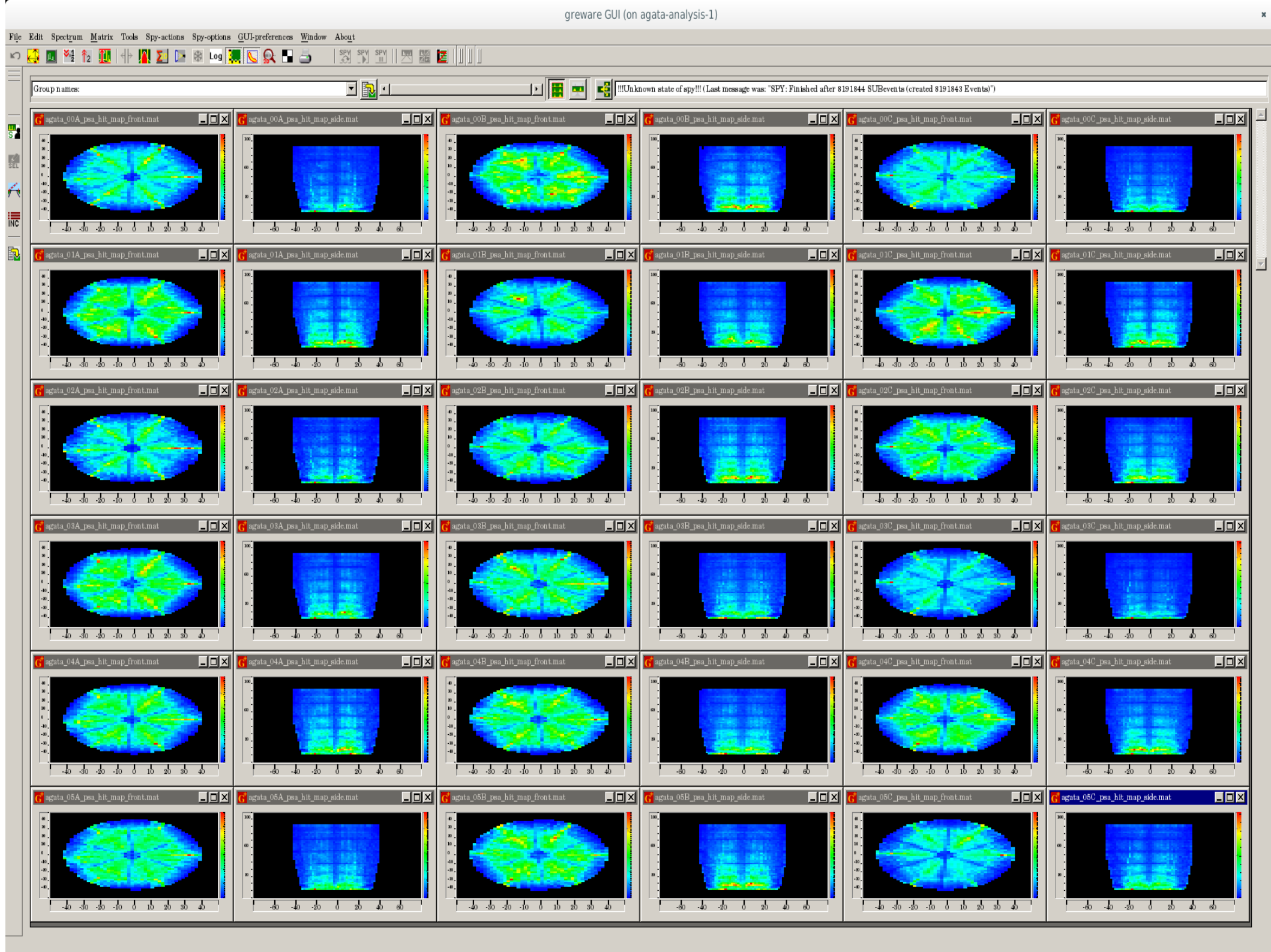
Analysis_timestamp_matrix.mat



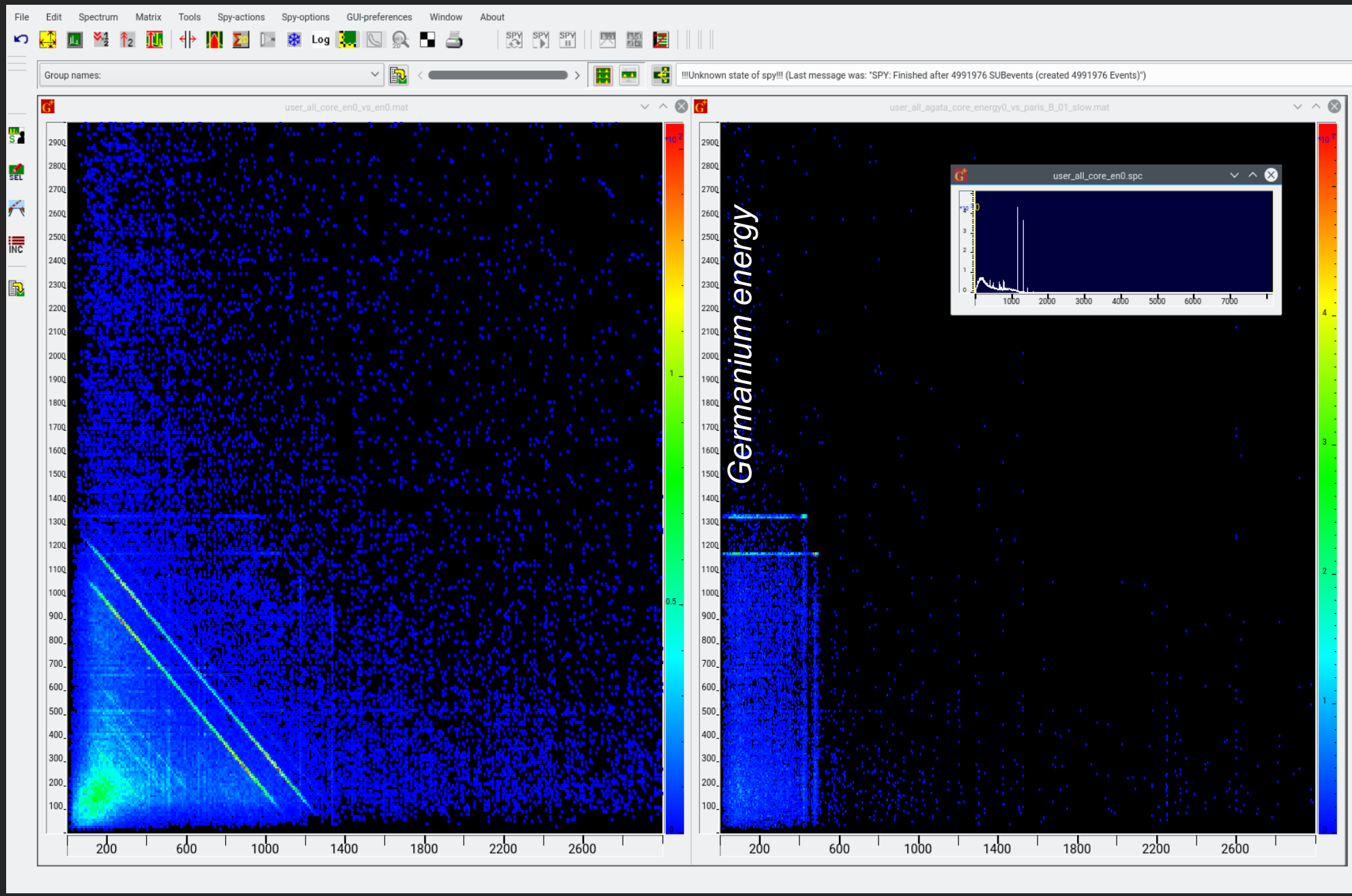
Analysis_timestamp_matrix_no_vamos.mat



Matrix X: 134.615 Y: 34.4024 cnt: 0 Analysis_timestamp_matrix.mat


greware GUI (on agata-analysis-1)
File Edit Spectrum Matrix Tools Spy-actions Spy-options GUI-preferences Window About
Group names:
!!!Unknown state of spy!!! (Last message was: "SPY: Finished after 8191844 SUBevents (created 8191843 Events)")
agata_00A_psa_hit_map_front.mat
agata_00A_psa_hit_map_side.mat
agata_00B_psa_hit_map_front.mat
agata_00B_psa_hit_map_side.mat
agata_00C_psa_hit_map_front.mat
agata_00C_psa_hit_map_side.mat
agata_01A_psa_hit_map_front.mat
agata_01A_psa_hit_map_side.mat
agata_01B_psa_hit_map_front.mat
agata_01B_psa_hit_map_side.mat
agata_01C_psa_hit_map_front.mat
agata_01C_psa_hit_map_side.mat
agata_02A_psa_hit_map_front.mat
agata_02A_psa_hit_map_side.mat
agata_02B_psa_hit_map_front.mat
agata_02B_psa_hit_map_side.mat
agata_02C_psa_hit_map_front.mat
agata_02C_psa_hit_map_side.mat
agata_03A_psa_hit_map_front.mat
agata_03A_psa_hit_map_side.mat
agata_03B_psa_hit_map_front.mat
agata_03B_psa_hit_map_side.mat
agata_03C_psa_hit_map_front.mat
agata_03C_psa_hit_map_side.mat
agata_04A_psa_hit_map_front.mat
agata_04A_psa_hit_map_side.mat
agata_04B_psa_hit_map_front.mat
agata_04B_psa_hit_map_side.mat
agata_04C_psa_hit_map_front.mat
agata_04C_psa_hit_map_side.mat
agata_05A_psa_hit_map_front.mat
agata_05A_psa_hit_map_side.mat
agata_05B_psa_hit_map_front.mat
agata_05B_psa_hit_map_side.mat
agata_05C_psa_hit_map_front.mat
agata_05C_psa_hit_map_side.mat

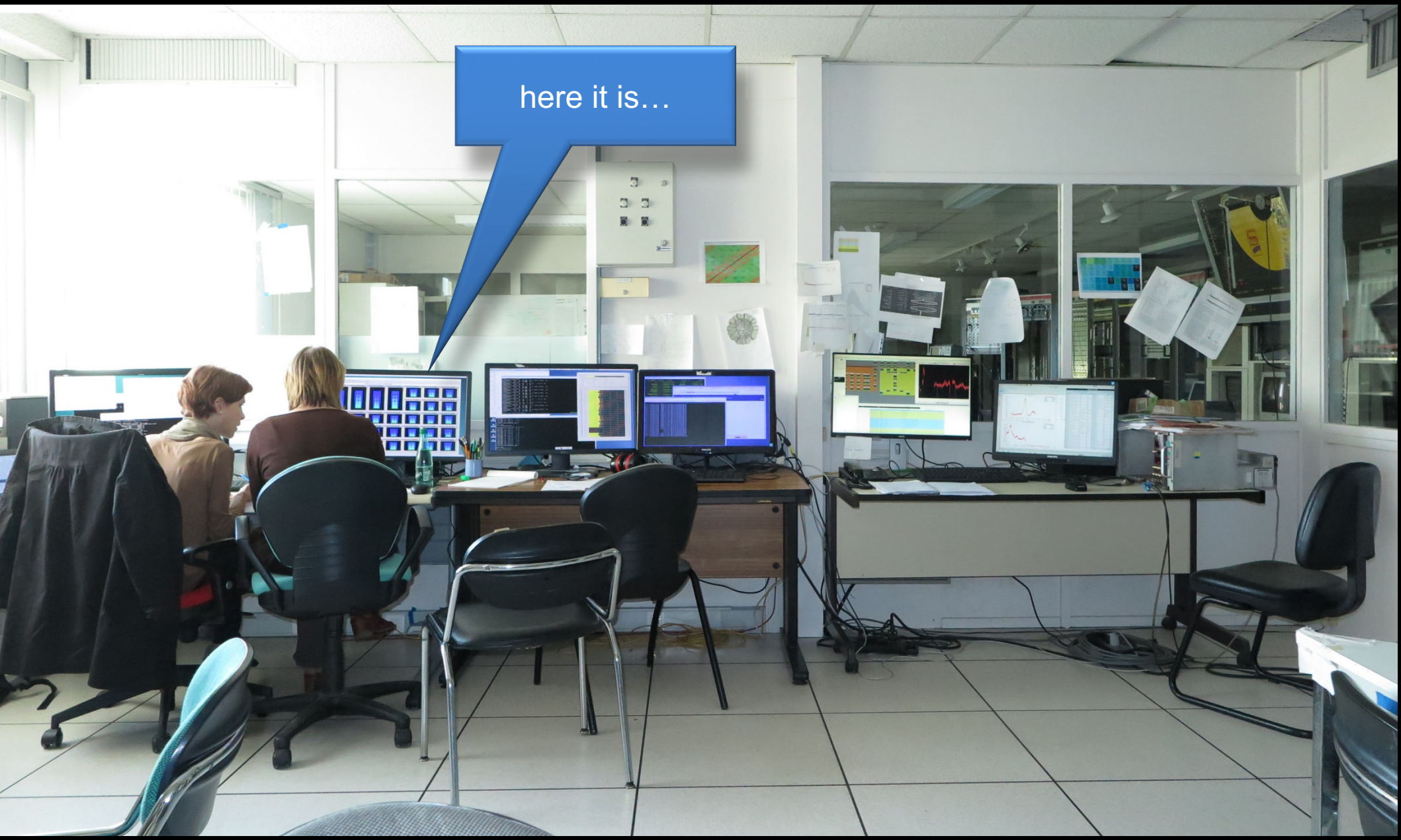
Does the matching of subevents work correctly?



Germanium energy

Germanium energy

Paris „slow” energy

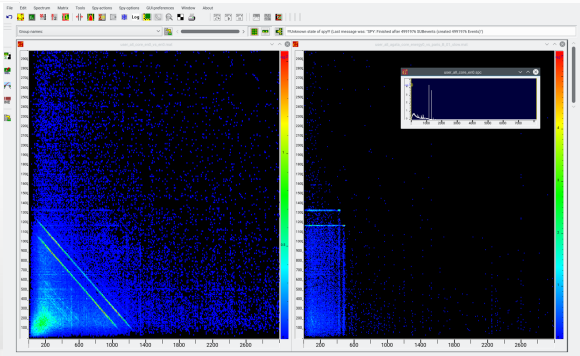

here it is…

Analysing online

To monitor ONLINE the currently collected run, it is enough to type

./spy -online

This will work only if there is a run currently opened by NARVAL.



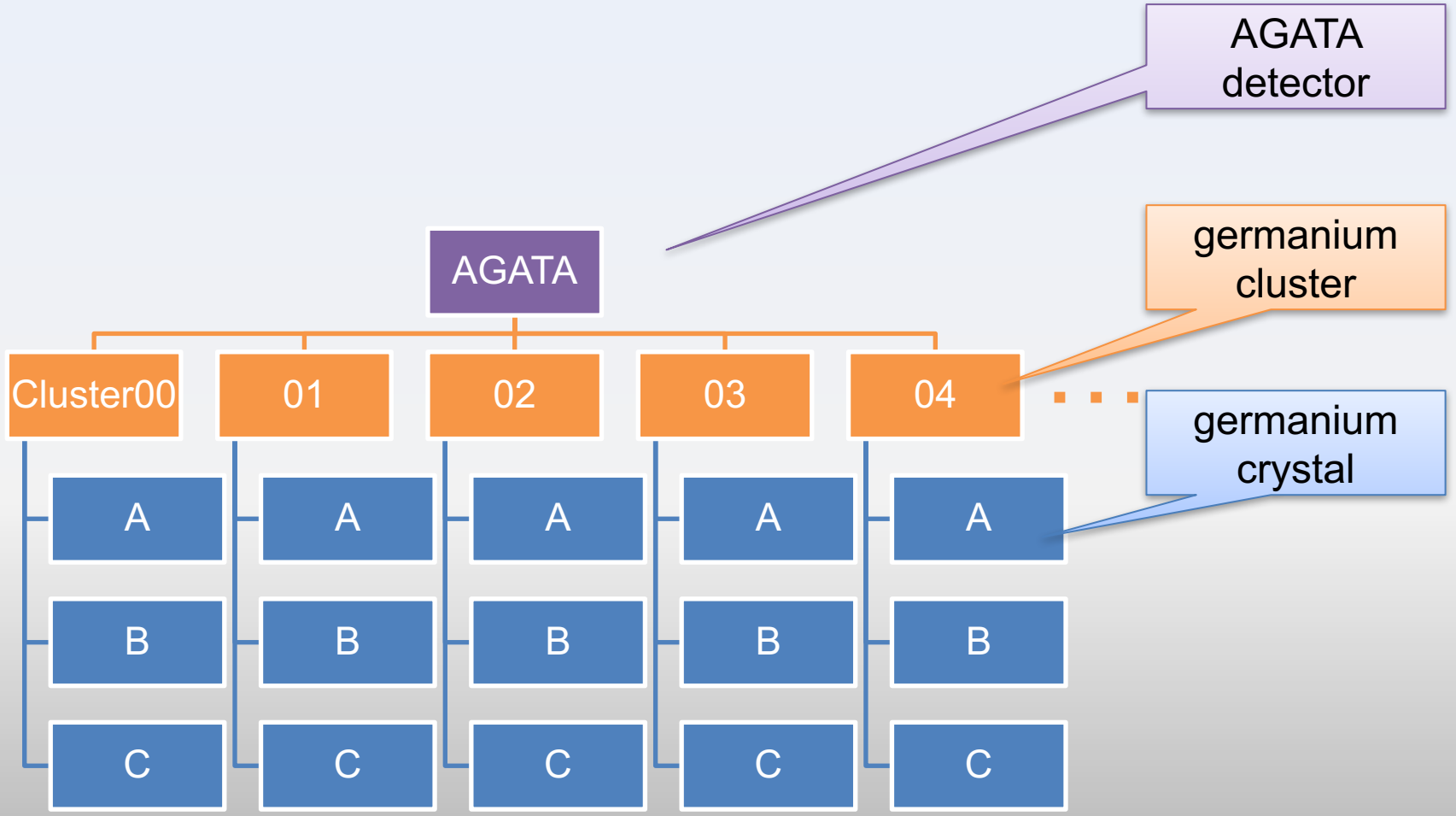
To analyse all the events collected during **current run** you type

./spy temporary_dir

This will work only if there is a run currently opened by NARVAL.
(if the run is already closed, it is available as normal run)

To build an (object oriented) analysis program –

means to build a software model of the experiment



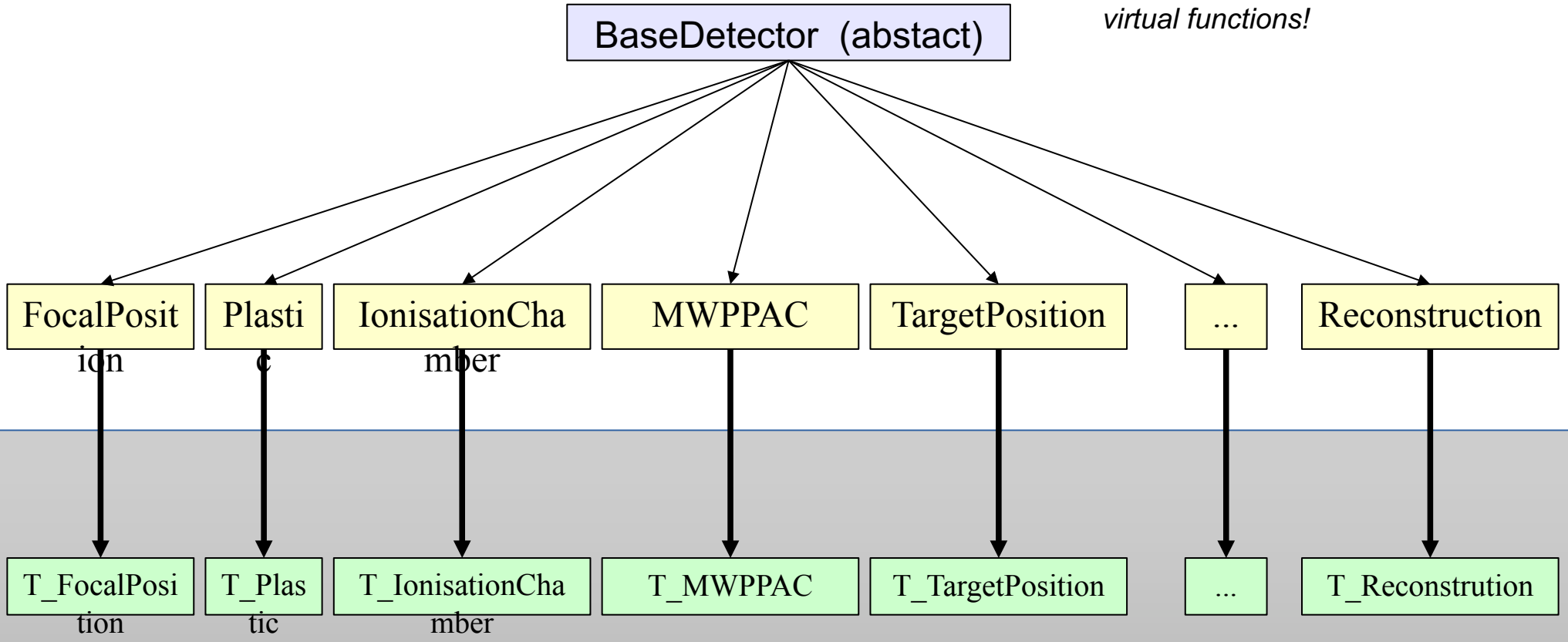
For example objects of AGATA

MFM Library

Antoine Lemasson

No CERN root

Classes:



Need to add:

* *default spectra*
* *incrementers*

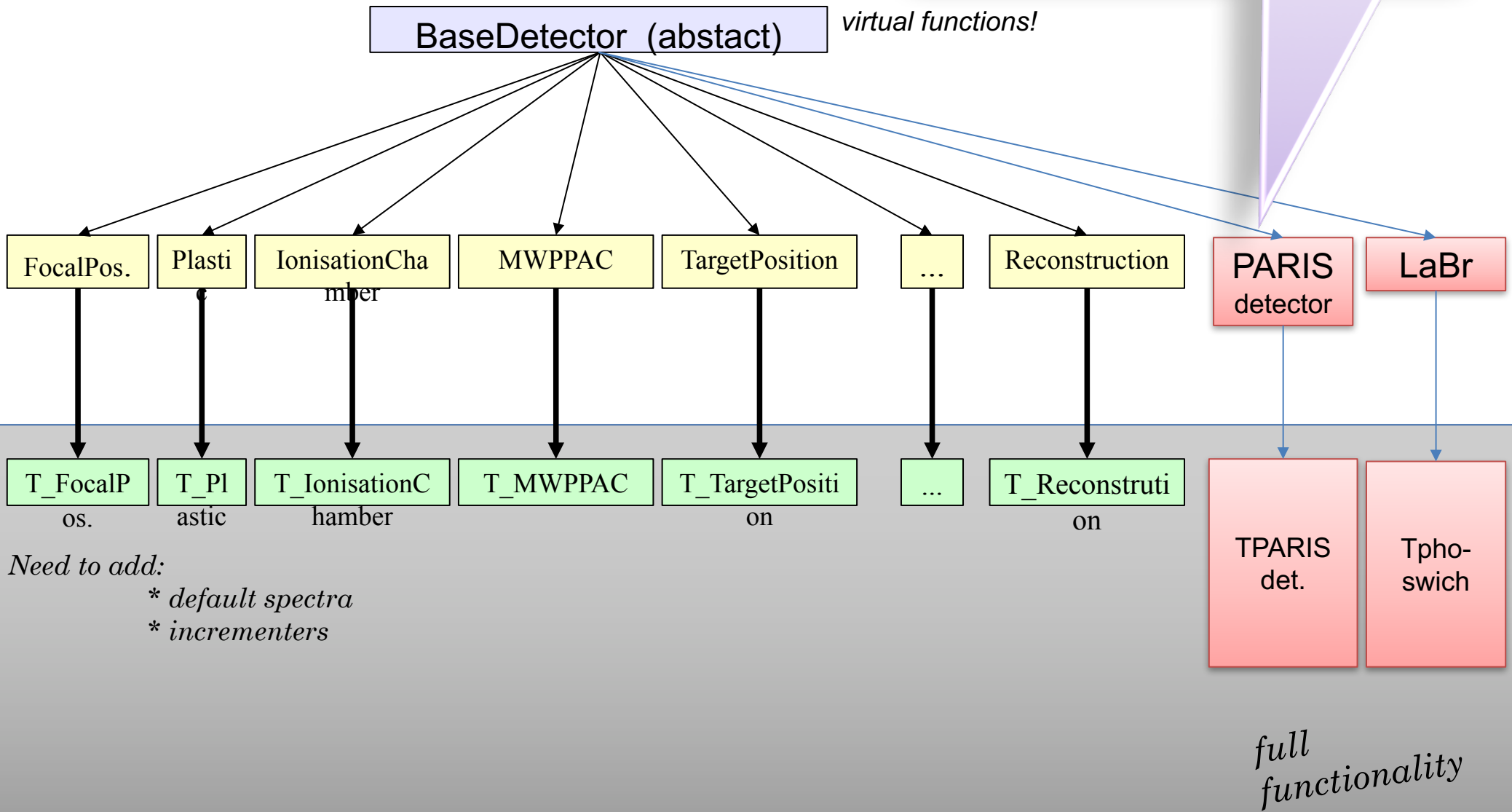
MFM Library

(No CERN root)

Antoine Lemasson

Classes:

Thanks to this, MFMlib will unpack Paris data for me



Need to add:

* *default spectra*
* *incrementers*

full functionality

„Analysis” – is something more, than just making a simple spectra of all possible signals

This would be a „monitoring”

We want to see some physics

Program variables – which are vital from a physicist point of view – I call:

Incrementers, because

- You can use them to increment your **spectra (or matrices)**
- You can use them to create your **conditions** (and conditional spectra)

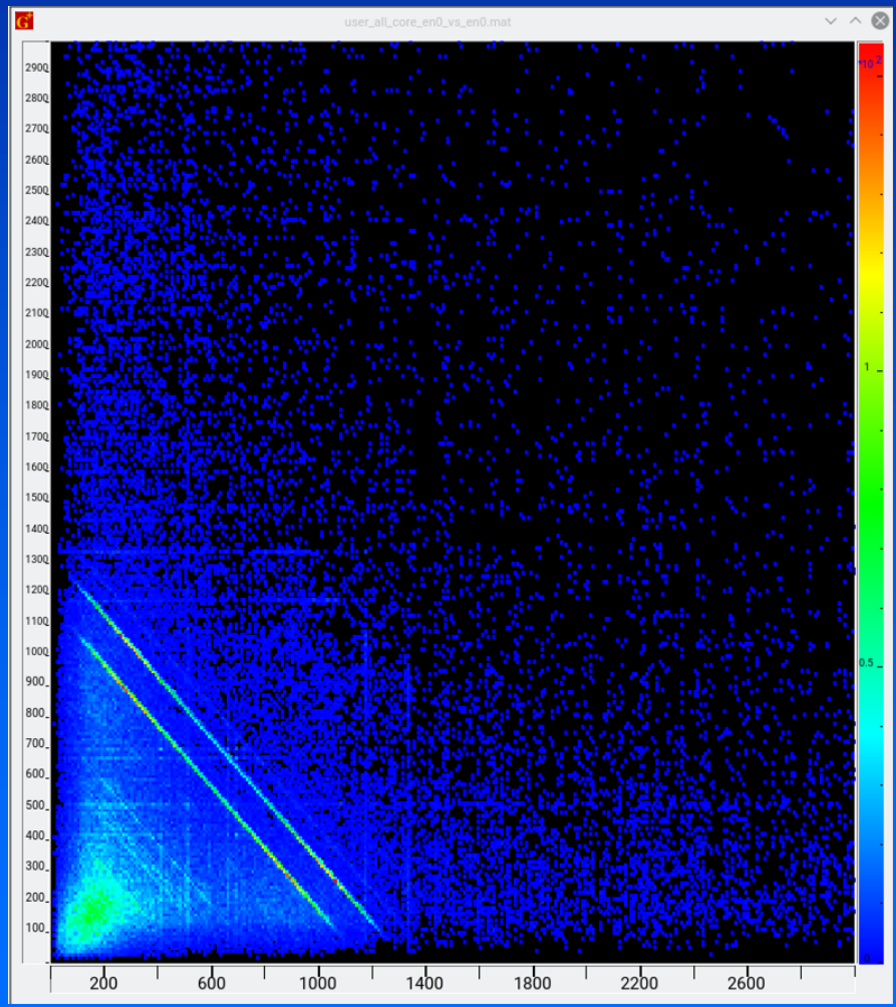
There are plenty of them in the program

Some incrementers available for AGATA crystals

agata_01A_core_energy0_when_fired
agata_01A_core_energy1_when_fired
agata_01A_core_time0_when_fired
agata_01A_core_time1_when_fired

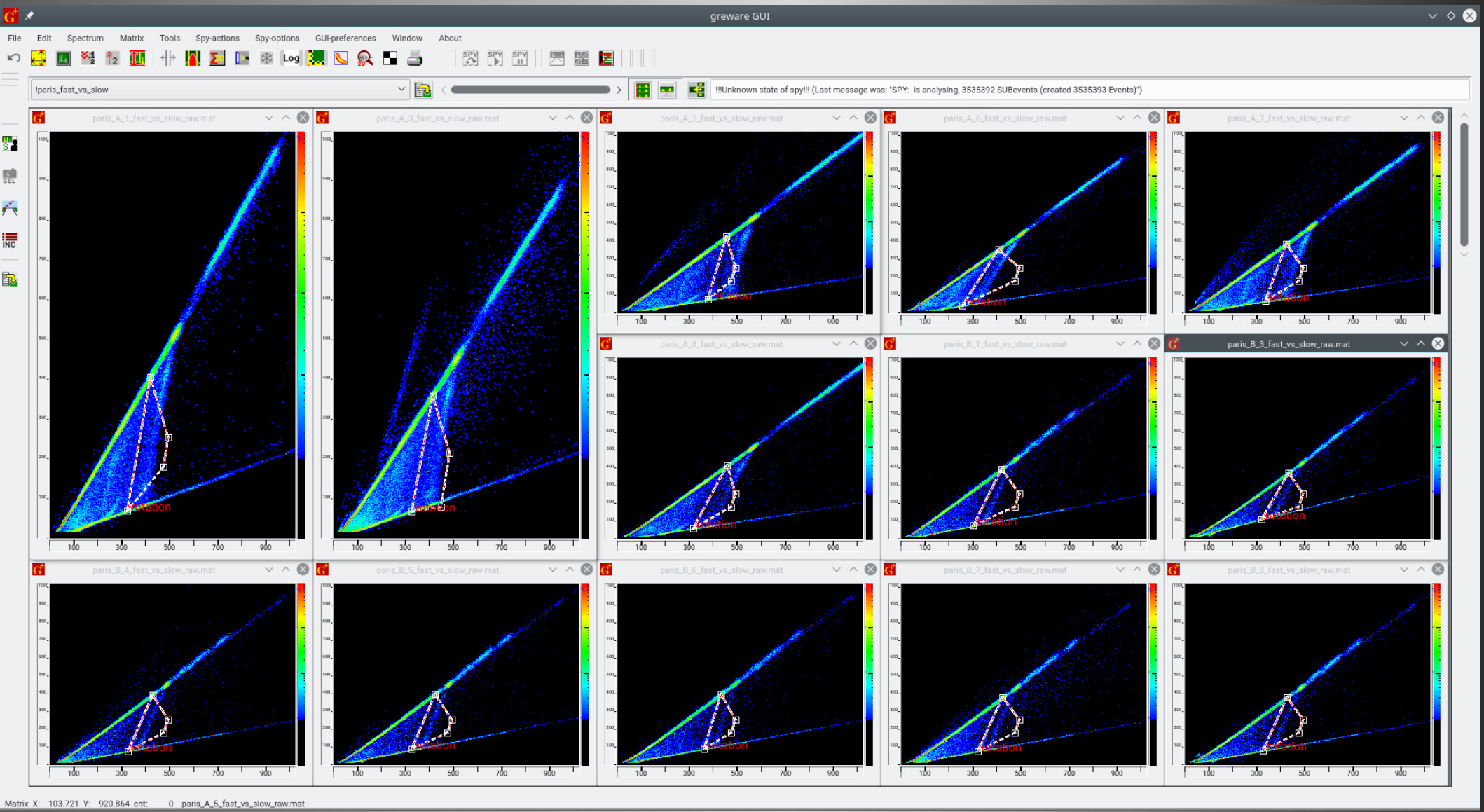
agata_01A_interaction_pt_x1_when_fired
agata_01A_interaction_pt_y1_when_fired
agata_01A_interaction_pt_z1_when_fired

× 28

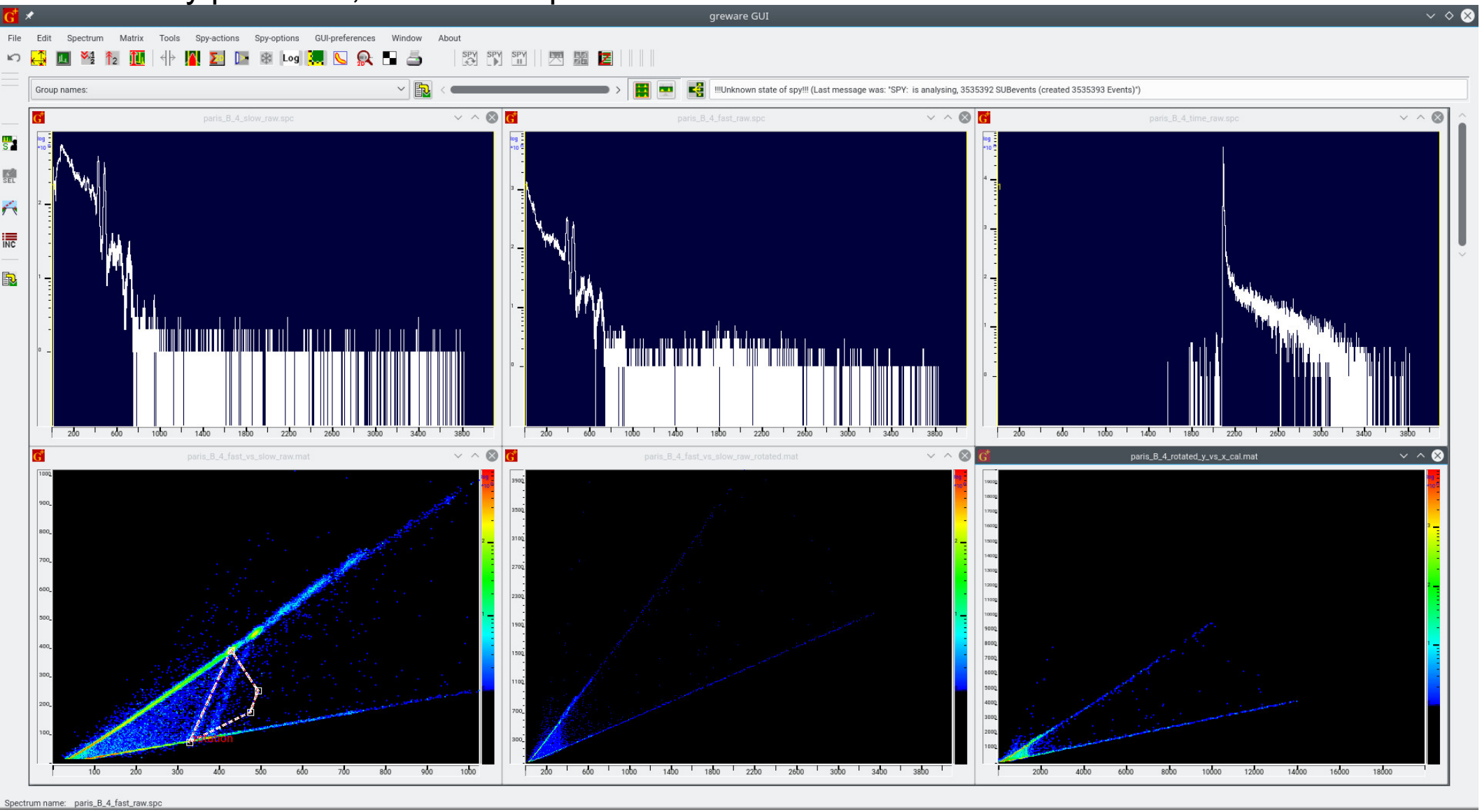


Incrementers to create TOTAL spectra

ALL_agata_core_energy0_when_fired
ALL_agata_core_energy1_when_fired
ALL_agata_core_time0_cal_when_fired
ALL_agata_core_time0_cal_when_fired


greware GUI
File
Edit
Spectrum
Matrix
Tools
Spy-actions
Spy-options
GUI-preferences
Window
About
!paris_fast_vs_slow
!!!Unknown state of spy!!! (Last message was: 'SPY: is analysing, 3535392 SUBevents (created 3535393 Events)')
paris_A_1_fast_vs_slow_raw.mat
paris_A_3_fast_vs_slow_raw.mat
paris_A_5_fast_vs_slow_raw.mat
paris_A_6_fast_vs_slow_raw.mat
paris_A_7_fast_vs_slow_raw.mat
paris_A_8_fast_vs_slow_raw.mat
paris_B_1_fast_vs_slow_raw.mat
paris_B_3_fast_vs_slow_raw.mat
paris_B_4_fast_vs_slow_raw.mat
paris_B_5_fast_vs_slow_raw.mat
paris_B_6_fast_vs_slow_raw.mat
paris_B_7_fast_vs_slow_raw.mat
paris_B_8_fast_vs_slow_raw.mat
Matrix X: 103.721 Y: 920.864 cnt: 0 paris_A_5_fast_vs_slow_raw.mat

elementary phoswich, his default spectra and his incrementers



paris_B_4_fast_raw
paris_B_4_slow_raw
paris_B_4_time_raw
…
paris_B_4_fast_cal
paris_B_4_slow_cal
paris_B_4_time_cal

Basic

paris_B_4_phi_degrees_when_fired
paris_B_4_theta_degrees_when_fired
paris_B_4_phi_radians_when_fired
paris_B_4_theta_radians_when_fired

Geometry

paris_B_4_rotated_x_when_ok
paris_B_4_rotated_y_when_ok

specific

How to analyse near-line (offline)?

You can start analysing data from a chosen run. Your runs you can see listed like this:

ls /agatadisks/e676/e676

run_0008.dat.04-07-17_19h42m59s
run_0016.dat.07-07-17_10h20m27s
run_0083.dat.10-07-17_17h45m38s
...
run_0104.dat.12-07-17_18h30m22s

If you want to analyse ("sort") the data from any particular run you need

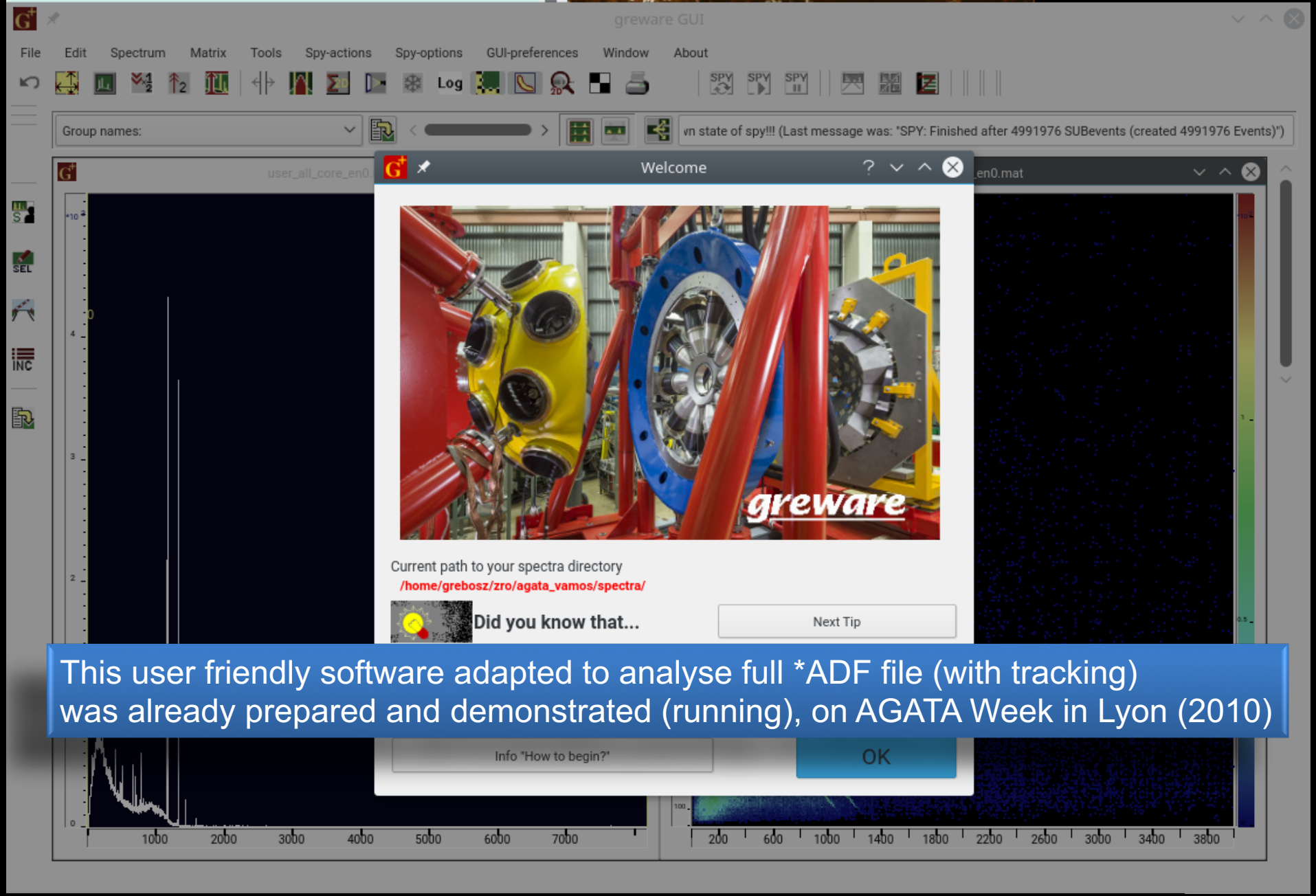
To start the spy you need a command

./spy [name of run directory]

For example, to analyse the run_0104.dat.12-07-17_18h30m22s - being int the directory /opt/data/GANIL/e676/greware/agata_vamos
you type:

./spy run_0104.dat.12-07-17_18h30m22s

However – without tracking…



Program for making ad hoc (!)

- User defined spectra,
- User defined conditions

(even very sophisticated), and remembering their definitions for future runs…

Thank you

```
//////////////////////////////////////////////////////////////////////////////
class TmyFocalPosition:
                        public FocalPosition, public Tincrementer_donnor, public
Tfrs_element
{
public:
                TmyFocalPosition(const Char_t *Name,
                                                        UShort_t Ndetectors,
                                                        Bool_t RawData,
                                                        Bool_t CalData,
                                Bool_t DefaultCalibration,
                                                                const Char_t
*NameExt);

                void create_my_spectra();
                void analyse_current_event();
                bool Treat();
                void make_preloop_action(ifstream &) {   }
                                        // read the calibration factors, gates
protected:
   spectrum_1D    *spec_xf, *spec_yf, *spec_tf, *spec_pf;
   spectrum_2D    *spec_xf_vs_yf, *spec_xf_vs_tf, *spec_pf_vs_tf,
//--------------------------*spec_y0_vs_y1, *spec_x0_vs_y3, *spec_x0_vs_y2;

   double          xf;                             // X focal Plane in mm
   double          yf;                             // Y focal Plane in mm
   double          tf;                                     // Theta focal
 Plane in mrad
   double          pf;                             // Phi focal Plane in mrad
   double          vamos_angle;            // Vamos Angle
```

5 incrementers in the class TfocalPlane

```
...

  named_pointer[name_of_this_element+"_x_in_mm"] = Tnamed_pointer(&xf, 0, this) ;

  named_pointer[name_of_this_element+"_y_in_mm"] = Tnamed_pointer(&yf, 0, this) ;

  named_pointer[name_of_this_element+"_theta_in_mrad"] = Tnamed_pointer(&tf, 0, this) ;

  named_pointer[name_of_this_element+"_phi_in_mrad"] = Tnamed_pointer(&pf, 0, this) ;


  named_pointer[name_of_this_element+"_vamos_angle"] =
                                                  Tnamed_pointer(&vamos_angle, 0, this) ;

...
```

Unpacking AGATA data

========== NEXT EVENT ==========

event:data ca000100

- data:tracked fa010105 - AGATA

  Energy= 355.717 XYZ1(72.4254, -144.66,-191.23) T1= 48.092 SegmentId 13 CoreId =44
  Energy= 305.232 XYZ1(-113.758, 29.5191,-255.077) T1= 39.6315 SegmentId 3 CoreId =7
  Energy= 540.04 XYZ1(52.0824, 50.8962,-267.76) T1= 58.2282 SegmentId 15 CoreId =3

- event:data:psa ca010102

  - data:psa fa010102
  - data:psa fa010102
  - data:psa fa010102

- data:ranc0 fa0201a0 → unpacking the 'data_ranc0' frame - VAMOS

========== NEXT EVENT ==========

Dear Andres,

I am writing to you, because I am not sure if my contribution would be interesting to the AGATA week participants.

Just now I returned from Ganil where we had an experiment AGATA+Paris detector. Adam May, being in Ganil some months ago asked me to prepare an online analysis for the coming experiment.

He wanted to have the Paris + AGATA data online on the screen without necessity of making so called replay, root trees, etc.

So I designed the spy program which accepts the data from PSA actors, makes spectra, and makes all this, what you, Andres, know with the name "Cracow Viewer", and what was fully rewritten and now it is known as "Greware".

/*...*/

The spy was designed to take the data (through sockets) from psa actors - so still before event building.
This is why it builds the events looking at timestamps:

1 timestamp of Vamos data
28 timestamps of particular Agata crystals.

Just in the beginning the experiment, 2 weeks ago I tested this, and had a lot of problems with accessing actors with sockets. I had a help from Oliver, and from local Ganil people, and finally, conclusion is that the socket connection is unreliable.
So I changed the tactics: PSA actors write their data on disk (every minute), so I immediately read the new information which actors just wrote, and make the analysis.
It works nicely.
(After this we are able to see spectra, use the incrementers to make conditional spectra, etc, etc.)