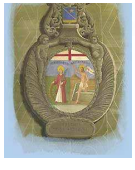
User libraries: the algorithms in the data flow

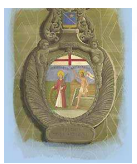
D. Mengoni

Università e Sezione INFN di Padova, Italia

Agata Week, Köln - Germany
March 30th ÷ April 1st, 2009

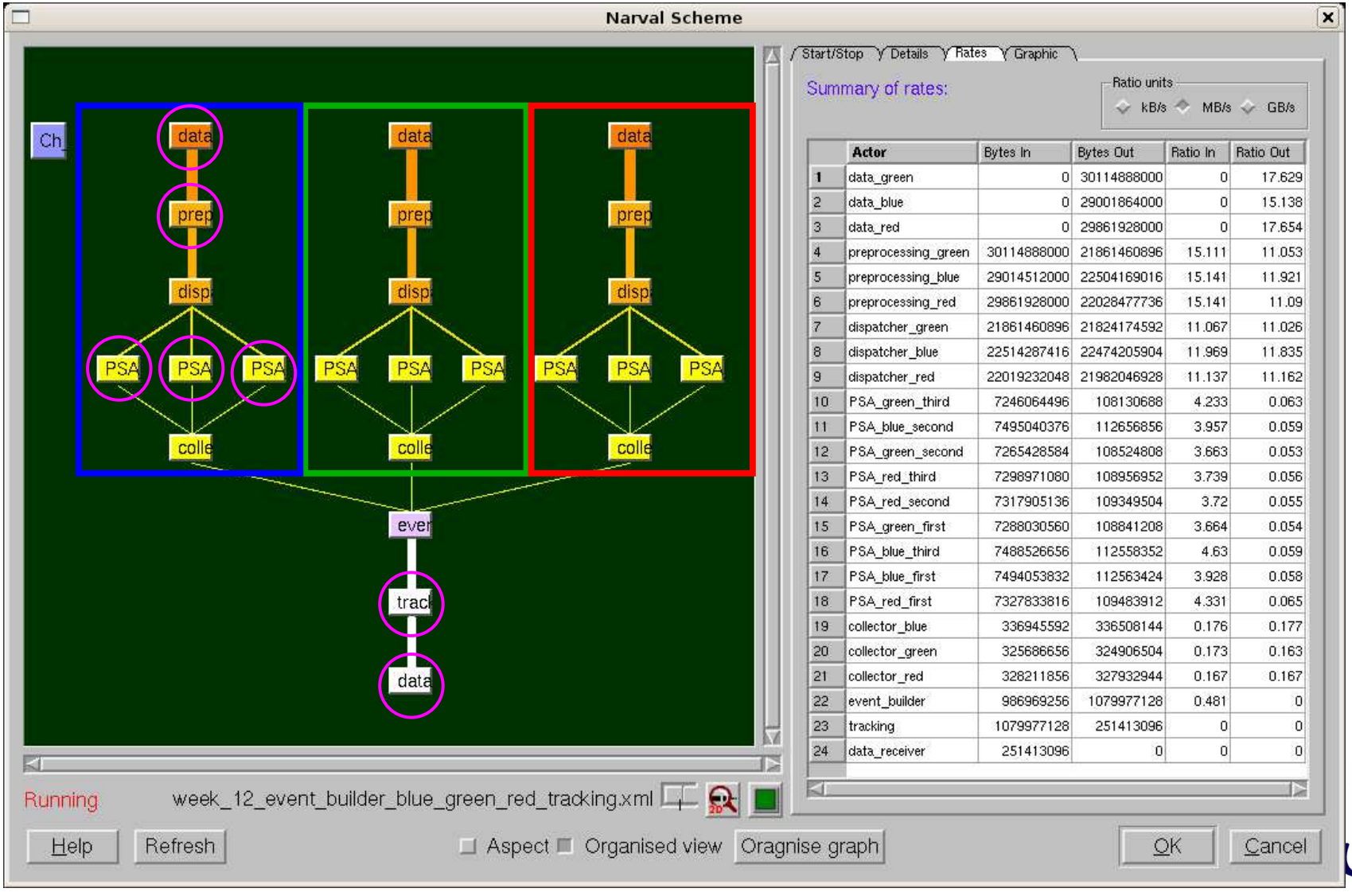


Outline



In-beam experiment

where are the user libraries



	Actor	Bytes In	Bytes Out	Ratio In	Ratio Out
1	data_green	0	30114888000	0	17.629
2	data_blue	0	29001864000	0	15.138
3	data_red	0	29861928000	0	17.654
4	preprocessing_green	30114888000	21861460896	15.111	11.053
5	preprocessing_blue	29014512000	22504169016	15.141	11.921
6	preprocessing_red	29861928000	22028477736	15.141	11.09
7	dispatcher_green	21861460896	21824174592	11.067	11.026
8	dispatcher_blue	22514287416	22474205904	11.969	11.835
9	dispatcher_red	22019232048	21982046928	11.137	11.162
10	PSA_green_third	7246064496	108130688	4.233	0.063
11	PSA_blue_second	7495040376	112656856	3.957	0.059
12	PSA_green_second	7265428584	108524808	3.663	0.053
13	PSA_red_third	7298971080	108956952	3.739	0.056
14	PSA_red_second	7317905136	109349504	3.72	0.055
15	PSA_green_first	7288030560	108841208	3.664	0.054
16	PSA_blue_third	7488526656	112558352	4.63	0.059
17	PSA_blue_first	7494053832	112563424	3.928	0.058
18	PSA_red_first	7327833816	109483912	4.331	0.065
19	collector_blue	336945592	336508144	0.176	0.177
20	collector_green	325686656	324906504	0.173	0.163
21	collector_red	328211856	327932944	0.167	0.167
22	event_builder	986969256	1079977128	0.481	0
23	tracking	1079977128	251413096	0	0
24	data_receiver	251413096	0	0	0

Libraries
What are they really?

C/C++ code

- The BASE CLASS usually binds with the ADF: I/O methods, configuration, initialization
- The DERIVED CLASS usually overloads the processing method where the job is done
- (symbols for Narval interface)

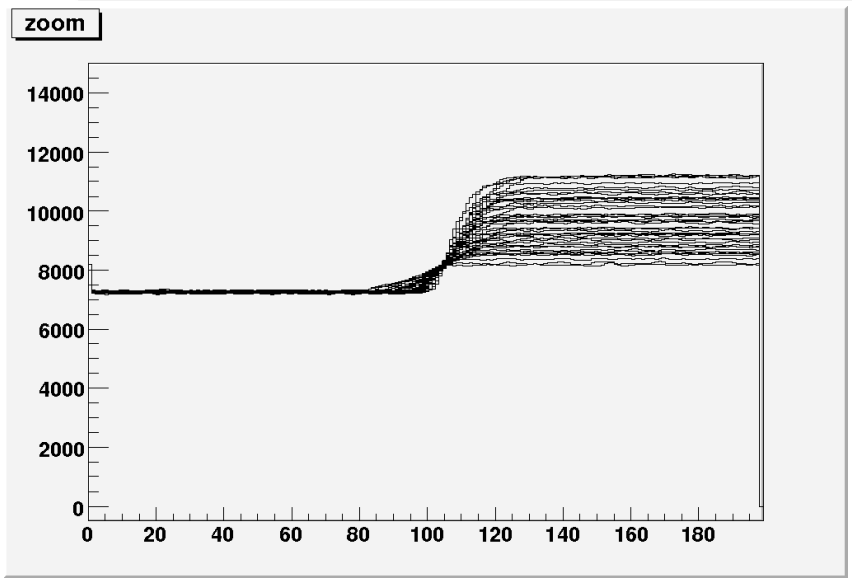
ORGANIZATION INSIDE NARVAL

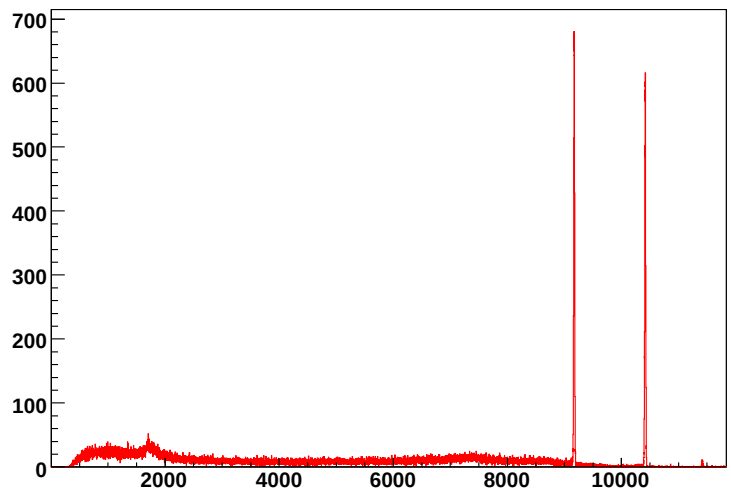
All the C/C++ code is in a dedicated machine where all the libraries have been ported and build. Afterwards they are copied to a common nfs directory, loaded and configured inside NARVAL.

Mezzanine producer
data reading from disk/carrier

Data import inside the DAQ

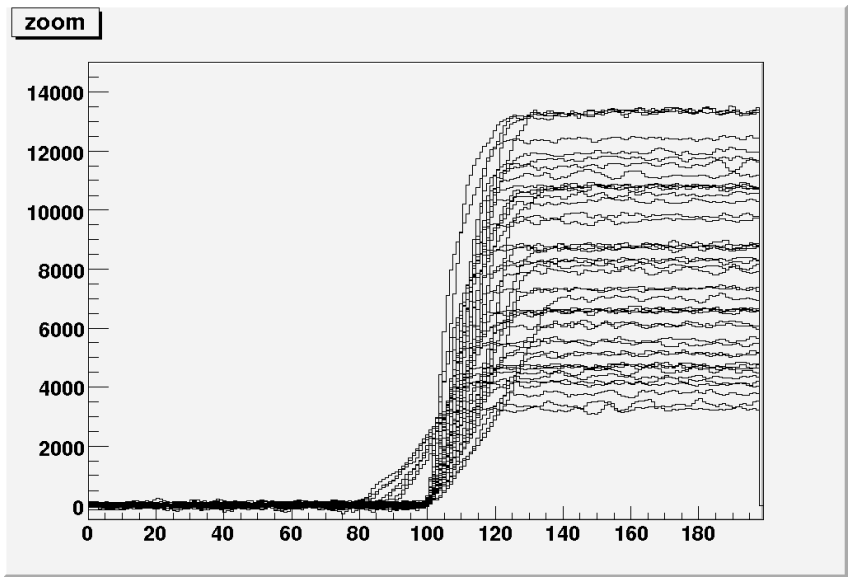
- Input:Disk/Carrier reading and mezzanine decoding
- Output: event Crystal Frame conversion
- Data compliant with ADF 2.0: 12.648 kB/ev

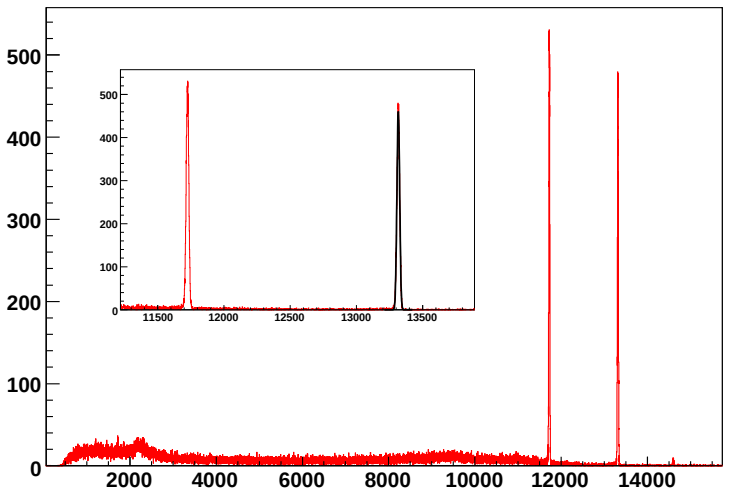




PSA preprocessing
ADF crystalFrame

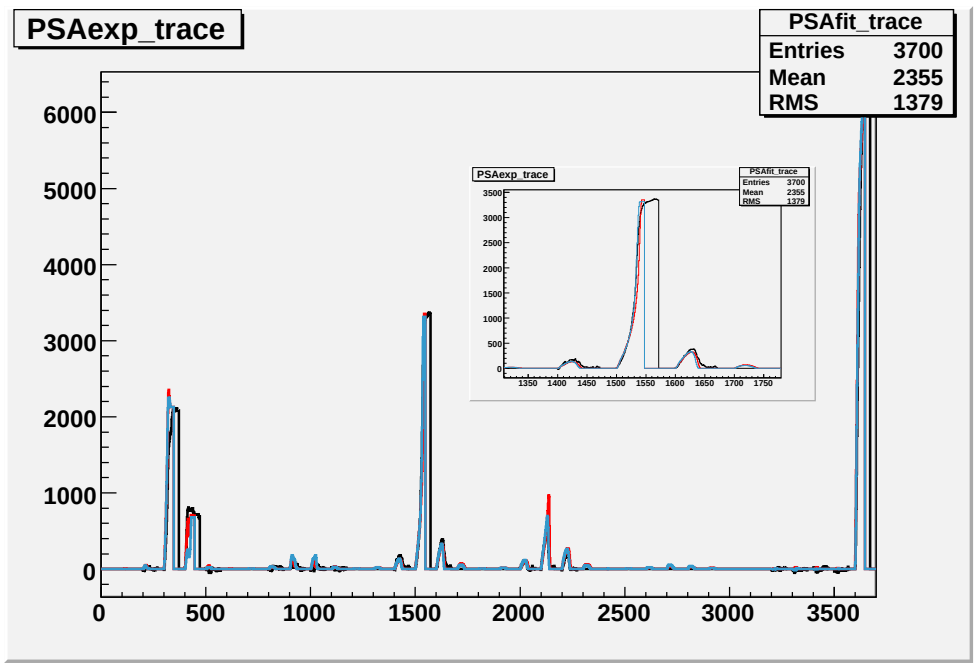
- Energy calibration and integral cross talk correction
- Time normalization, baseline and offset removal, alignement.

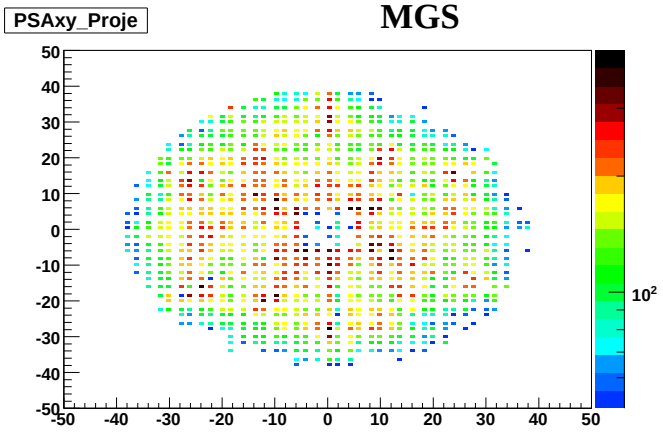


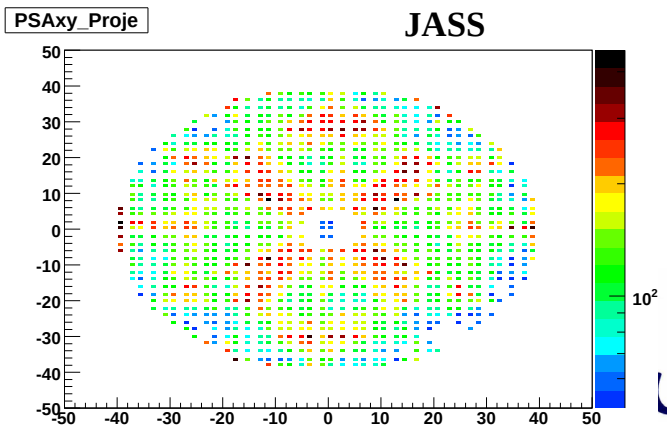


PSA
ADF psaFrame

- Simple grid search: JASS and MGS basis (2 mm, 5 ns).
- Comparison: different exp point in the same segment.







The emulators

How to debug the algorithms

... for libraries it looks Narval but ...

- It runs locally: no distributed system
- Currently only one detector at once

Nevertheless it has next advantages:

- Code development using standard tools: **debugger**, **profiling**, **leakage**, etc.
- Porting of the code is straightforward

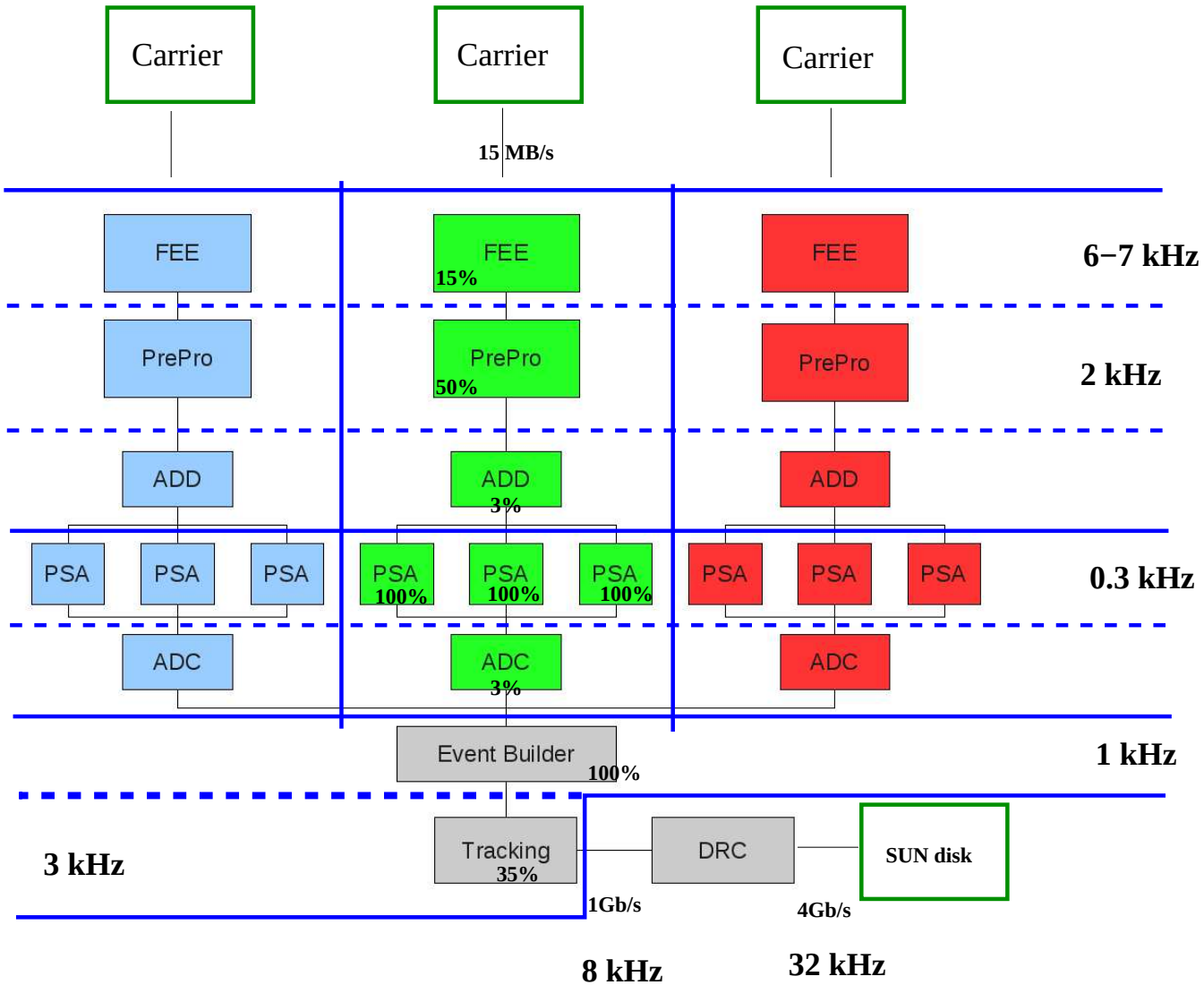
Therefore ...

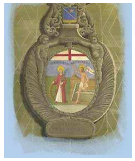
use one of the emulators already available or Narval print-outs

INFN

Experiment Data rate $\gtrsim$ 1 kHz per crystal

How do we proceed with PSA,EB,ADF,DISK?



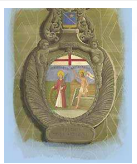


Summary and conclusions

- ON-LINE analisys at a rate of $\gtrsim$ 1 kHz
- Data replaying at almost the same rate.
- At the moment PSA is the bottle-neck.

TO DO/ ON GOING

- Presort of data $\rightarrow$ Data quality check: who?
- Improvement in the algorithms and (maybe) faster optic link (up to 4 Gb/s) between narvaldsx and the farm.
- Balance between efficiency and flexibility of the data stream
- Merger for future ancillary?



INFN