



*4th AGATA-GRETINA/GRETA Tracking Arrays Collaboration Meeting,
November 2024 - Argonne National Laboratory*

AGATA computing model

O.Stézowski
and the Data Processing Group

Many thanks to the Data Processing Team

G.Baulieu, Ch. Bonnin, N.Dosme, J.Dudouet, S. Elloumi, Ph. Gauron, A. Goasduff, M.Gulmini,
A. Korichi, J. Jacob, V. Lafage, E. Legay, P. Lejeannic,
J. Ljungvall, G.Philippon, R.Molina, M. Roetto, M. Tauriga-Quere, N.Toniolo

Outlines

- Current Data Processing Model
 - Limitations
 - What is to be improved ?
- Phase 2 Data Processing model
 - New electronics = new data pipeline
 - New monitoring
 - New architecture
- Conclusions

Current Data Processing Model

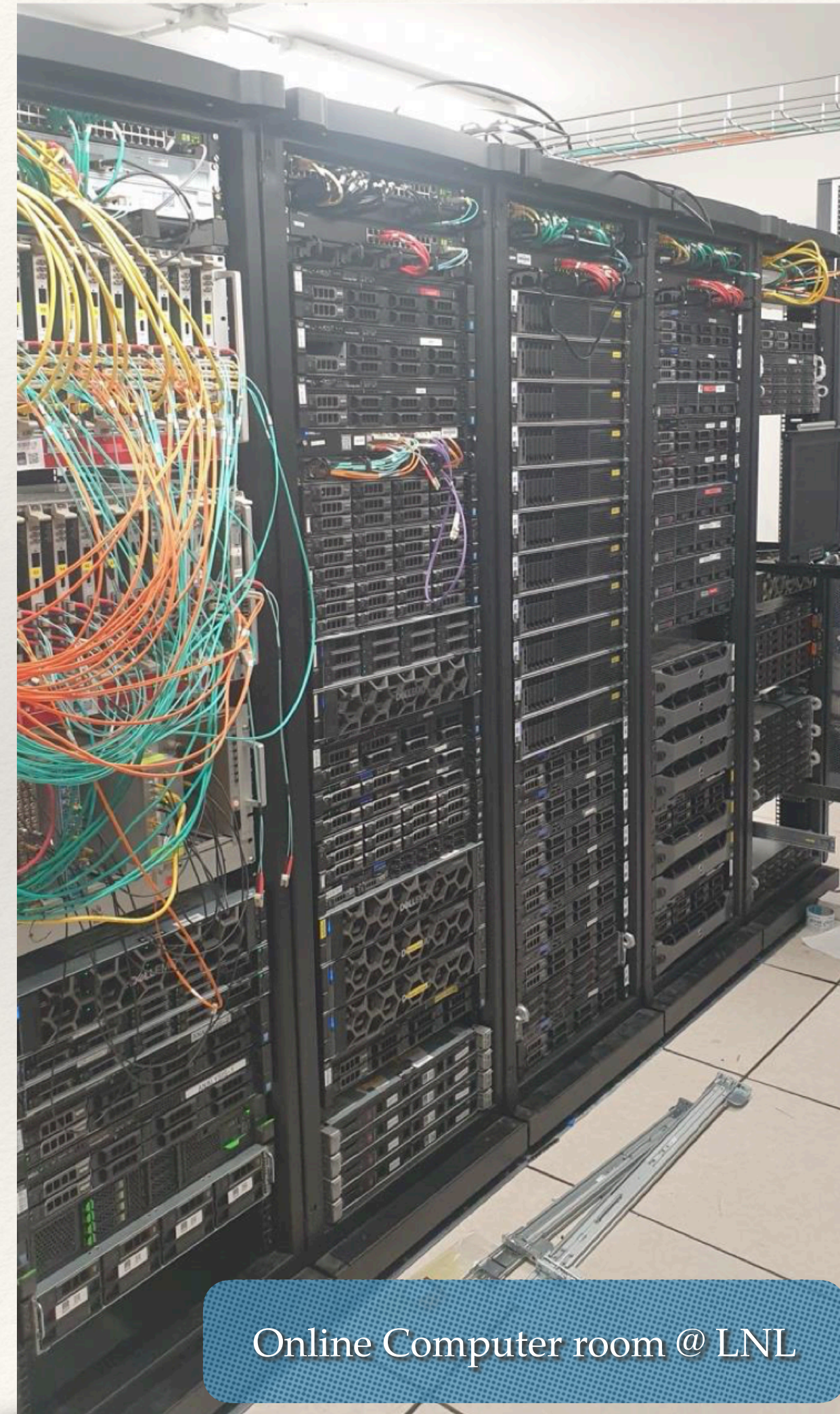


Detector room

+ ancillaries

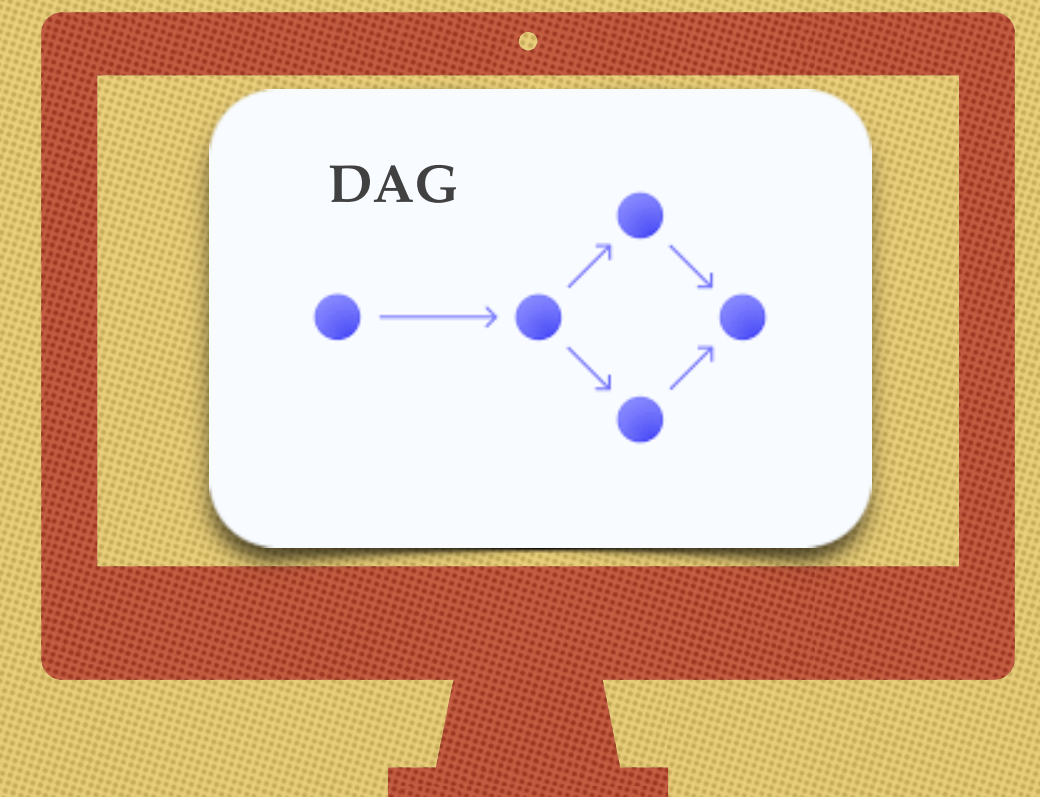
To run AGATA we do need computing power

Many communication channels



Online Computer room @ LNL

We need to run a processing graph



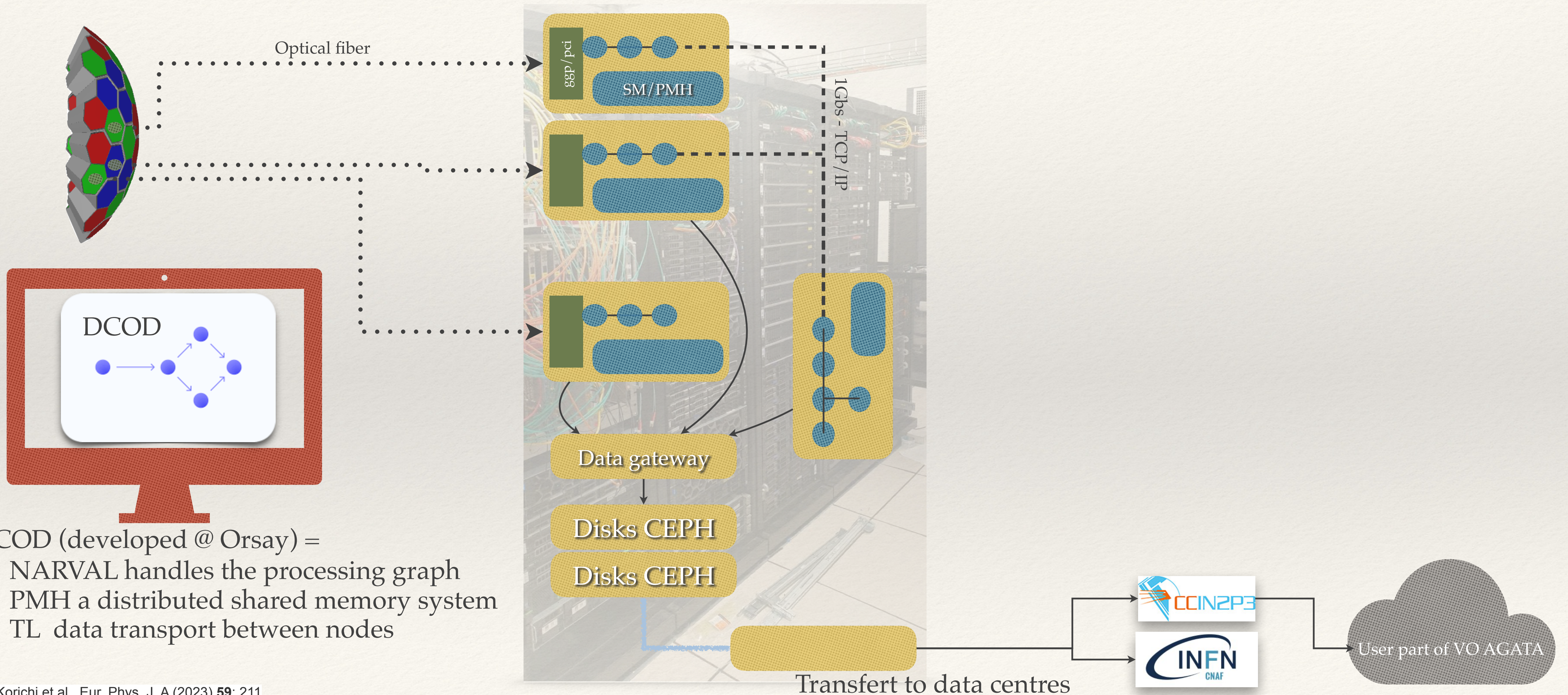
We have resources for online

We don't have dedicated for offline
→ a standalone program for replay

AS MUCH AS POSSIBLE
online/offline shared same code ...

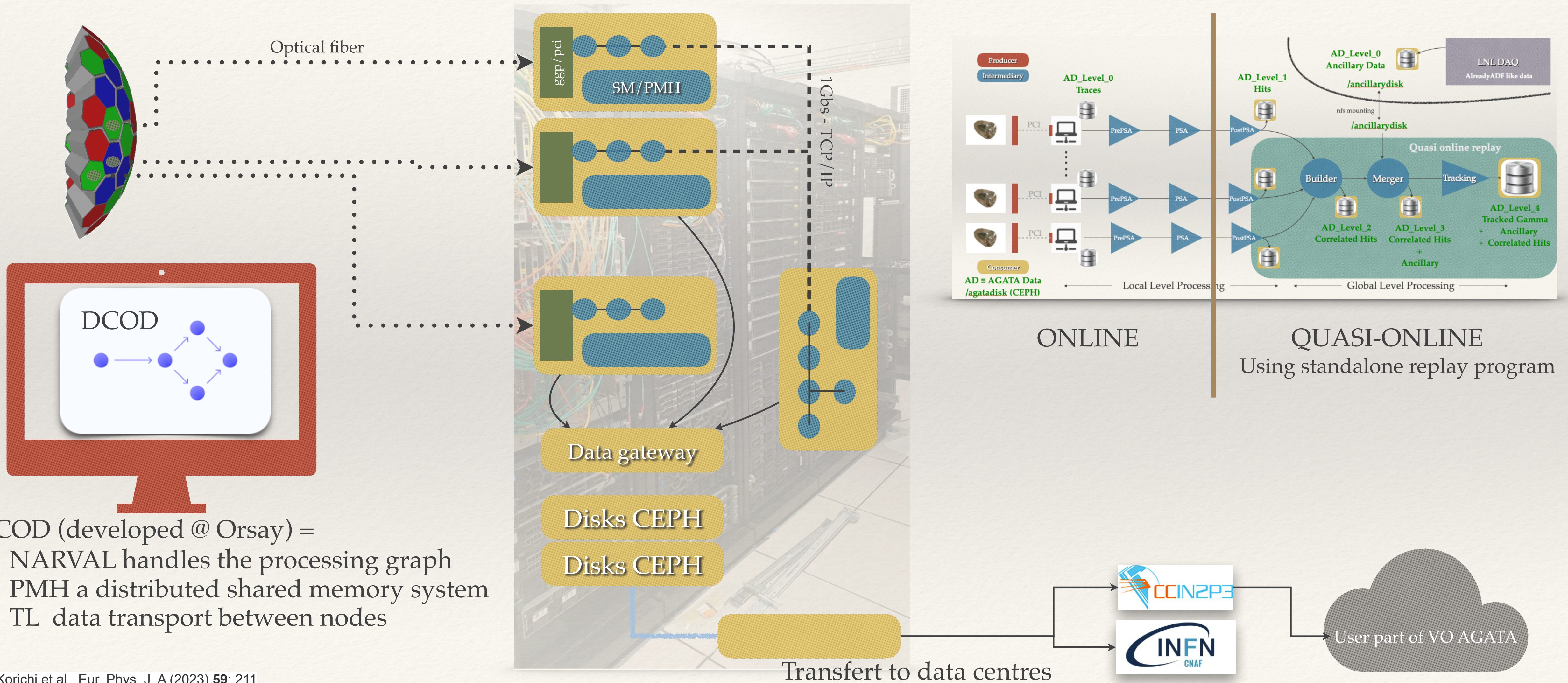
This presentation is about the data channel

Current Data Processing Model - Online



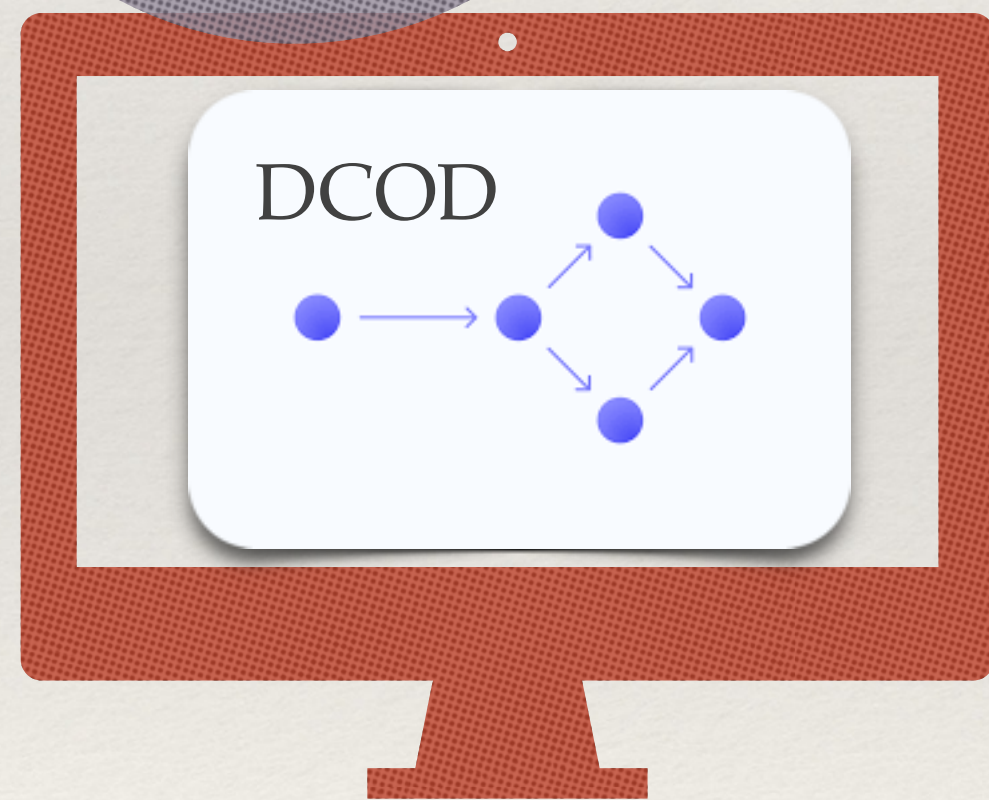
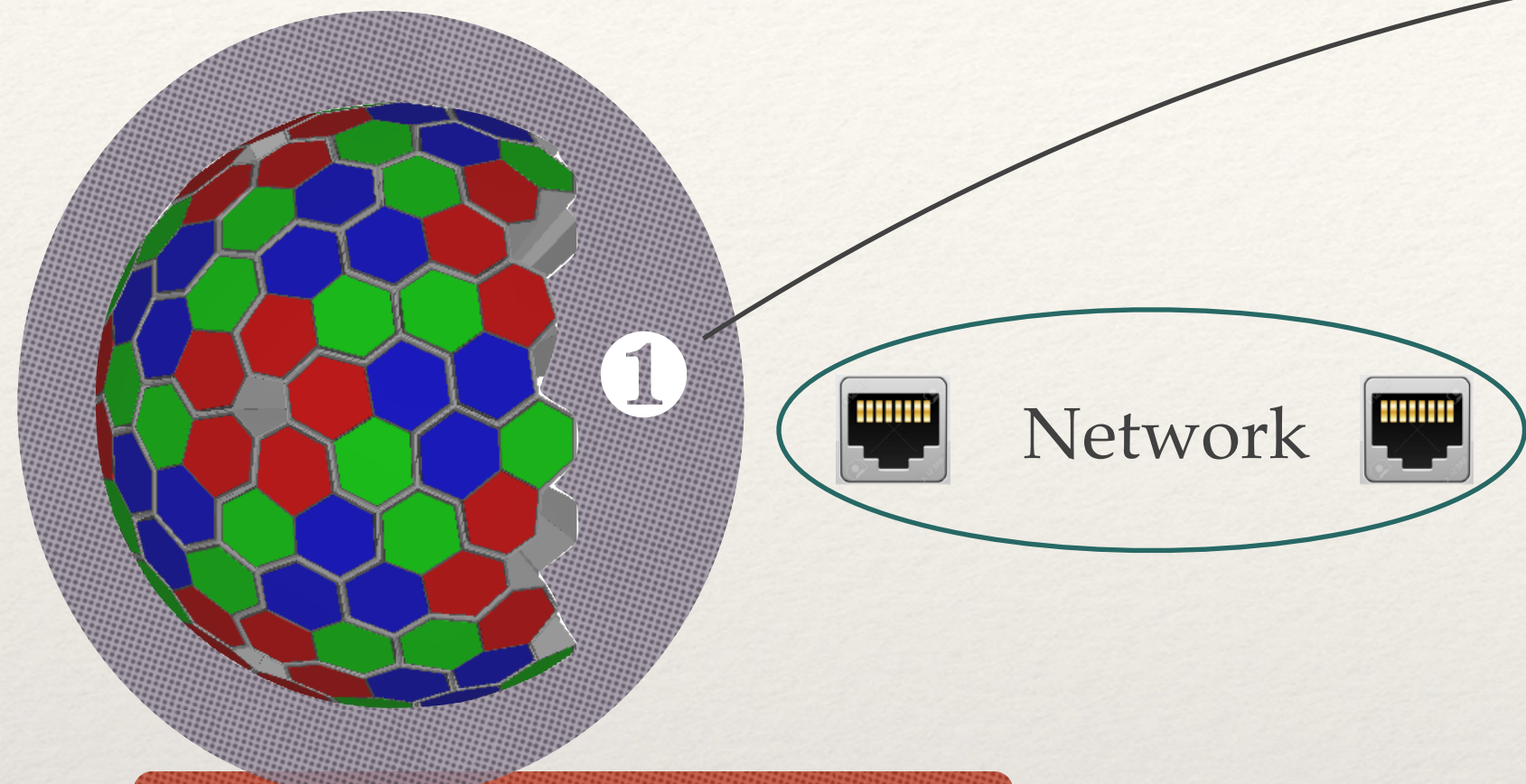
DCOD (developed @ Orsay) =
 NARVAL handles the processing graph
 PMH a distributed shared memory system
 TL data transport between nodes

Current Data Processing Model - Online

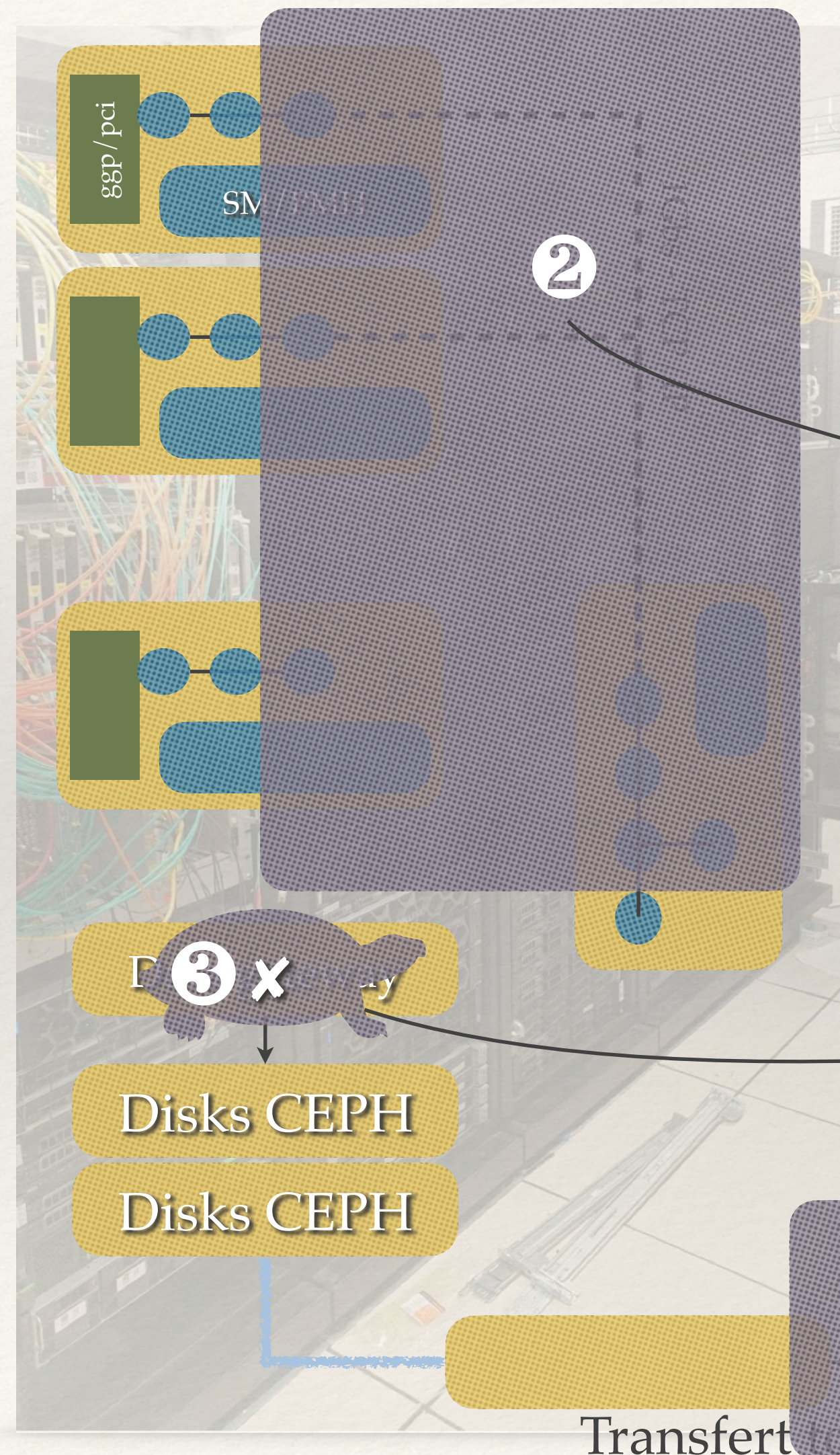


DCOD (developed @ Orsay) =
 NARVAL handles the processing graph
 PMH a distributed shared memory system
 TL data transport between nodes

Future Data Processing Model - What to be changed



DCOD (developed @ Orsay) =
 NARVAL handles the processing graph
 PMH a distributed shared memory system
 TL data transport between nodes



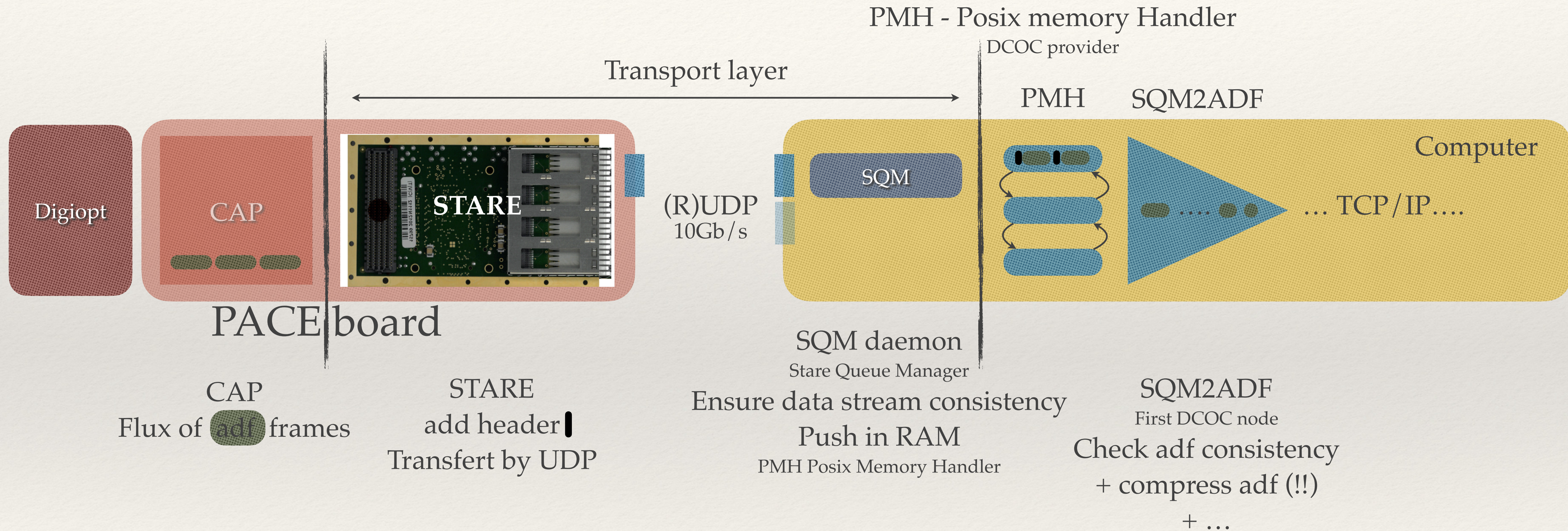
Not scalable : new v2 electronics
 → new data pipeline
 → higher rates **1**

New PSA/Tracking algorithms
 → more computing power
 [heterogeneous hardware ex : GPU]
 → more 'efficient' processing & monitoring
 → scalable offline replay (single pc, cluster, ...)

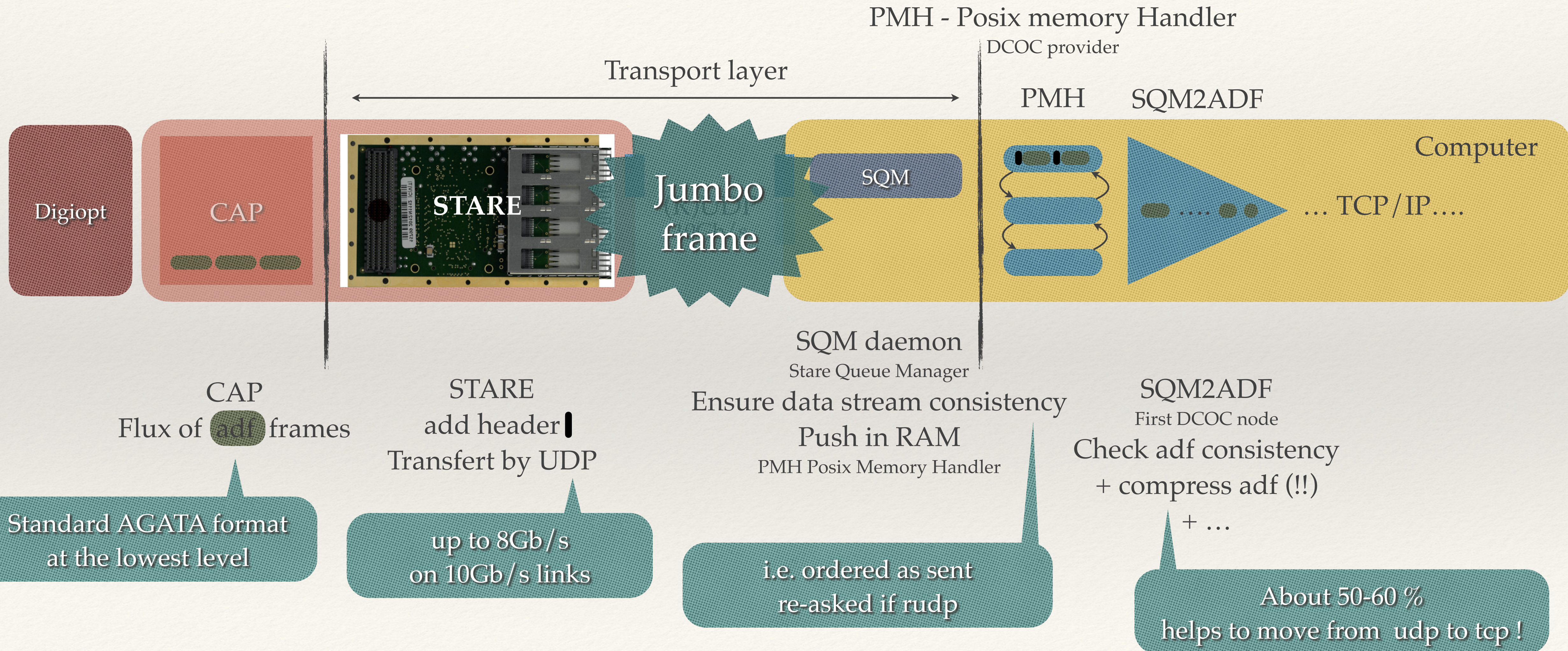
Bottleneck to be removed, cephfs toward
 (required recent operating system) **3**

Better Data Management [DMP, FAIR approach] **4**

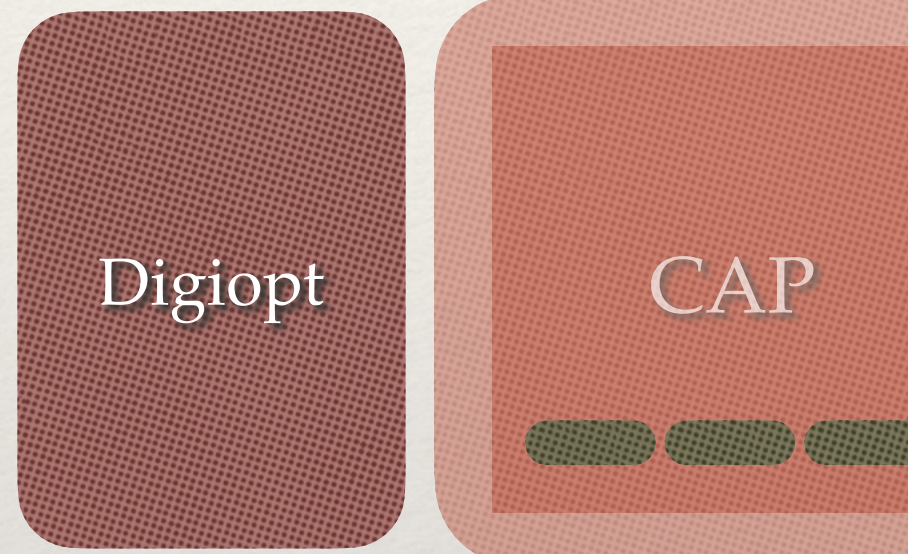
① V2 electronics - New data pipeline



① V2 electronics - New data pipeline



① V2 electronics - New data pipeline



All the components almost completed
+ tested in various environments / configuration
Full integration with real electronic boards still to be realised

BUT PACE emulator [CPU] developed, able to run @ up to 8Gb/s

With recent computers, chain validated up to 50kHz*
in fact more but 50 kHz is our upper limit - except PSA - for developments
rudp is a real advantage in having a reliable data stream

Standard AGATA format
at the lowest level

CAP
Flux of **adf** frames

up to 8Gb/s
on 10Gb/s links

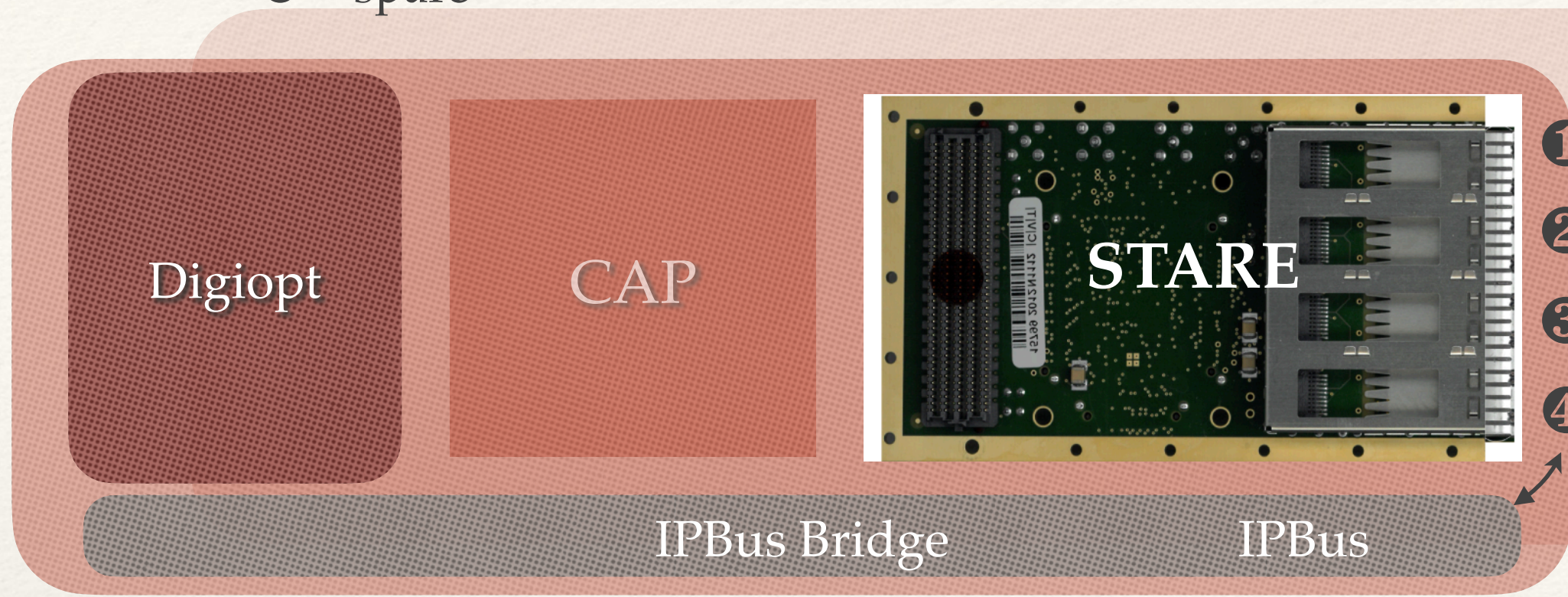
re-asked if rudp

* 50 kHz = 400 Mb/s = 3.2 Gb/s

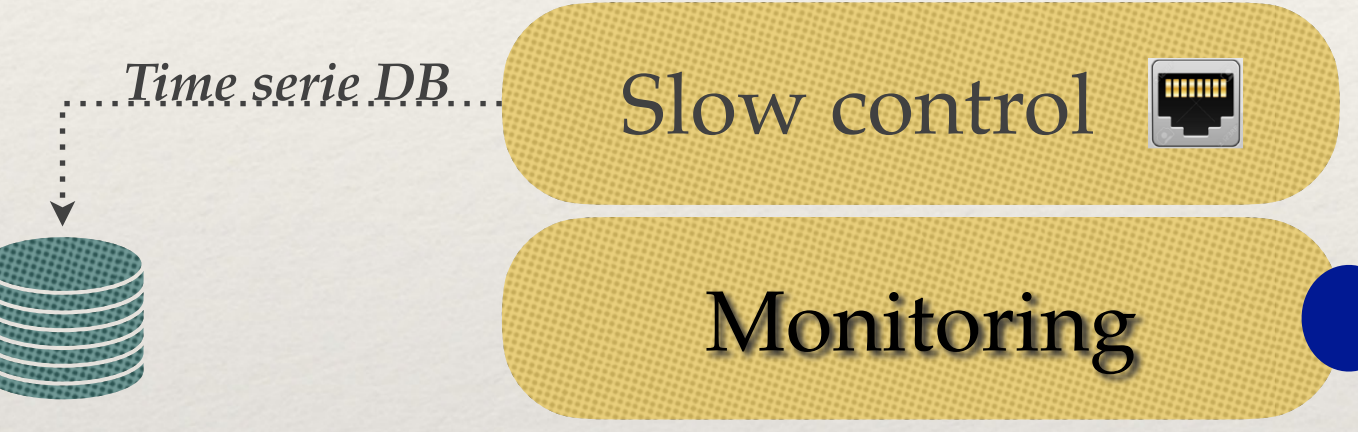
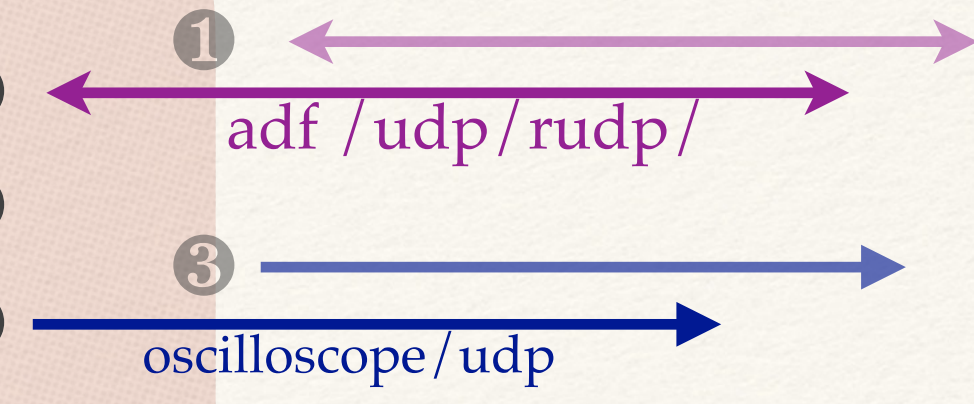
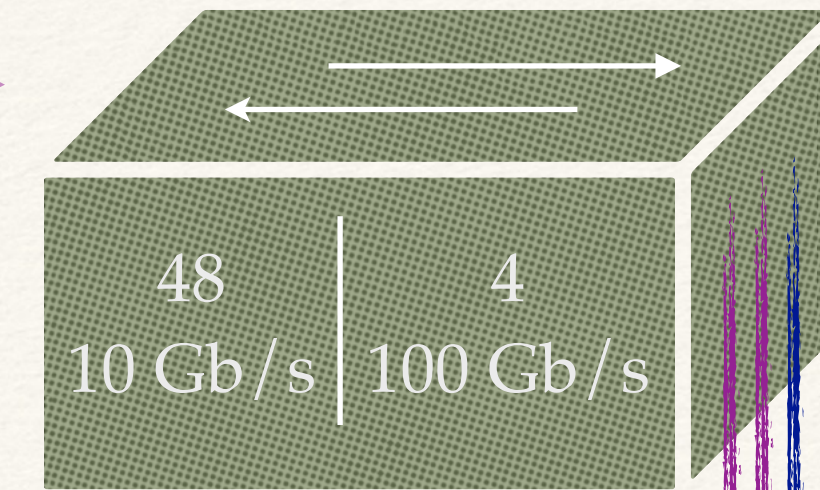
About 50-60 %
less pressure on tcp/ip net.

180 Channels

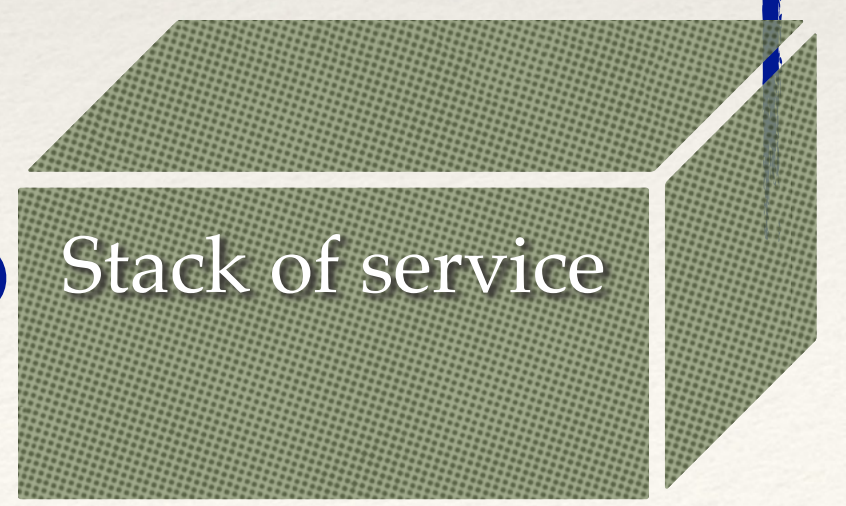
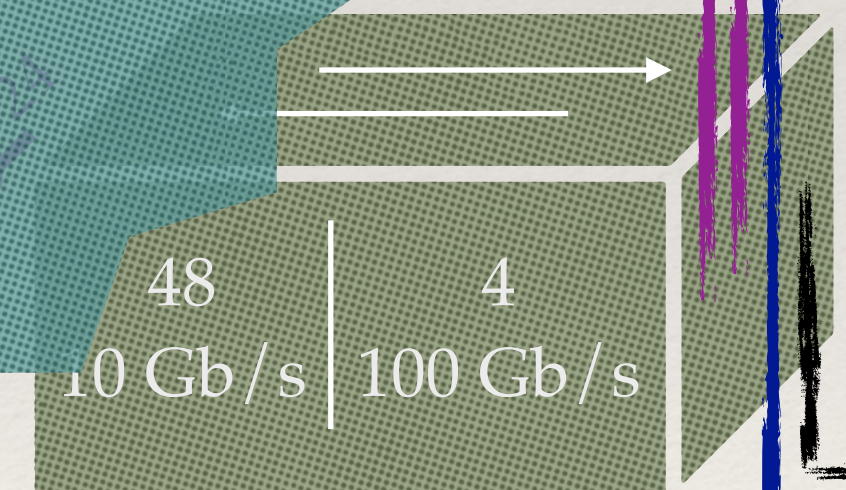
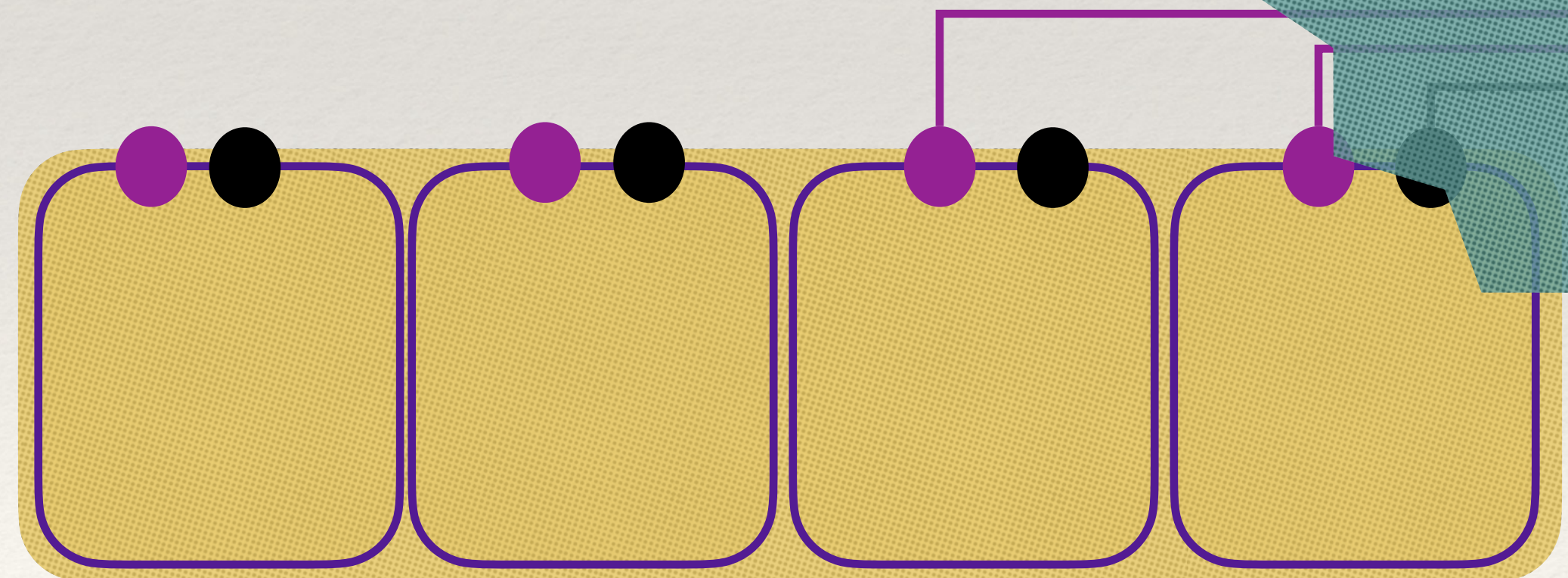
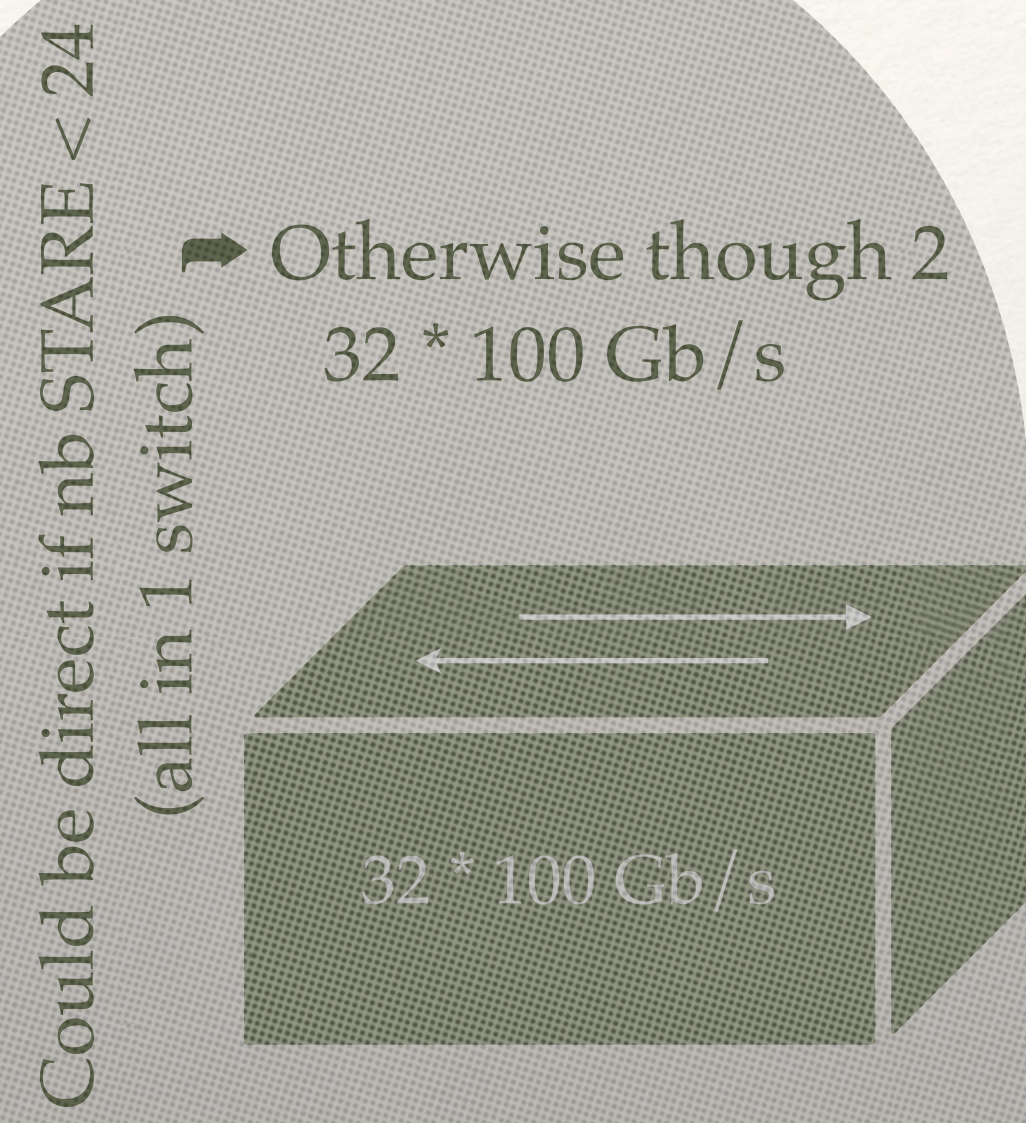
② = spare



1 switch = 24 STARE

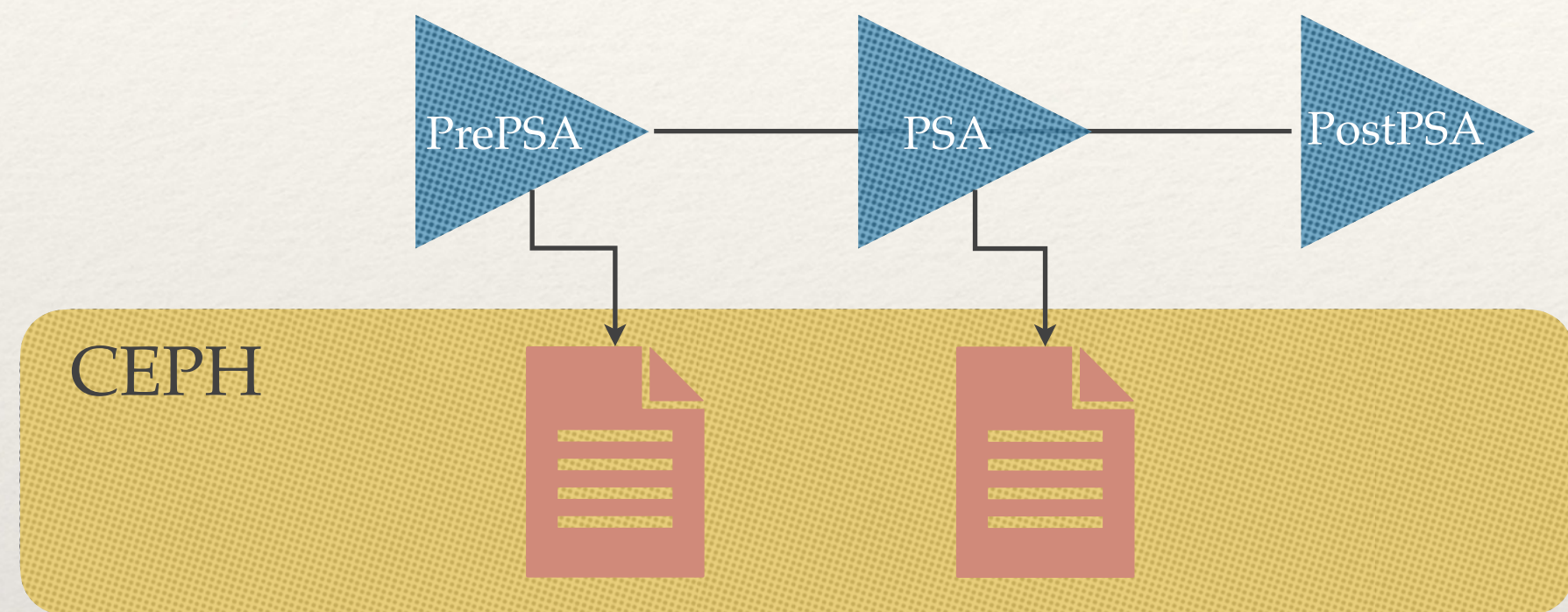


Traffic on the network should be well mastered !!!
Advanced monitoring needed !

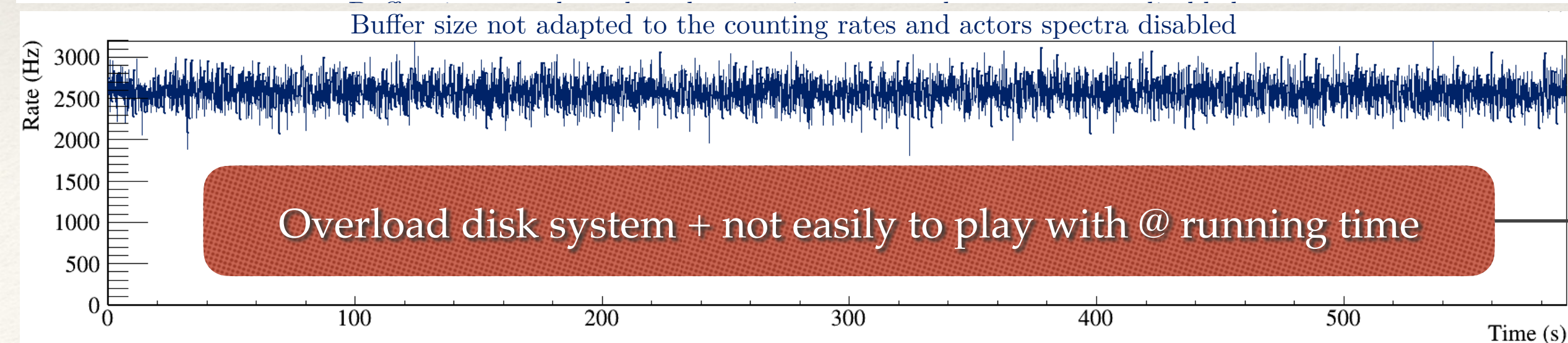
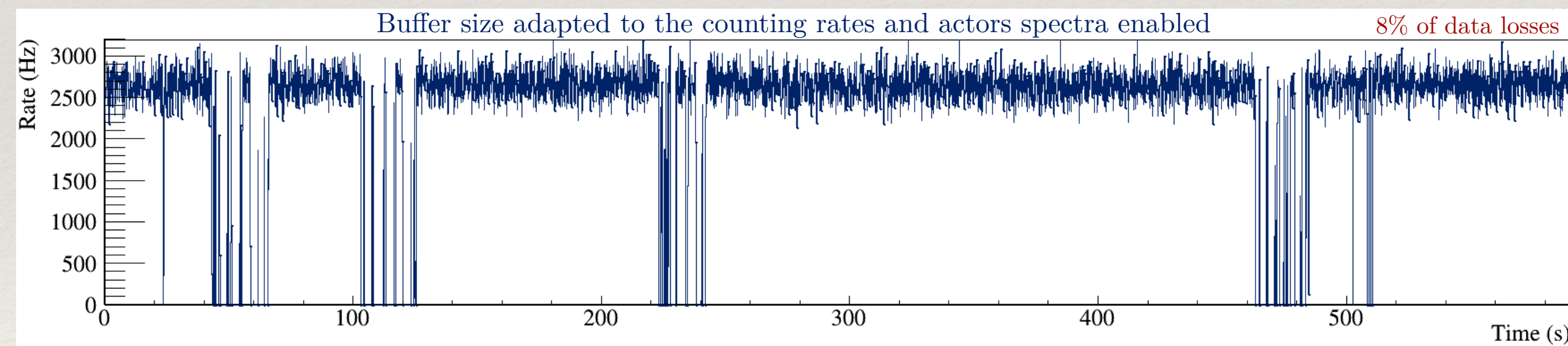


Some monitoring we have worked with so far ...

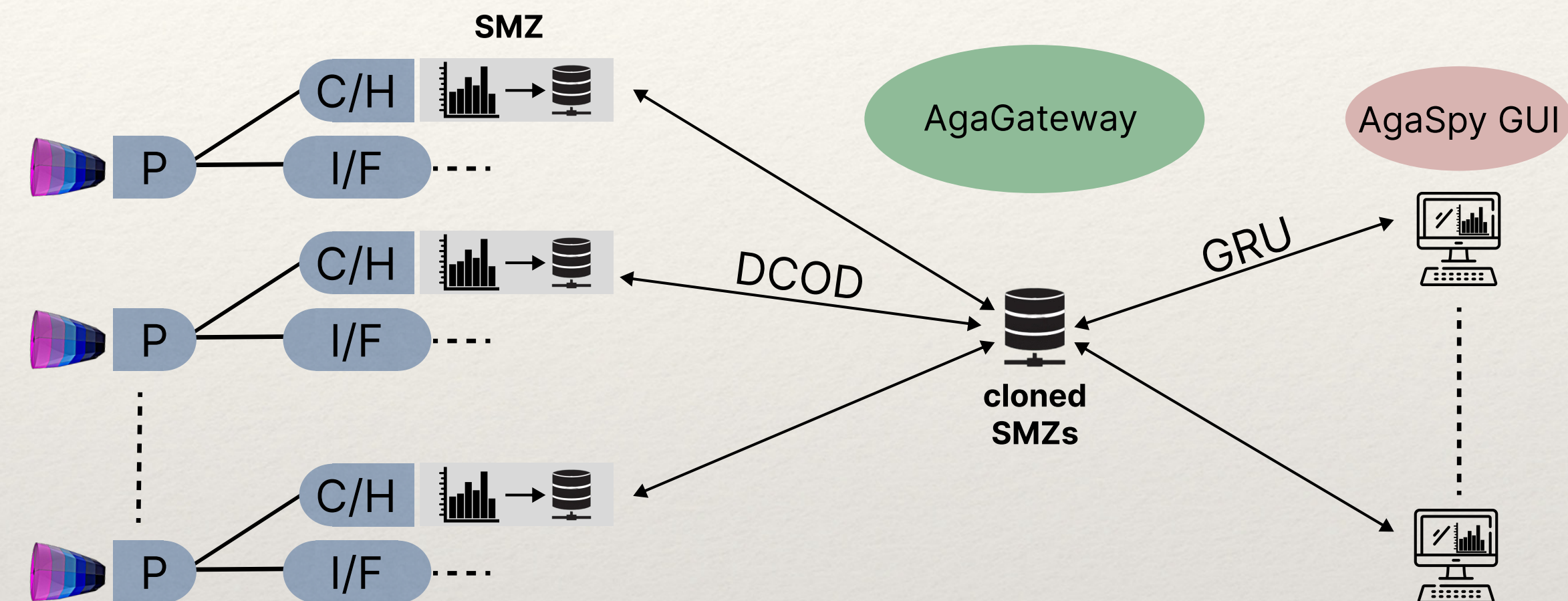
Method 1 direct dump of spectra built in processing nodes



➔ It generates losses of collected data !



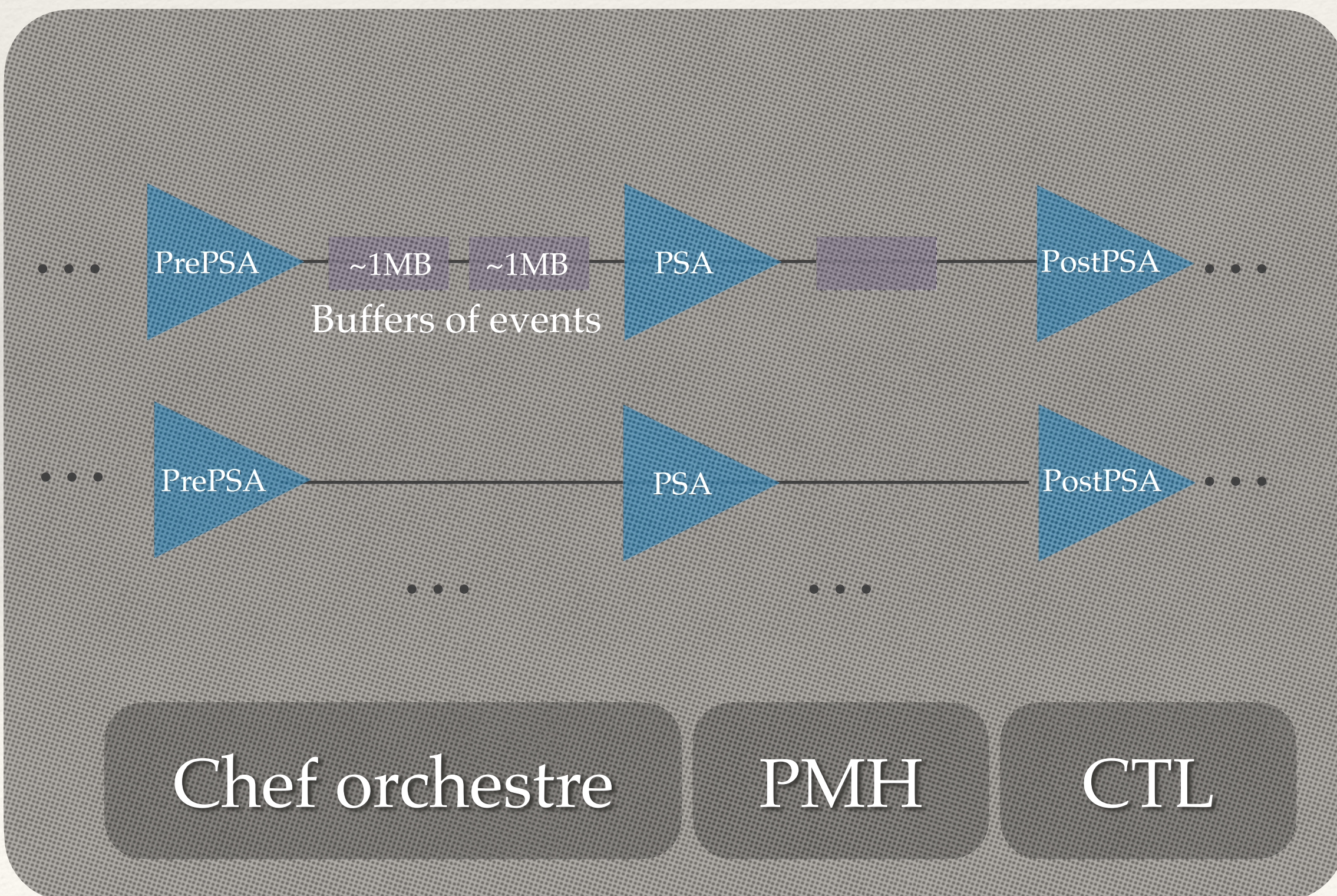
Method 2 dedicated processes upgrading shared memories



It requires duplications of data many streams
A bit more flexible but not enough

We need to produce & access @ demand
We should avoid overloading the workflow
We would like to check time evolution of inner variable

② New Data Processing - general scheme



DCOD for online is a our foundation

② New Data Processing - general scheme

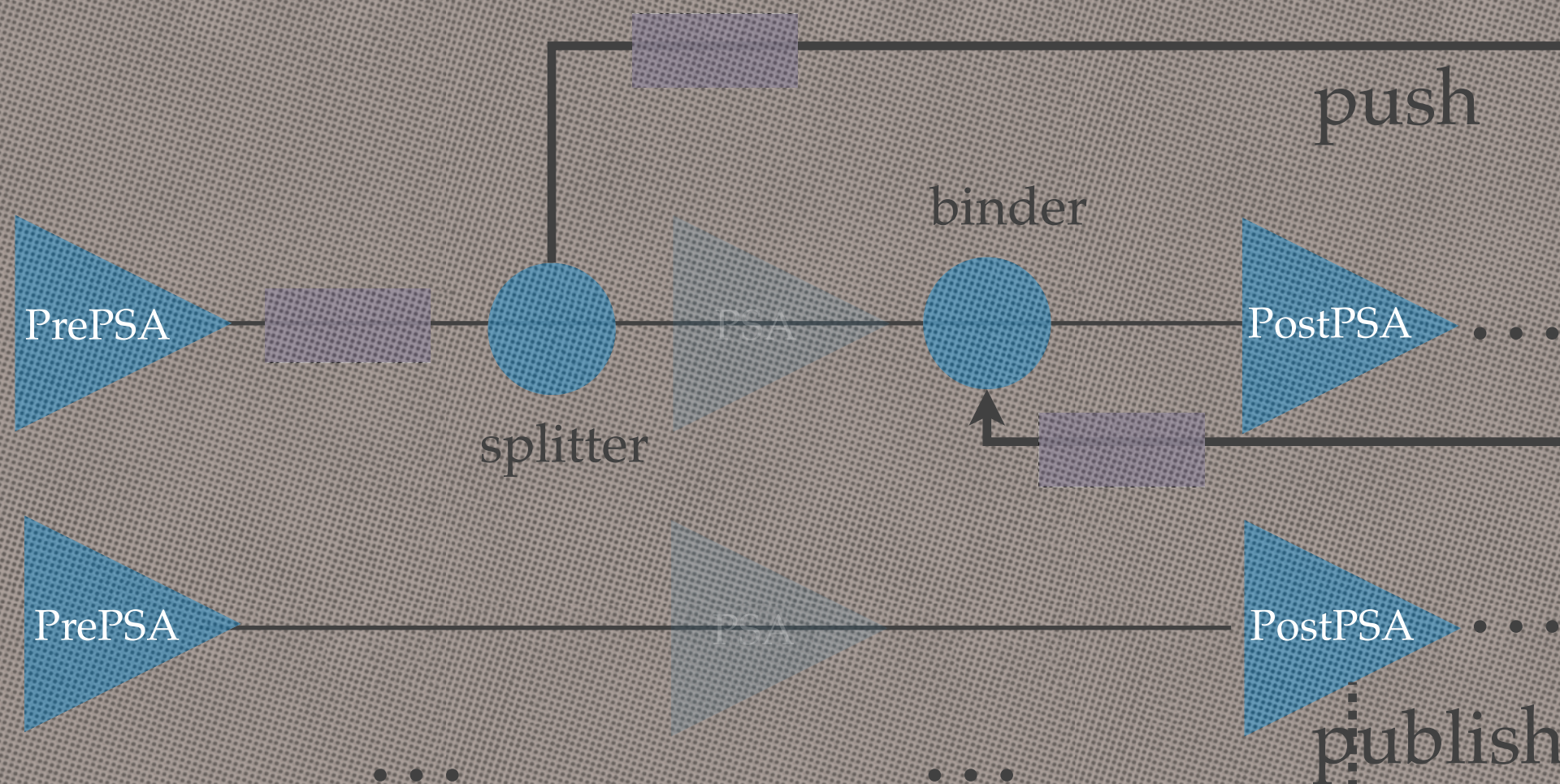
We use  redis as Shared Memory & Message Broker

Open software
High performances

SM

MB

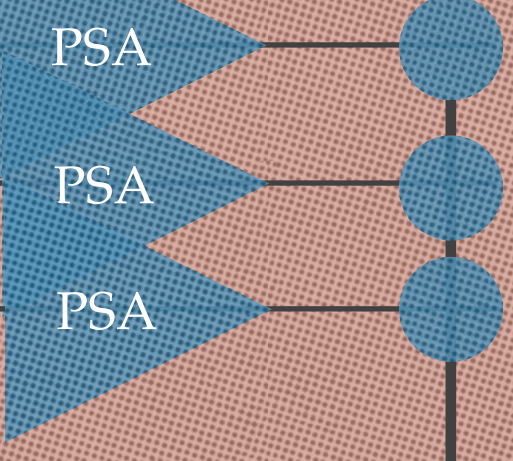
We add micro services through containers (docker, docker-swarm)



SM / Redis

SM / Redis

containerized app.



one can run as many as needed with portainer

docker-swarm portainer


MB / Redis

Time series DB



full monitoring control on / off / reset ...



 AGASPY

 grafana

Chef orchestre PMH CTL

DCOD for online is a our foundation

2 New Data Processing

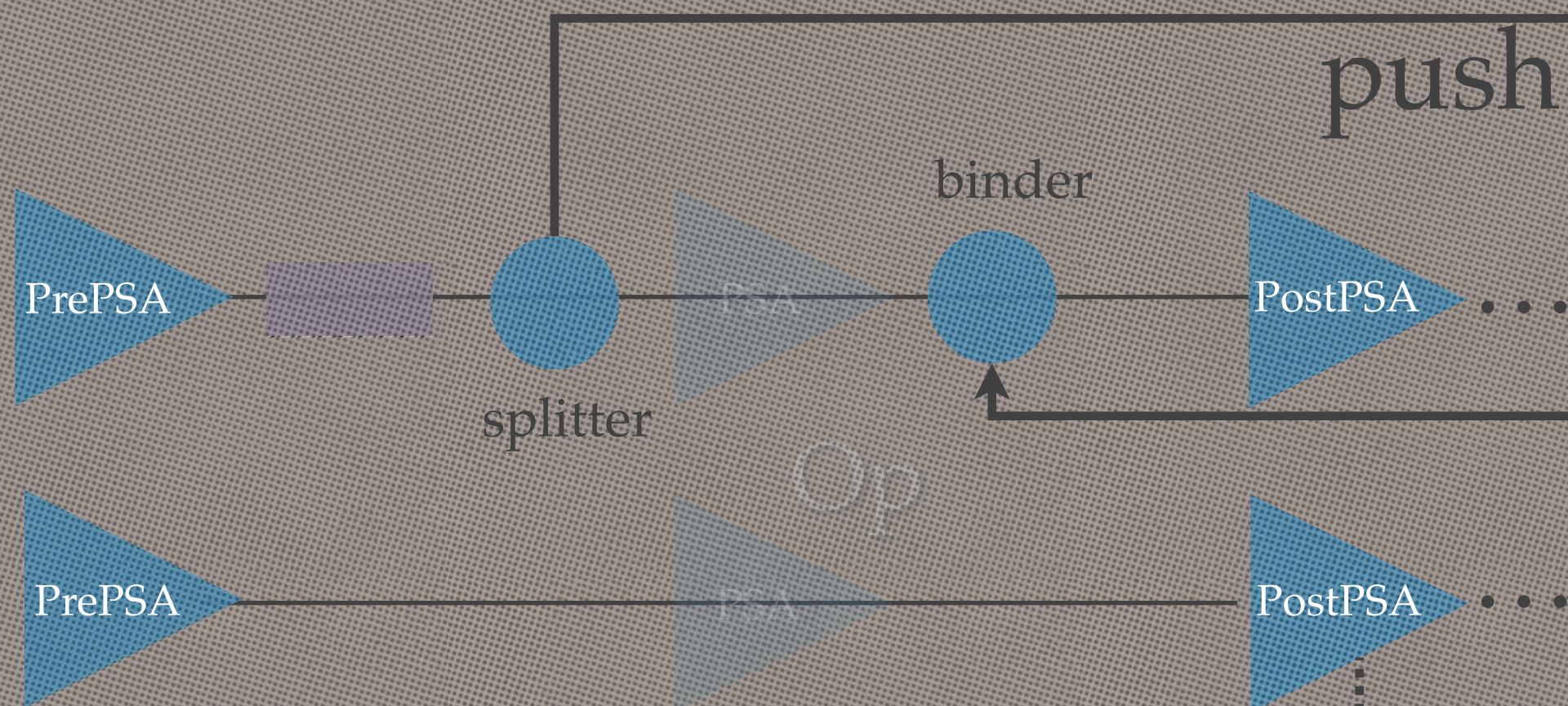
We use  redis as Shared Memory & Message Broker

Open software
High performances

SM

MB

We add micro services through containers (docker, docker-swarm)



SM/Redis
SM/Redis
Scalable
Can run on cluster

Scalable
One can add as many computers as required
GPU included

one can run as many as needed with portainer

MB/Redis

Timestamp variable/spectra

full monitoring control
on / reset /

UI

Scalable

Chef orchestrate

PMH

CTL

DCOD for online is a our foundation



② New Data Processing - where we are

New components of the processing are

- ↳ developed to fit such new processing
- ↳ concurrency is pushed as much as possible

We have many components ready

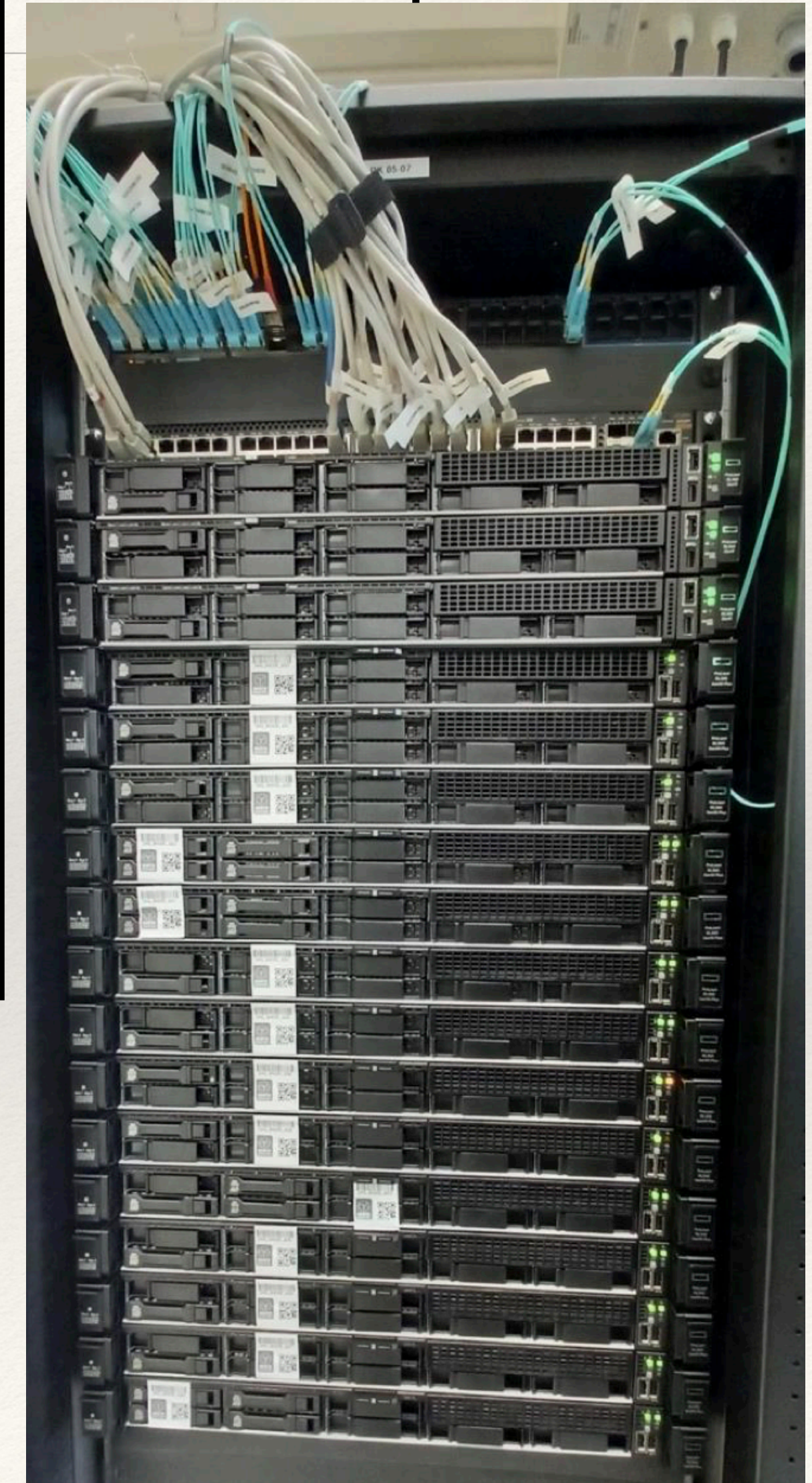
- ↳ developed in 'local' non optimised environments

We have set up in 2024 a dedicated HPC farm@Orsay

- ↳ fully 10Gb/s

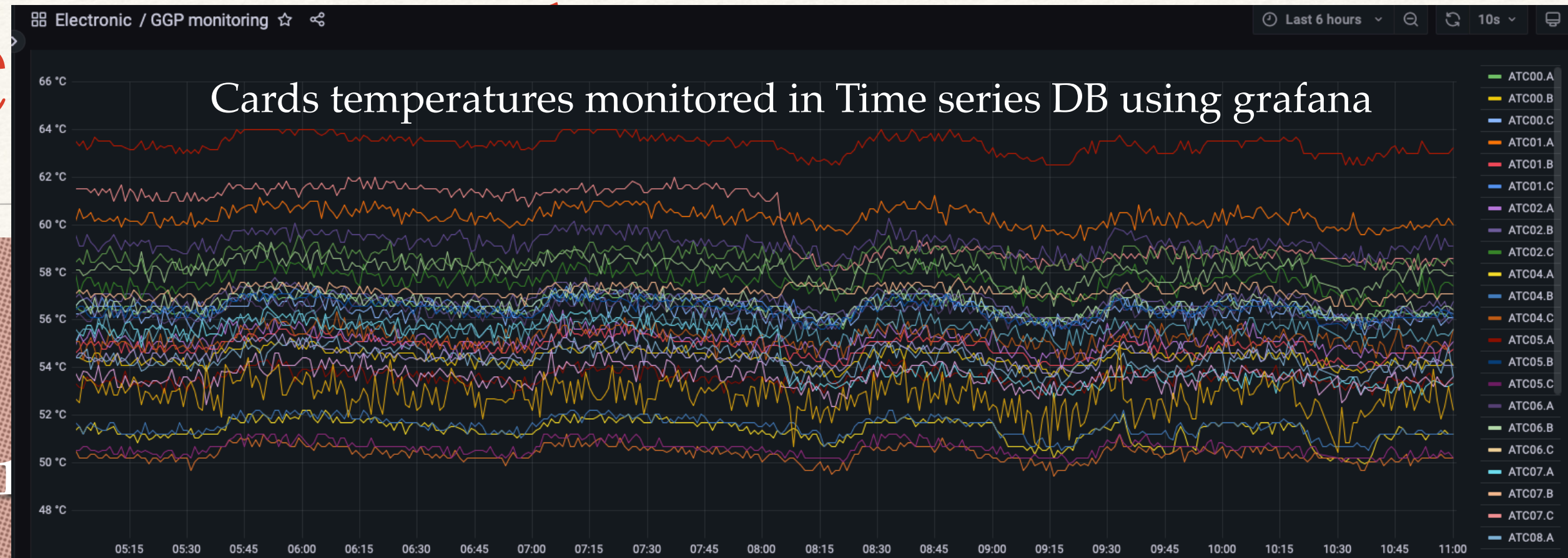
We have already components running at Legnaro

- ↳ for global workflow & electronic cards

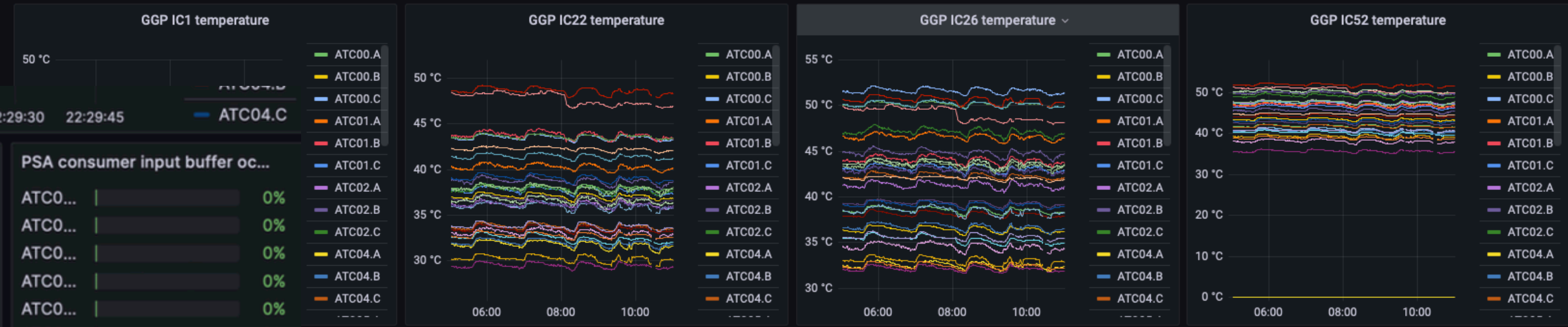
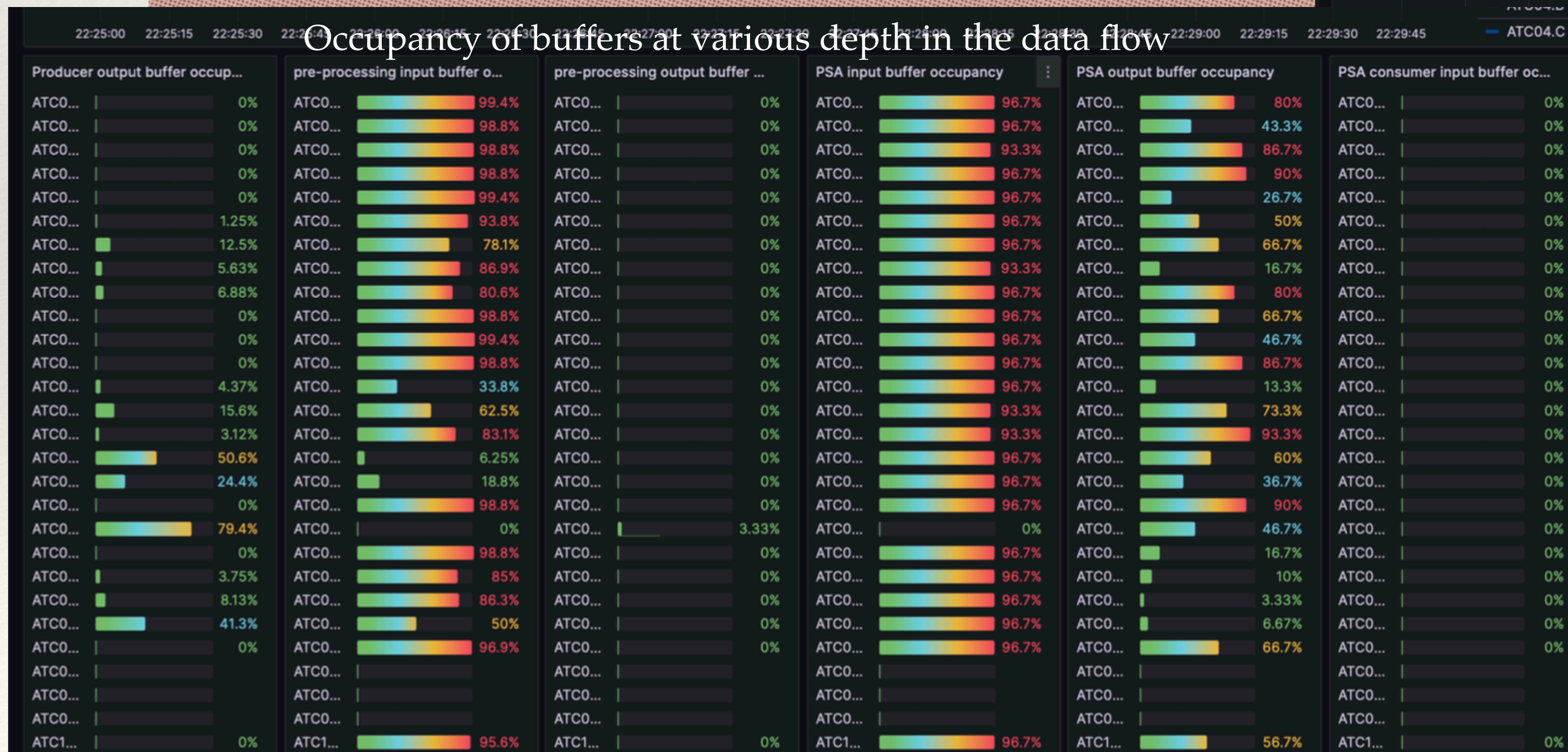


2 New Data Process

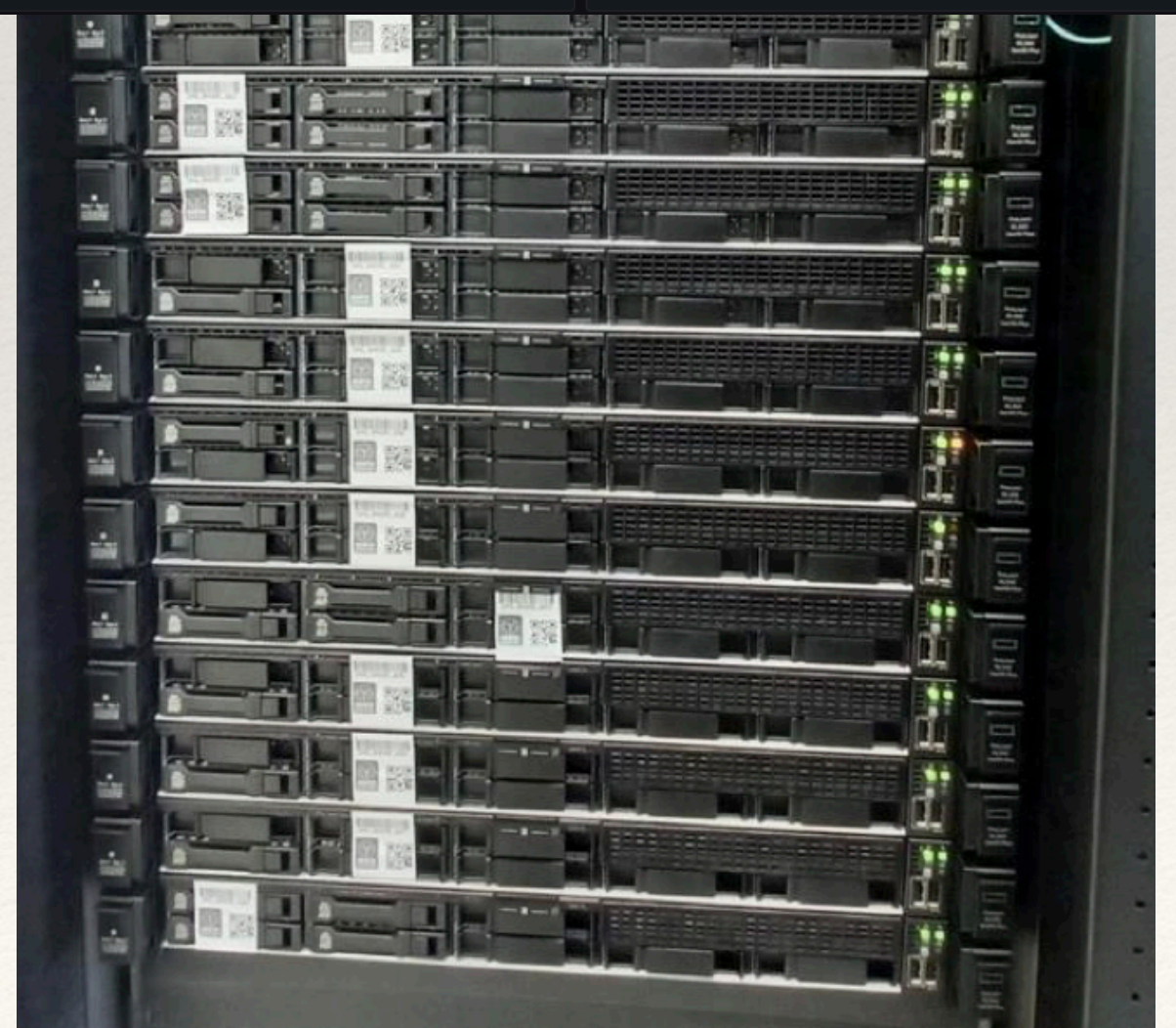
New components of the processing are
→ developed to fit such new processing
→ concurrency is pushed as much as



Occupancy of buffers at various depth in the data flow

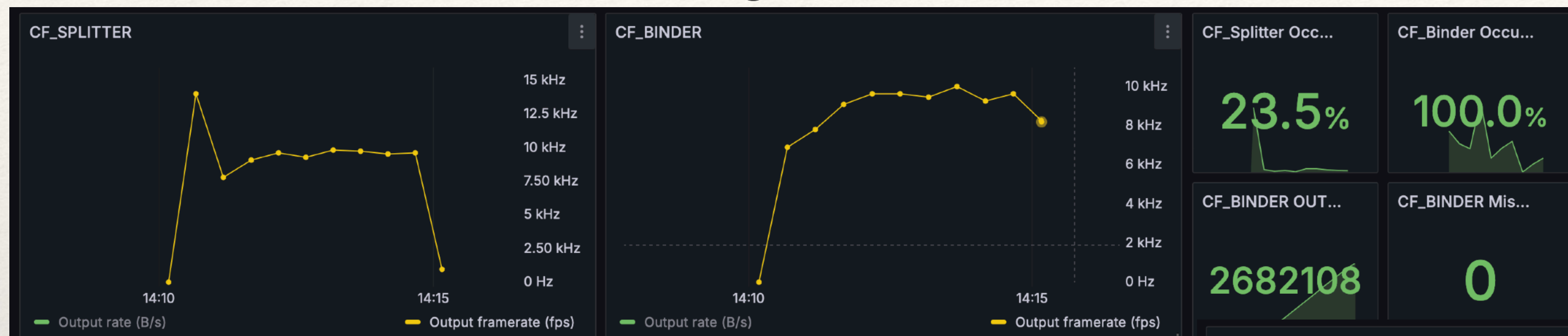


rsay

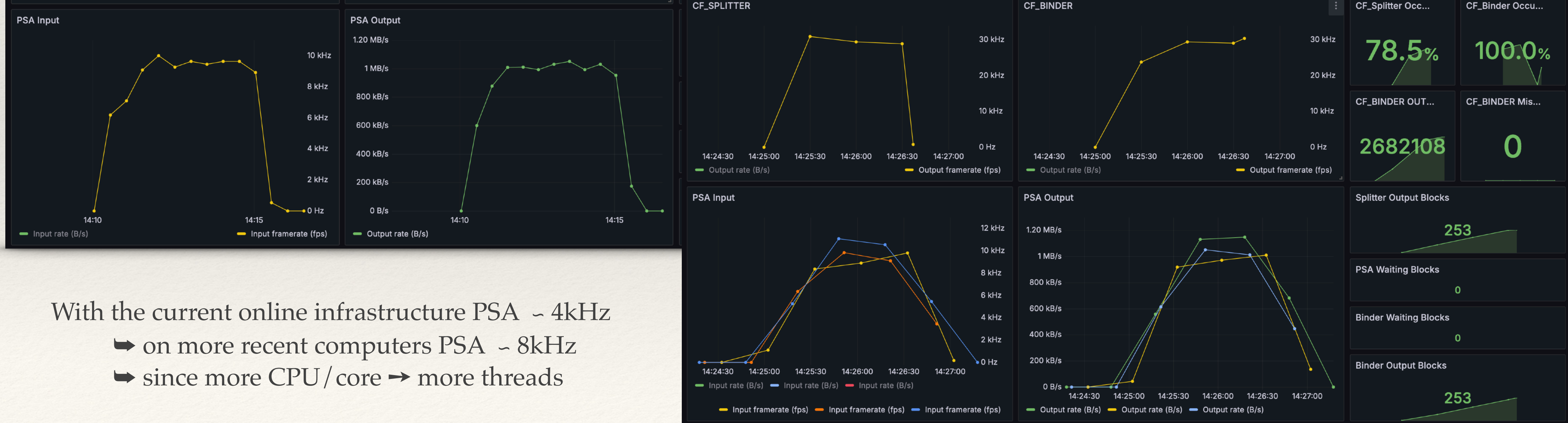


② New Data Processing with the current PSA

1 PSA running @ 10 kHz



3 PSA running @ 10 kHz in parallel → 30kHz




With the current online infrastructure PSA ~ 4kHz

- ➔ on more recent computers PSA ~ 8kHz
- ➔ since more CPU/core → more threads

2 New Data Processing with the current PSA

Portainer web interface, control processing by containers

➔ can add a service in case the current processing is too slow

snode023 
manager
CPU: 48
Memory: 202.48 GB
ready


portainer_agent
Image: portainer/agent:2.19.5
Status: running
Update: 2024-11-04 15:25:12

service_grafana
Image: grafana/grafana:11.2.0
Status: running
Update: 2024-11-14 15:11:31

service_influxdb
Image: influxdb:2.7.6-alpine
Status: running
Update: 2024-11-14 15:11:31

service_redis
Image: redis:latest
Status: running
Update: 2024-11-14 15:11:46

service_ts_collector
Image: gitlab-registry.in2p3.fr/ip2igamma/docker_images:prod
Status: running
Update: 2024-11-14 15:11:48

snode030 
worker
CPU: 48
Memory: 134.84 GB
ready

portainer_agent
Image: portainer/agent:2.19.5
Status: running
Update: 2024-11-04 15:25:19

service_redis
Image: redis:latest
Status: running
Update: 2024-11-14 15:11:41

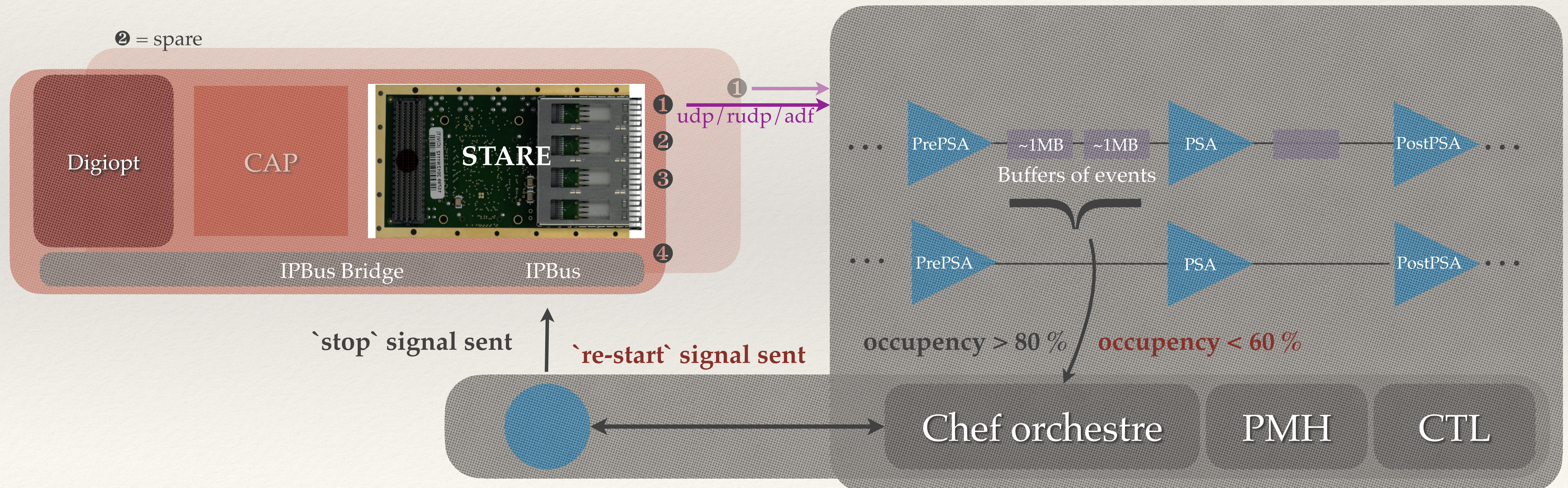
slow_path_psa
Image: gitlab-registry.in2p3.fr/ip2igamma/docker_images:prod
Status: running
Update: 2024-11-14 15:14:39

Name	Stack	Image	Scheduling Mode	Published Ports	
fast_path_fast_path	fast_path	gitlab-registry.in2p3.fr/ip2igamma/docker_images:prod	replicated 0 / 1	Scale -	
Status Filter Task Actions Slot Node Last Update					
complete	vqxa4ydnwi16l1xeia72td36x		1	snode021	2024-11-14 15:18:30
portainer_agent	portainer	portainer/agent:2.19.5	global 21 / 21	-	
portainer_portainer	portainer	portainer/portainer-ce:2.19.5	replicated 1 / 1	Scale 8000:8000 9000:9000 9443:9443	
service_grafana	service	grafana/grafana:11.2.0	replicated 1 / 1	Scale 3000:3000	
service_influxdb	service	influxdb:2.7.6-alpine	replicated 1 / 1	Scale -	
service_redis	service	redis:latest	global 21 / 21	-	
service_ts_collector	service	gitlab-registry.in2p3.fr/ip2igamma/docker_images:prod	replicated 1 / 1	Scale -	
slow_path_psa	slow_path	gitlab-registry.in2p3.fr/ip2igamma/docker_images:prod	replicated 3 / 3	Scale -	
Status Filter Task Actions Slot Node Last Update					
running	yv1f23apupum7s7zs6szktrkp		1	snode030	2024-11-14
running	5l1d5h1z6p8hmfnc9y4pwnt15		2	hp-CZJ3480D11	2024-11-14
running	dx4cmxwzglnd6r2hpss9wye1c		3	snode032	2024-11-14

② New Data Processing back pressure management

Pressure on RAM in the whole system is measured in real time

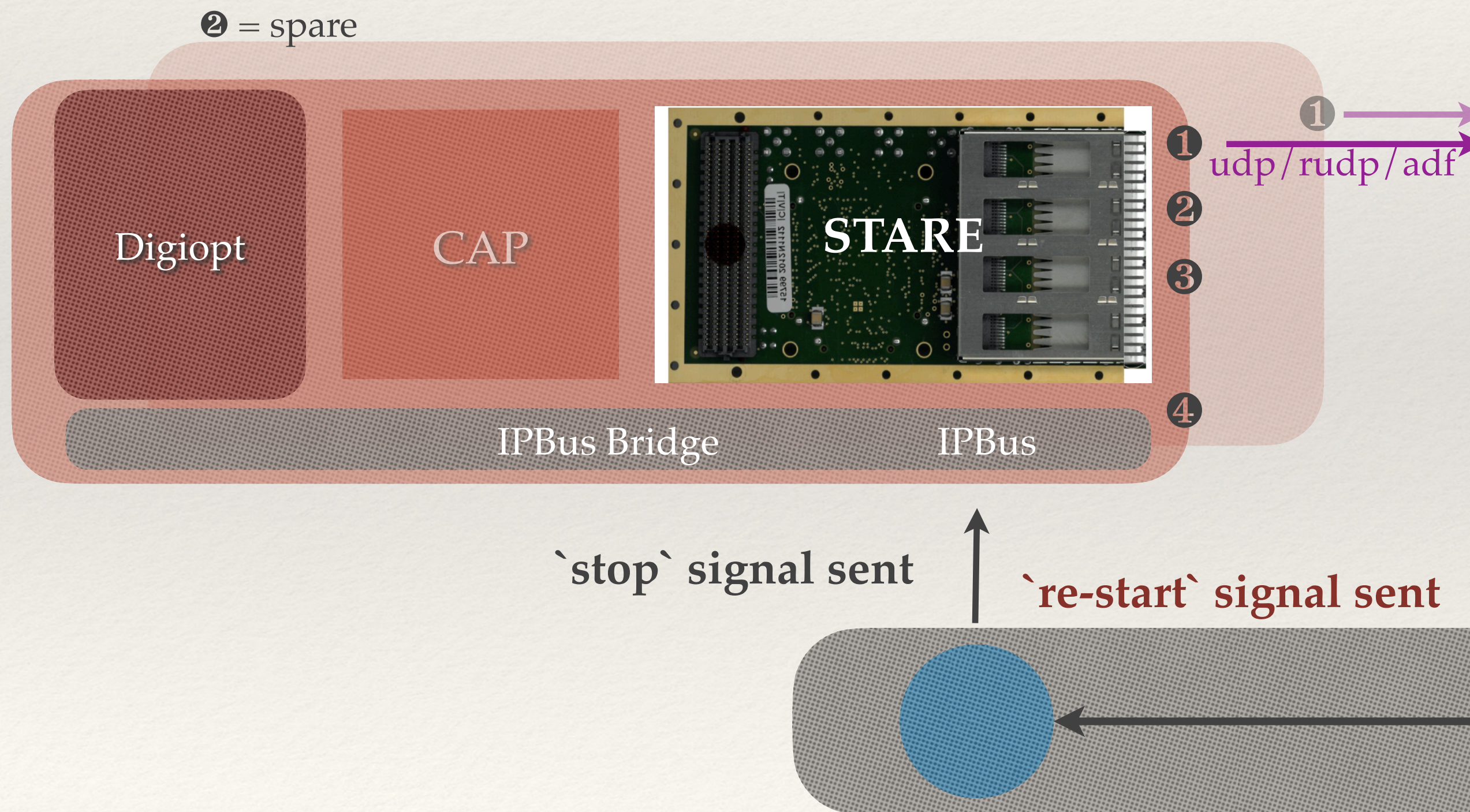
➔ signals can be sent to the PACE via IPBus to regulate



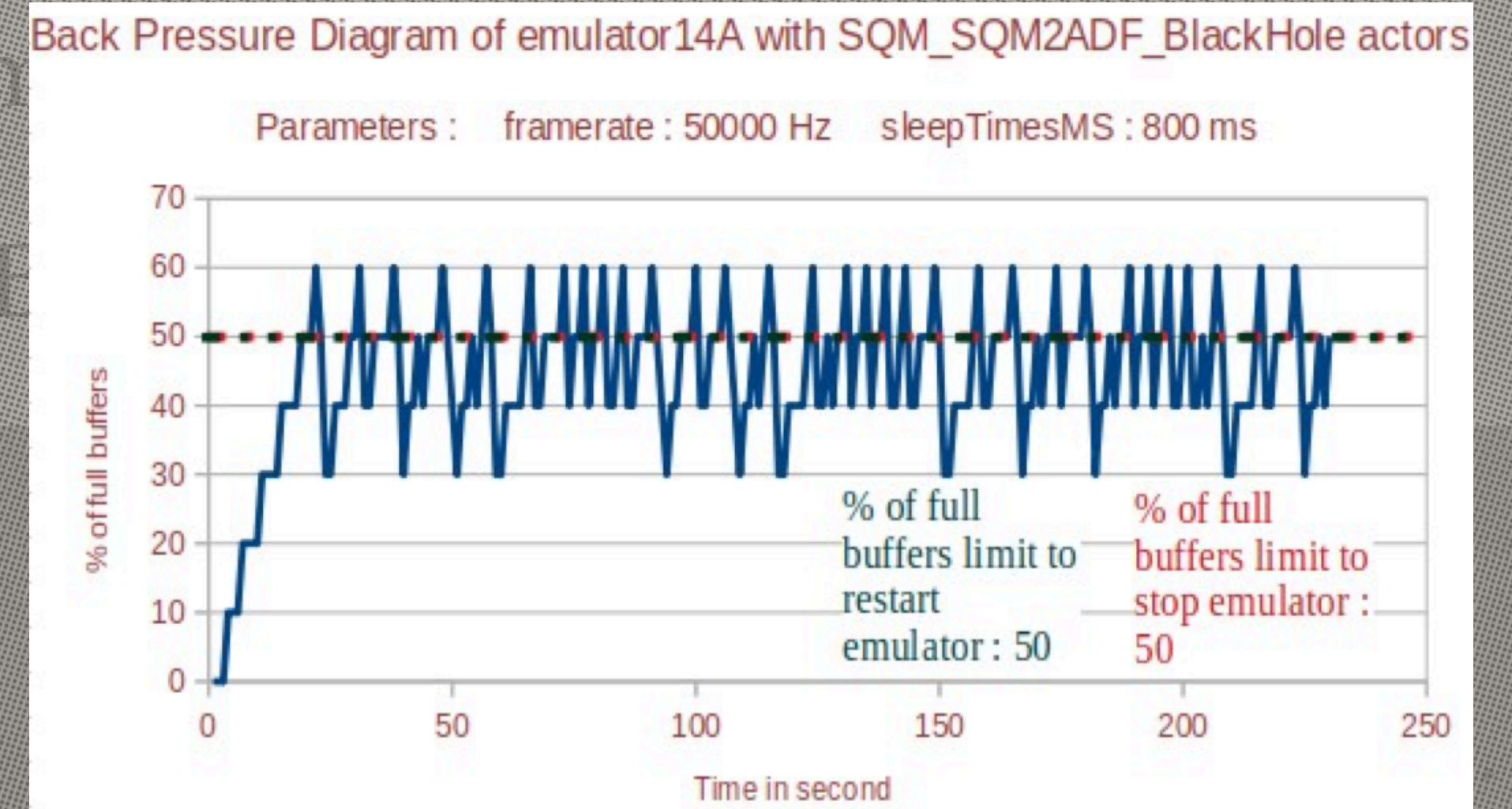
② New Data Processing back pressure management

Pressure on RAM in the whole system

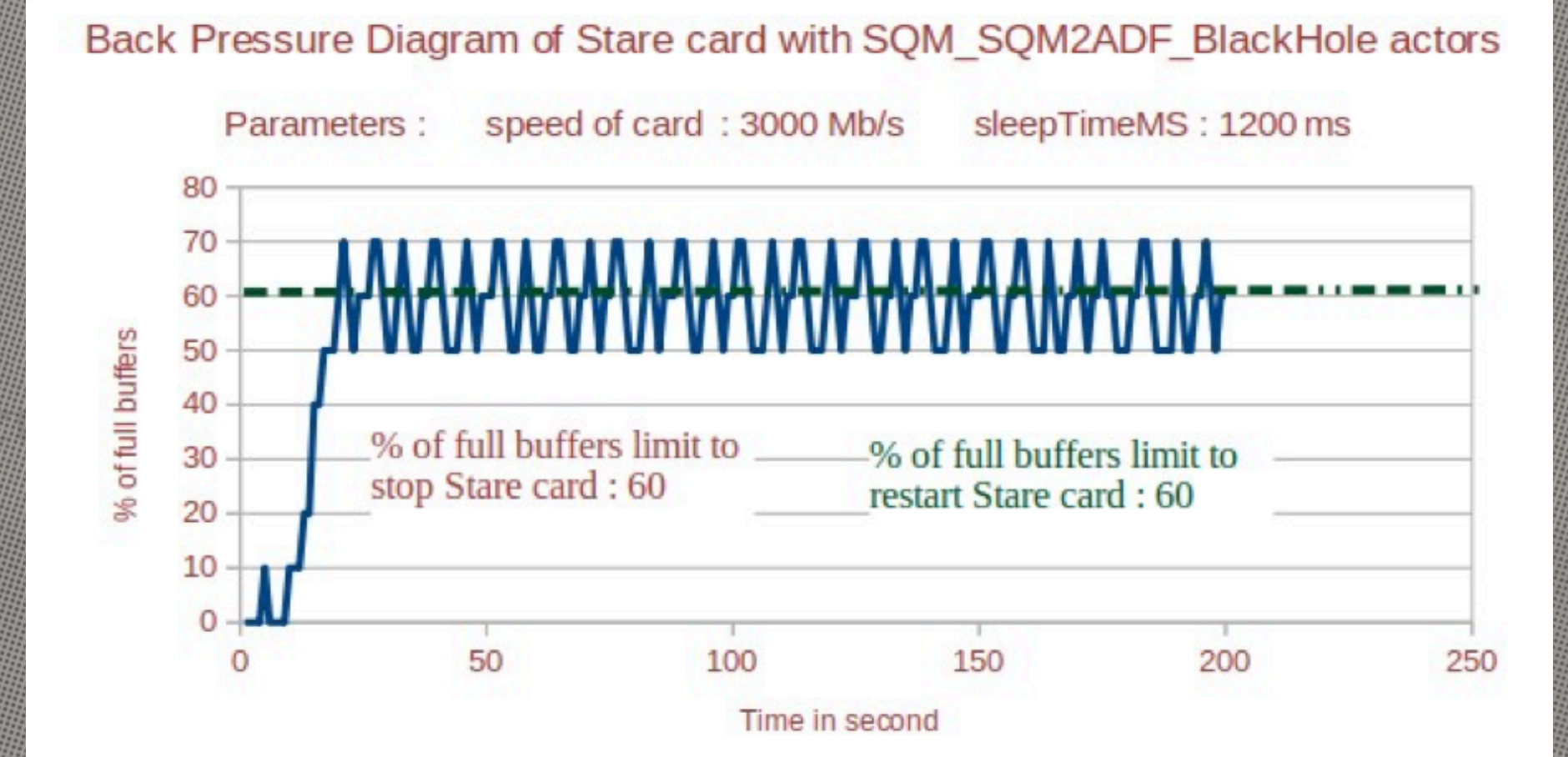
➔ signals can be sent to the PACE via



Test with STARE Emulator



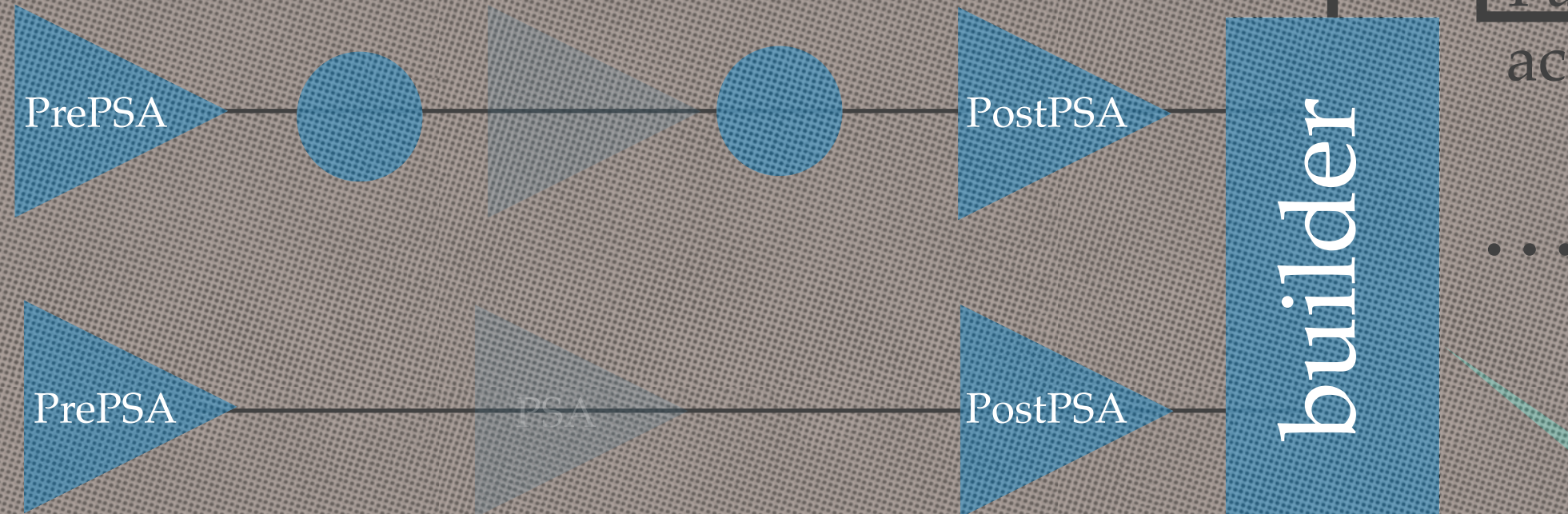
Test with STARE card



2 New Data Processing - Software trigger

pop

We add micro services through containers (docker, docker-swarm)



'Fast' lines, PSA not done here
But partial events - no hits - are sent to builder
Buffers of incomplete data waiting in SM/REDIS

SM/Redis

SM/Redis

Pop only events
in accepted pattern

containerized app.

PSA

PSA

PSA

docker-swarm
portainer

At this level we have :
Coincidences w/ wo ancillary
Energies of segments/core
Full ancillary if there
.....

One can build
complex / physical triggers
before applying PSA !

DCOD for online is a our foundation

Conclusions

We have developed many components of our new processing model

↳ Local level data pipeline already well stressed up to 50 kHz included traces compressions

DCOD for 'fast' processing lines + micro services in containers for 'slow' ones

↳ We have run the current PSA up to 40 kHz + tested a dummy NN based on on GPU

↳ We have a more flexible / efficient / scalable processing & monitoring system

↳ We need to fully qualify the performances ... with have dedicated infrastructure fully 10Gb/s

We need a better data [meta] management ... FAIR approach

↳ We have started the first steps ... Ex : meta data sub-directory included in raw data set + data from time series database