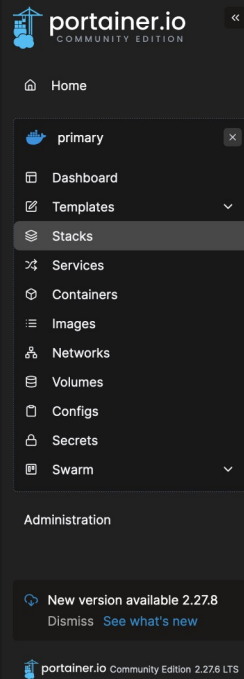


A Distributed Agata Emulator



replay_local_initializer	gitlab-registry.in2p3.fr/ip2gamma/docker_images:prod	replicated	1 / 1	Scale	-	2025-07-01 10:44:30
Status	Filter	Task	Actions	Slot	Node	Last Update
running	tux501nqeallrbftt08mizljn			1	agata-analysis-6	2025-07-01 10:44:30
replay_local_processor	gitlab-registry.in2p3.fr/ip2gamma/docker_images:prod	replicated	11 / 11	Scale	-	2025-07-01 10:44:40
Status	Filter	Task	Actions	Slot	Node	Last Update
running	1w0mlb4etmynn2zv6stwyvuv			6	agata-analysis-6	2025-07-01 10:44:40
running	2apgy1qxulpkdoktfyuybyfgo			9	agata-analysis-8	2025-07-01 10:44:40
running	8oncw3mya4jvblxolgtkezy2b			2	agata-analysis-8	2025-07-01 10:44:40
running	afmcv3i7ihz35flqphm97k682			10	agata-analysis-8	2025-07-01 10:44:40
running	cn8f4dsqpm6cna2ktmg9lgkrf			5	agata-analysis-7	2025-07-01 10:44:40
running	j7jzdpd2fz14ui2amd7acefq			1	agata-analysis-6	2025-07-01 10:44:40
running	tkokogplj5gglgw1qm2p4535z			4	agata-analysis-7	2025-07-01 10:44:40
running	w4rx96do7xnu1fe3hgnxdlccp			11	agata-analysis-7	2025-07-01 10:44:40
running	xdxmvj081e3yw6w73nj6bc3wc			8	agata-analysis-6	2025-07-01 10:44:40
running	xqjjwoveqwoi4f1ajvlt71vn			3	agata-analysis-8	2025-07-01 10:44:40



Cluster and Scalability
Local level Processing
Global Level Processing

G. Baulieu

New analysis servers set up at Legnaro (P. Le Jeannic)

- 4 new machines
 - 2 x Xeon Silver 4310 (48 threads) @ 2.1 GHz
 - 128 GB of RAM
 - 10 Gbs access to data (anodeds5)

New analysis servers set up at Legnaro (P. Le Jeannic)

- 4 new machines
 - 2 x Xeon Silver 4310 (48 threads) @ 2.1 GHz
 - 128 GB of RAM
 - 10 Gbs access to data (anodeds5)

Processing will become more demanding

- From ~33 crystals to 135 (up to 180?) : at least a factor 4

New analysis servers set up at Legnaro (P. Le Jeannic)

- 4 new machines
 - 2 x Xeon Silver 4310 (48 threads) @ 2.1 GHz
 - 128 GB of RAM
 - 10 Gbs access to data (anodeds5)

Processing will become more demanding

- From ~33 crystals to 135 (up to 180?) : at least a factor 4

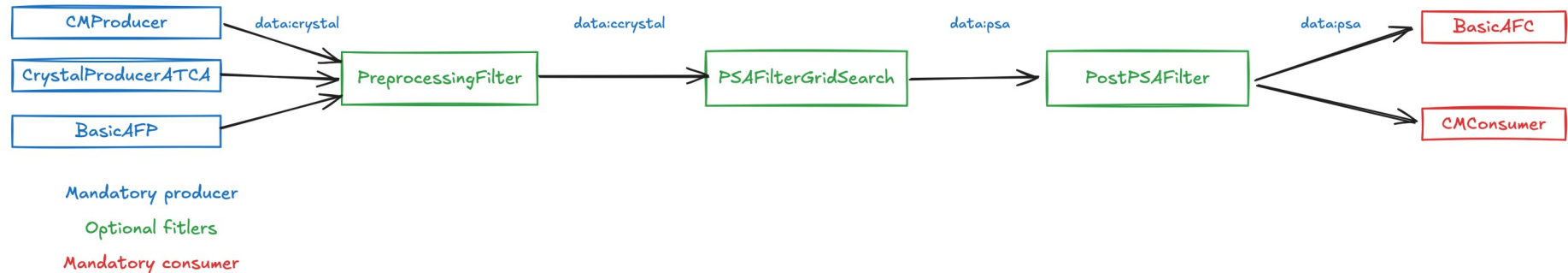
We need to be able to use computing power where it is : dispatched on different servers

Distributed Agata Emulator (DAEmule)

- A new AGAPRO emulator
- Designed to be scalable and able to run on clusters
- Using the same actors and configuration folders (from genconf.py)
- Different instances can communicate through a Central Memory system (REDIS implemented)
- Still under development, first tests and data validation performed at Legnaro end of July.

Local level processing

- Subset of actors available



- Each actor runs in its own thread
- Producer : choice between Central Memory, .cdat files or ADF files
- Filters : On/Off on Preprocessing, PSA and PostPSA
- Consumer : choice between ADF file and Central Memory
- 3 run modes :
 - Simple
 - Parallel
 - Batch

Local level processing

- Simple mode :

A single emulator on a single crystal folder, on a single machine.

- Parallel mode :

As many emulators as crystal folders, on a single machine. Define the number of emulators run in parallel. (~FEMUL behaviour)

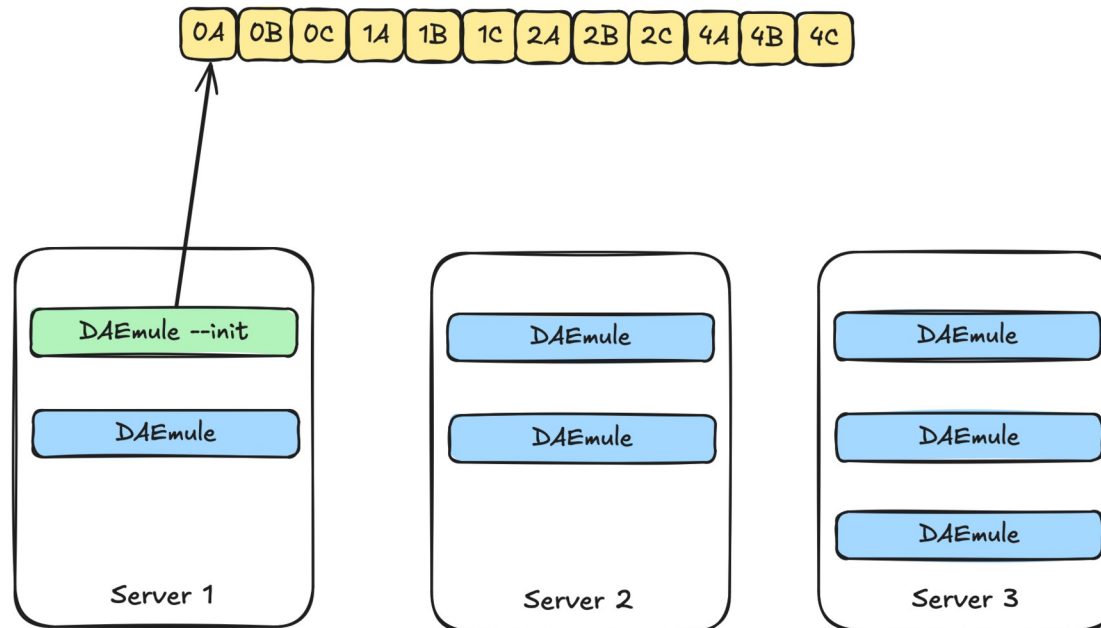
- Batch mode :

A single emulator per DAEmule instance but many instances on different servers.

Each DAEmule instance run on one crystal folder and then ask for a new one.

Local level processing

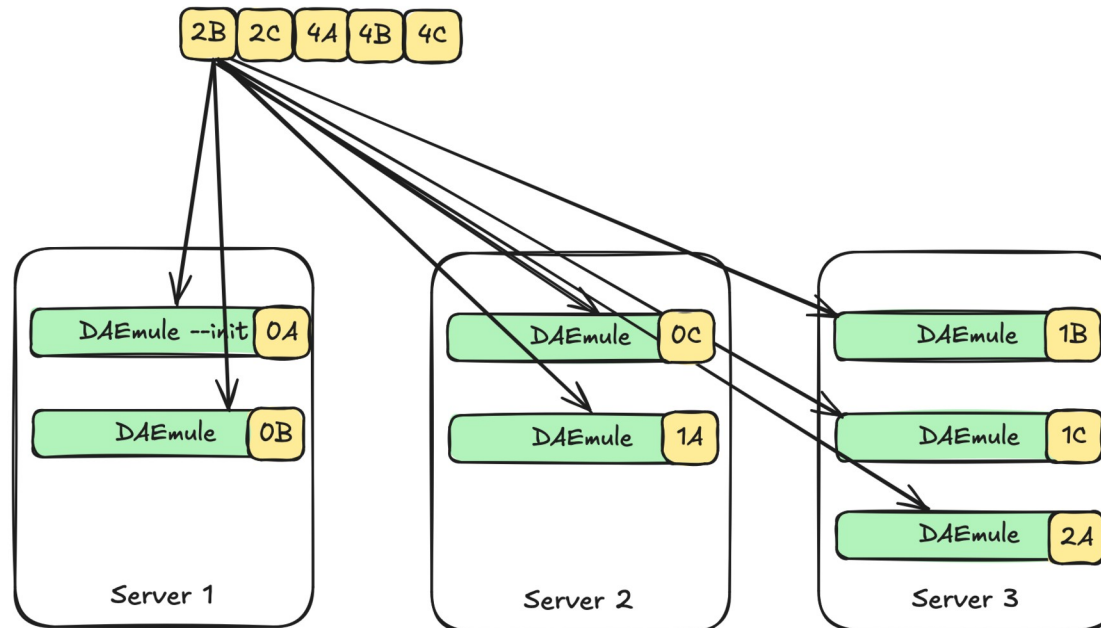
- Batch mode overview :



→ from configuration file to Central Memory list

Local level processing

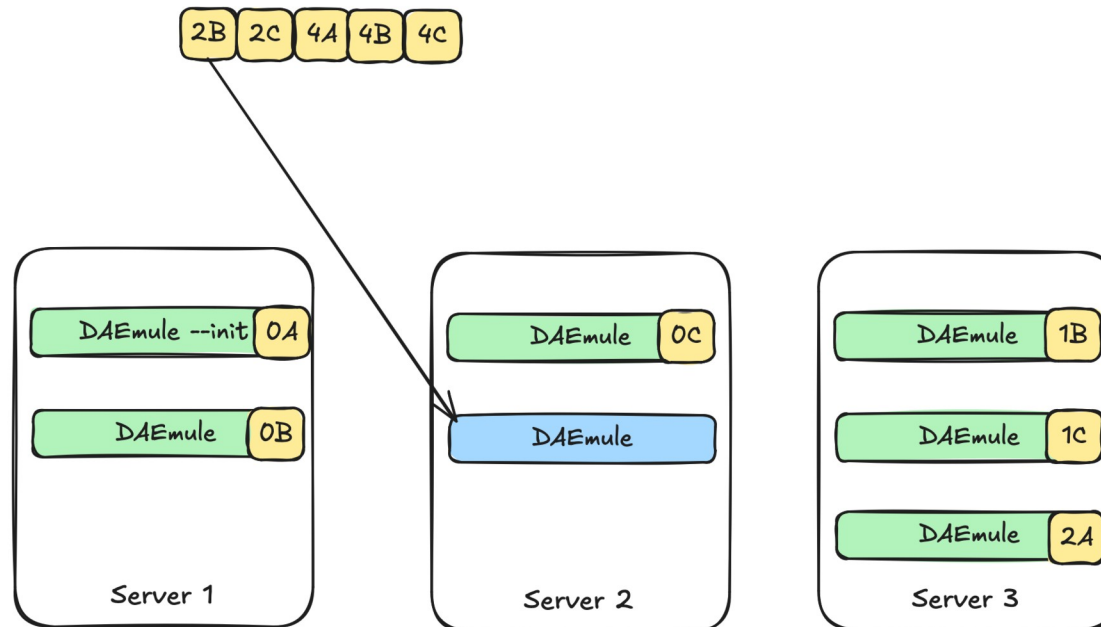
- Batch mode overview :



→ each instance fetches a workload from central memory (atomic)

Local level processing

- Batch mode overview :

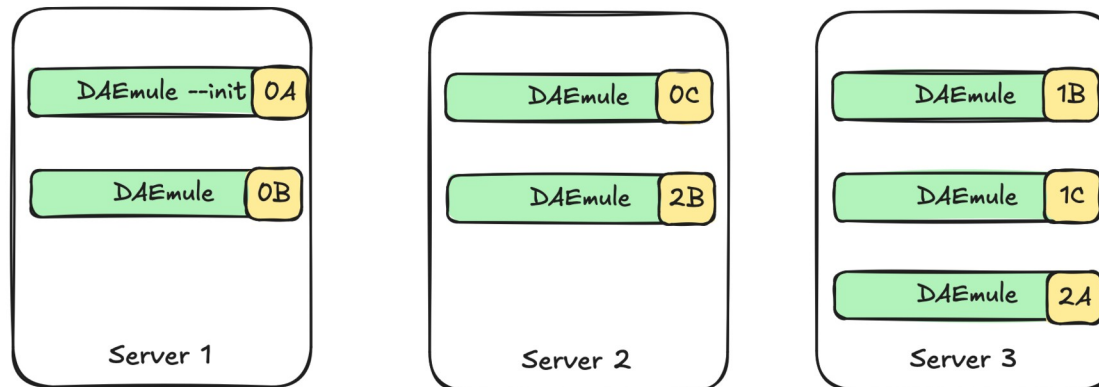


→ when an instance is done with its workload, it asks for a new one

Local level processing

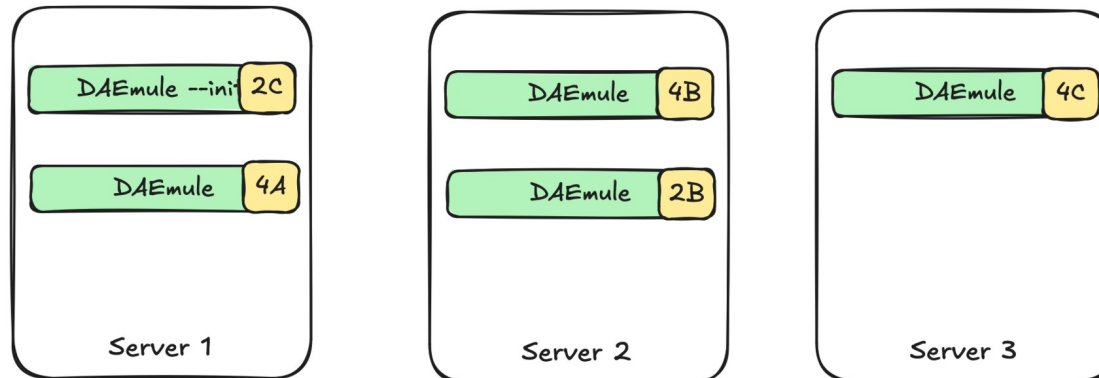
- Batch mode overview :

2C 4A 4B 4C



Local level processing

- Batch mode overview :



→ if the workload list is empty, the instance stops

Handling the cluster

- Launching many instances on different servers is a pain
 - we can automatize this process using **Docker Swarm** and **Portainer**

Handling the cluster

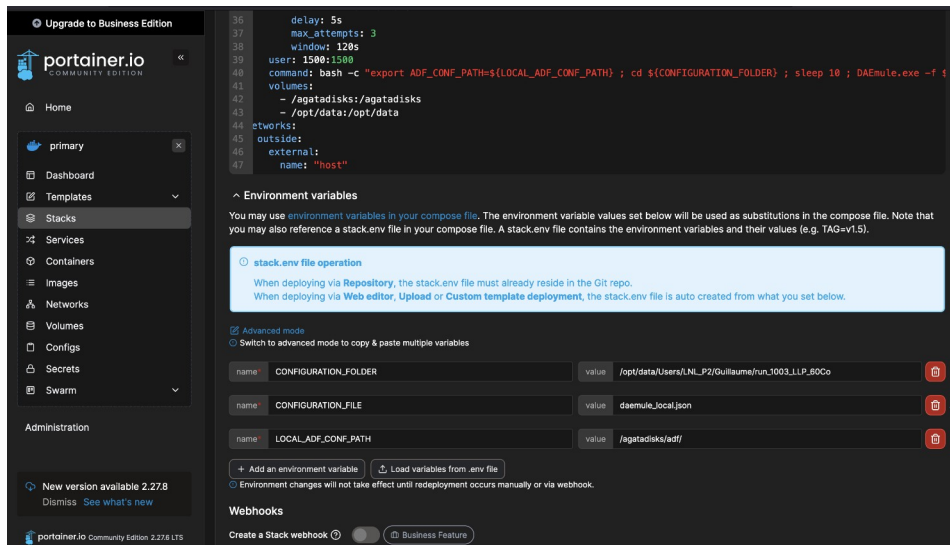
- Launching many instances on different servers is a pain
 - we can automatize this process using **Docker Swarm** and **Portainer**
- Docker Swarm will handle the processes launches on different servers using Docker containers

Handling the cluster

- Launching many instances on different servers is a pain
 - we can automatize this process using **Docker Swarm** and **Portainer**
- Docker Swarm will handle the processes launches on different servers using Docker containers
- Portainer is a web interface to Docker Swarm

Handling the cluster

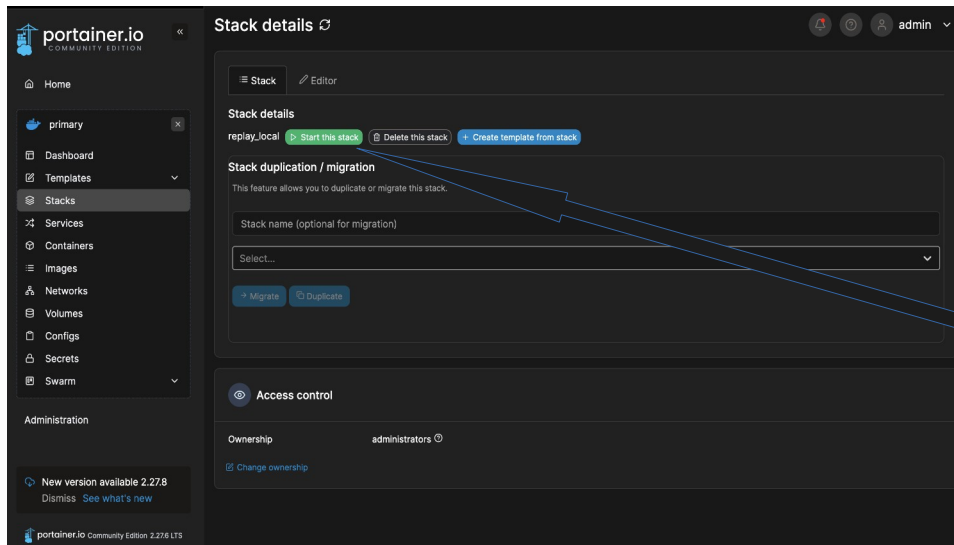
- Launching many instances on different servers is a pain
 - we can automatize this process using **Docker Swarm** and **Portainer**
- Docker Swarm will handle the processes launches on different servers using Docker containers
- Portainer is a web interface to Docker Swarm



Define the
configuration file
location

Handling the cluster

- Launching many instances on different servers is a pain
 - we can automatize this process using **Docker Swarm** and **Portainer**
- Docker Swarm will handle the processes launches on different servers using Docker containers
- Portainer is a web interface to Docker Swarm



Click on the Start button

Handling the cluster

- Launching many instances on different servers is a pain
 - we can automatize this process using **Docker Swarm** and **Portainer**
- Docker Swarm will handle the processes launches on different servers using Docker containers
- Portainer is a web interface to Docker Swarm

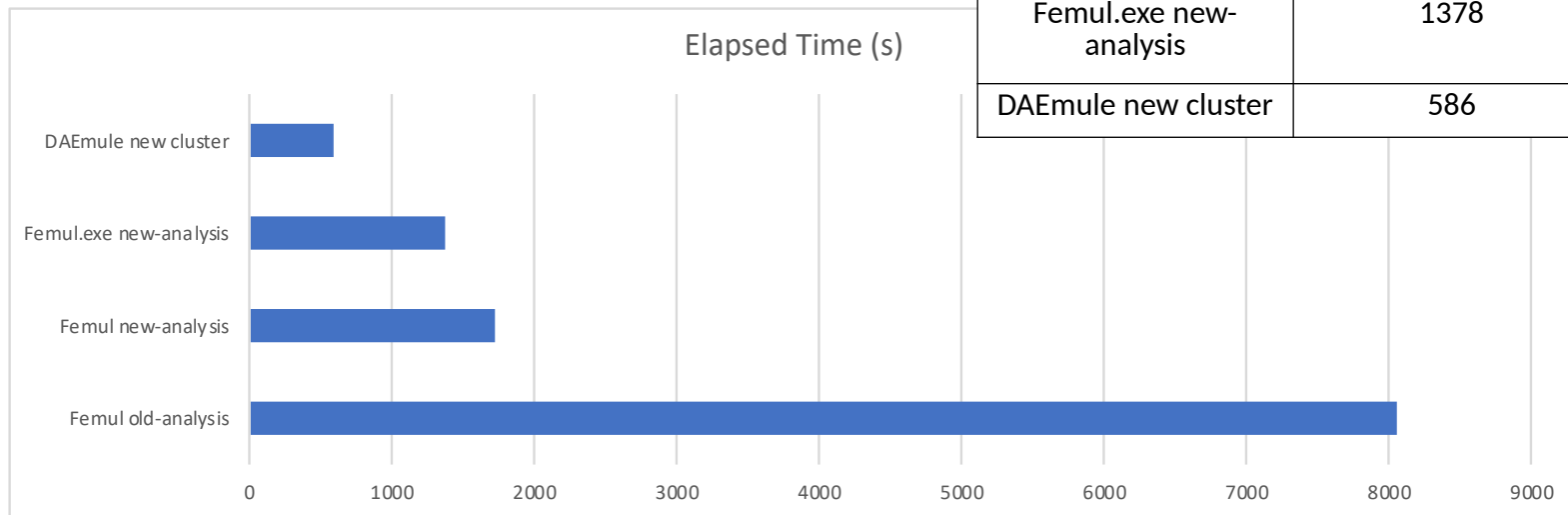
The screenshot shows the Portainer.io web interface. On the left is a sidebar with navigation options: Home, primary (selected), Dashboard, Templates, Stacks, Services, Containers, Images, Networks, Volumes, Configs, Secrets, and Swarm. The main area displays two services from the 'primary' stack:

- replay_local_initializer**: gitlab-registry.in2p3.fr/ip2igamma/docker_images/prod, replicated 1 / 1. It has one task running on node 'agata-analysis-6' with ID 'tux501nqeal1rbftt0mlzljn'.
- replay_local_processor**: gitlab-registry.in2p3.fr/ip2igamma/docker_images/prod, replicated 11 / 11. It has 11 tasks running across various nodes (agata-analysis-6, -7, -8) with IDs like '1wdm1b4etayna2zv6stwyvuv', '2apgy1xulpkdtkfyywyfgo', etc.

It's running !

Tests on analysis cluster @ Legnaro

- Data from /opt/data/LNL/EXP_049/AgaPrep/run_1004_LLP_60Co/ (29 crystals)
- Read .cdat files
- PreProcessingFilter
- PSAFilter
- Write .adf files



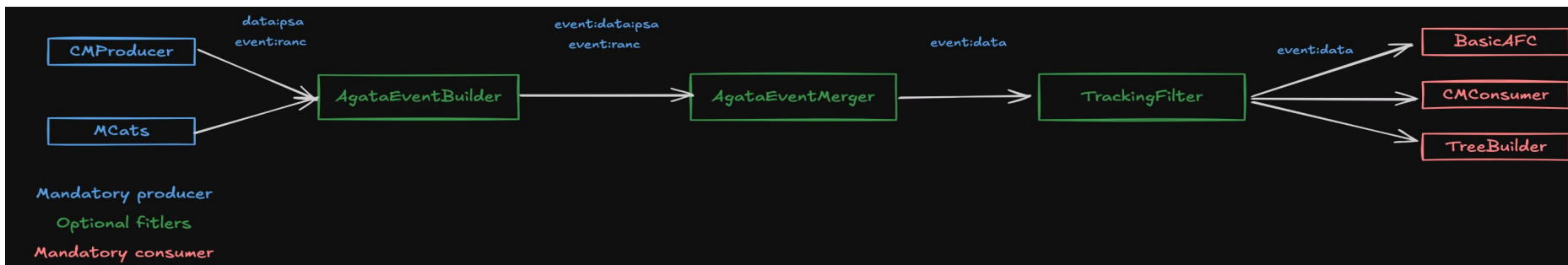
	Elapsed Time (s)	%
Femul old-analysis	8050	
Femul new-analysis	1718	-78,66
Femul.exe new-analysis	1378	-82,88
DAEmule new cluster	586	-92,72

Global level processing

- New actors added to Agapro to manage global level :
 - *Multi-Channels Agata Time Sorter* (MCats) : N entries from Central Memory
→ One sorted output
 - *AgataEventBuilder* : from data:psa to event:data:psa
 - *AgataEventMerger* : from event:data:psa + event:ranc to event:data

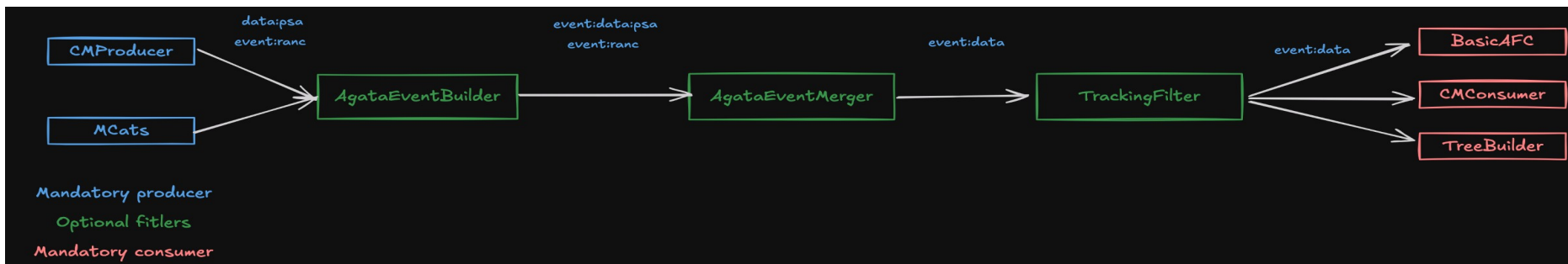
Global level processing

- New actors added to Agapro to manage global level :
 - *Multi-Channels Agata Time Sorter* (MCats) : N entries from Central Memory
→ One sorted output
 - *AgataEventBuilder* : from data:psa to event:data:psa
 - *AgataEventMerger* : from event:data:psa + event:ranc to event:data
- Different topology



Global level processing

- New actors added to Agapro to manage global level :
 - *Multi-Channels Agata Time Sorter* (MCats) : N entries from Central Memory
→ One sorted output
 - *AgataEventBuilder* : from data:psa to event:data:psa
 - *AgataEventMerger* : from event:data:psa + event:ranc to event:data
- Different topology

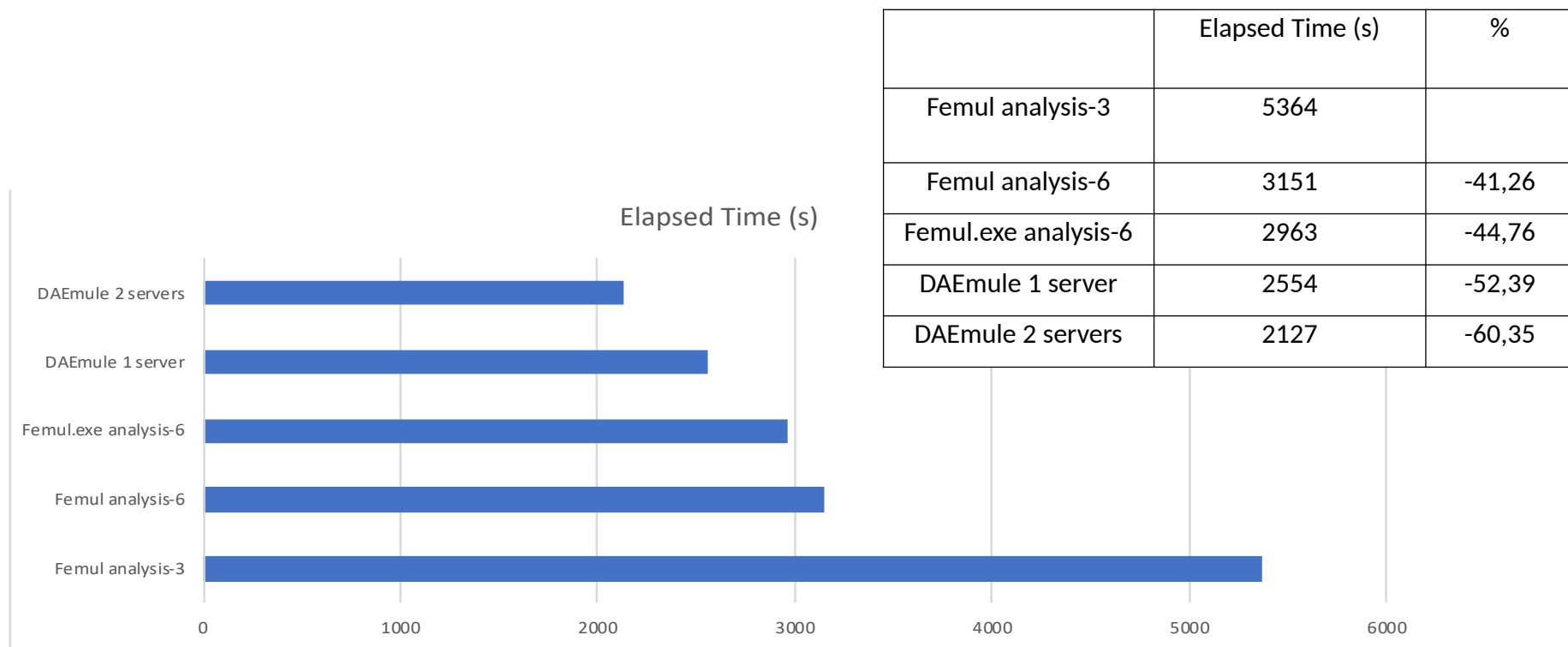


CMProducer and CMConsumer are used to split the processing on 2 servers, for example :

- Mcats + EventBuilder + EventMerger + CMConsumer on server 1
- CMProducer + TrackingFilter + TreeBuilder on server 2

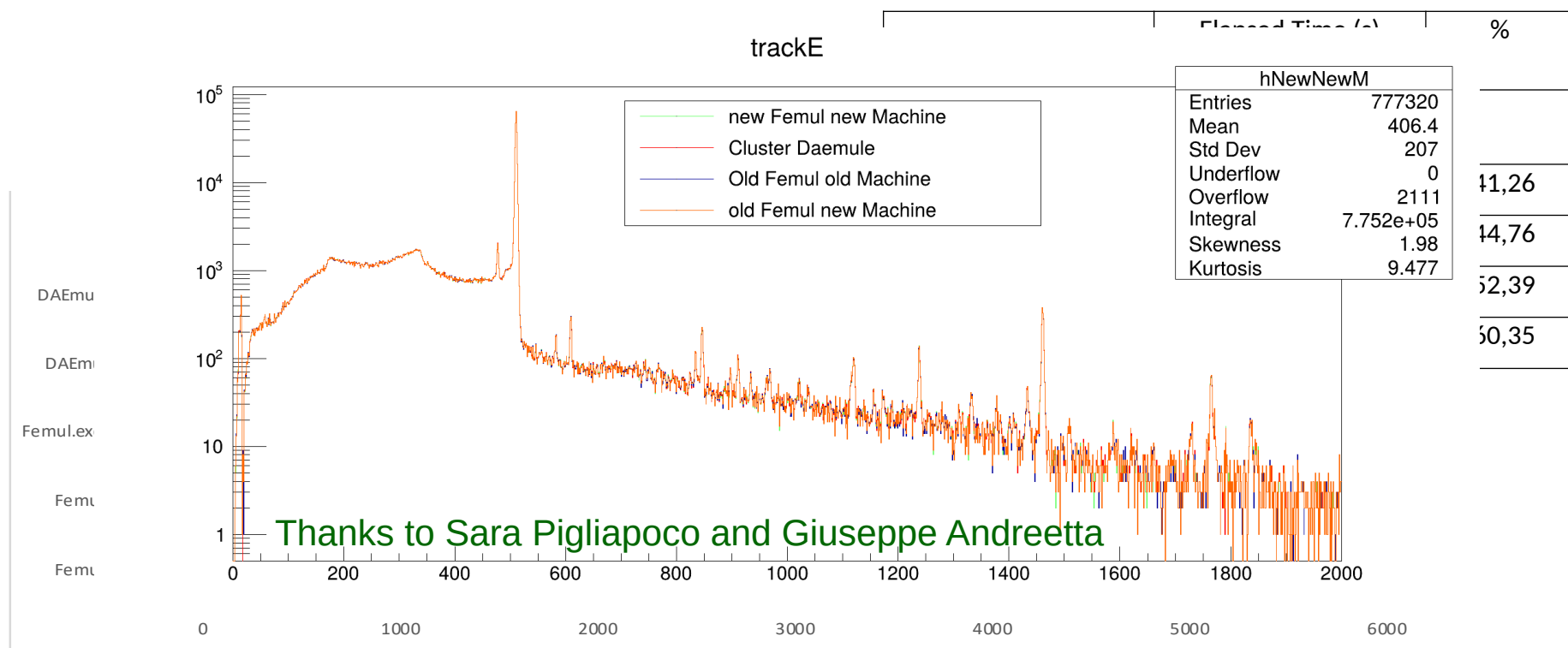
Tests on analysis cluster @Legnaro

- Data from /opt/data/LNL/EXP_049/Replay/run_0078/ (29 crystals)



Tests on analysis cluster @Legnaro

- Data from /opt/data/LNL/EXP_049/Replay/run_0078/ (29 crystals)



Conclusion

- First version of an Agapro emulator able to run on clusters
- Tests at Legnaro on both local and global level processing are fine
- Need to improve integration with Docker Swarm and Portainer.io to ease usage
- Some documentation at
<https://gbaulieu.pages.in2p3.fr/handbook-dev/binaries/DAAEmule/>
- Need to organize a first tutorial session : goal is to be ready for next experiment!