A data acquisition system for the Cerenkov Telescope Array

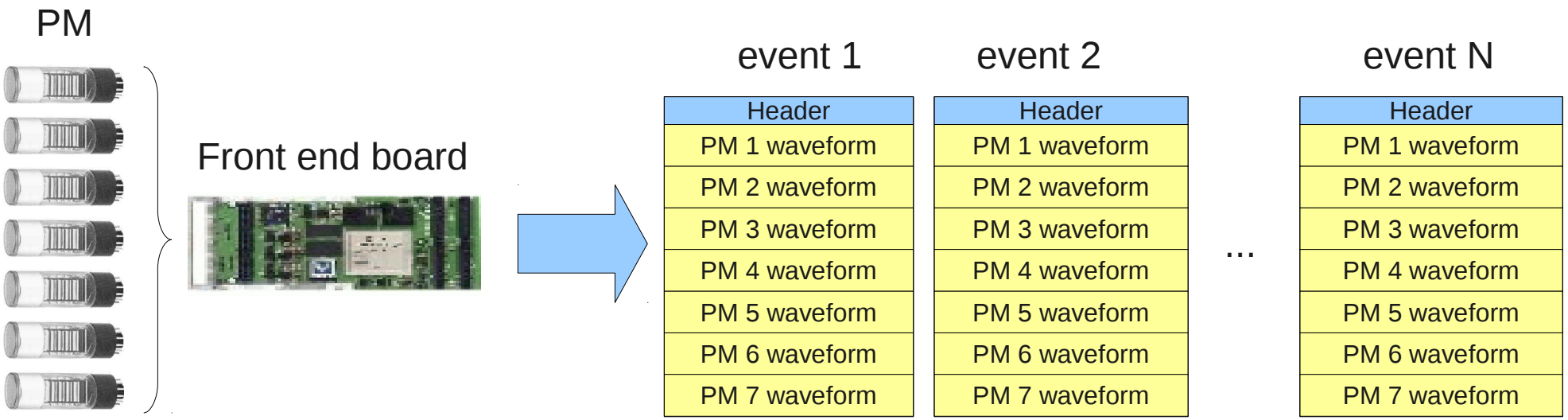
Dirk HOFFMANN
CPPM/IN2P3/CNRS

Julien HOULES
CPPM/IN2P3/CNRS

Contact :
houles@cppm.in2p3.fr

Hypothesis

Camera data flow



Whole Camera ~ 2100 PM -> 300 front end boards

Data flow hypothesis

- ~ 2100 pixels camera

- L1 trigger rate : 10 kHz
- Size of waveform for 1 PM : 144 bytes (16 bit * 72 samples)
- All the L1 events are sent

→ Max theorical bandwidth = 10000 * 2100 * 144 = 3 GB/s
= 24 Gb/s

- 7 detectors for each front end board : 300 boards/camera

→ Each board generates a flow of 3000/300 = 10 MB/s
= 80 Mb/s

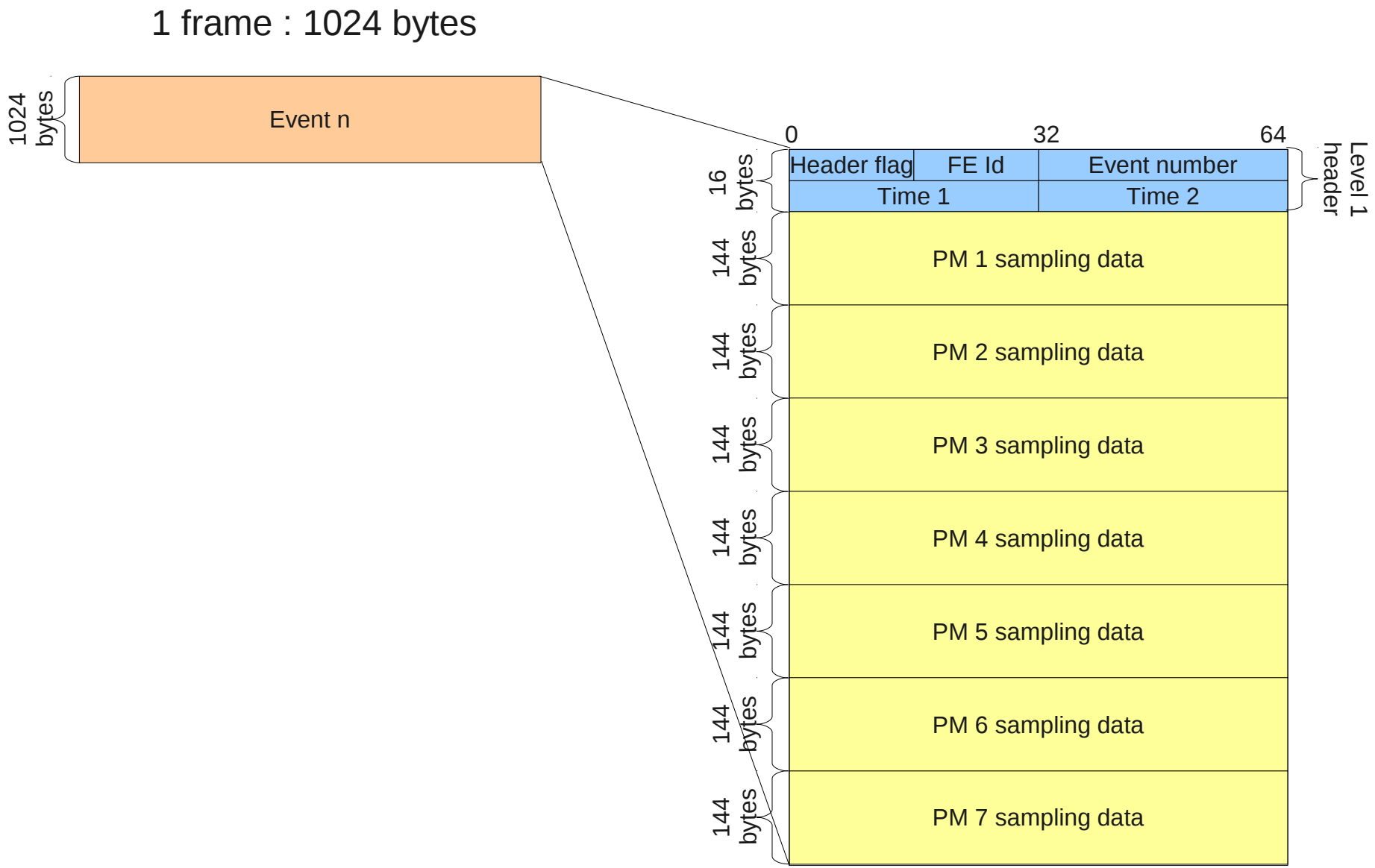
https://martwiki.in2p3.fr/twiki/bin/view/CTA/DataAcquisition

Data format hypothesis

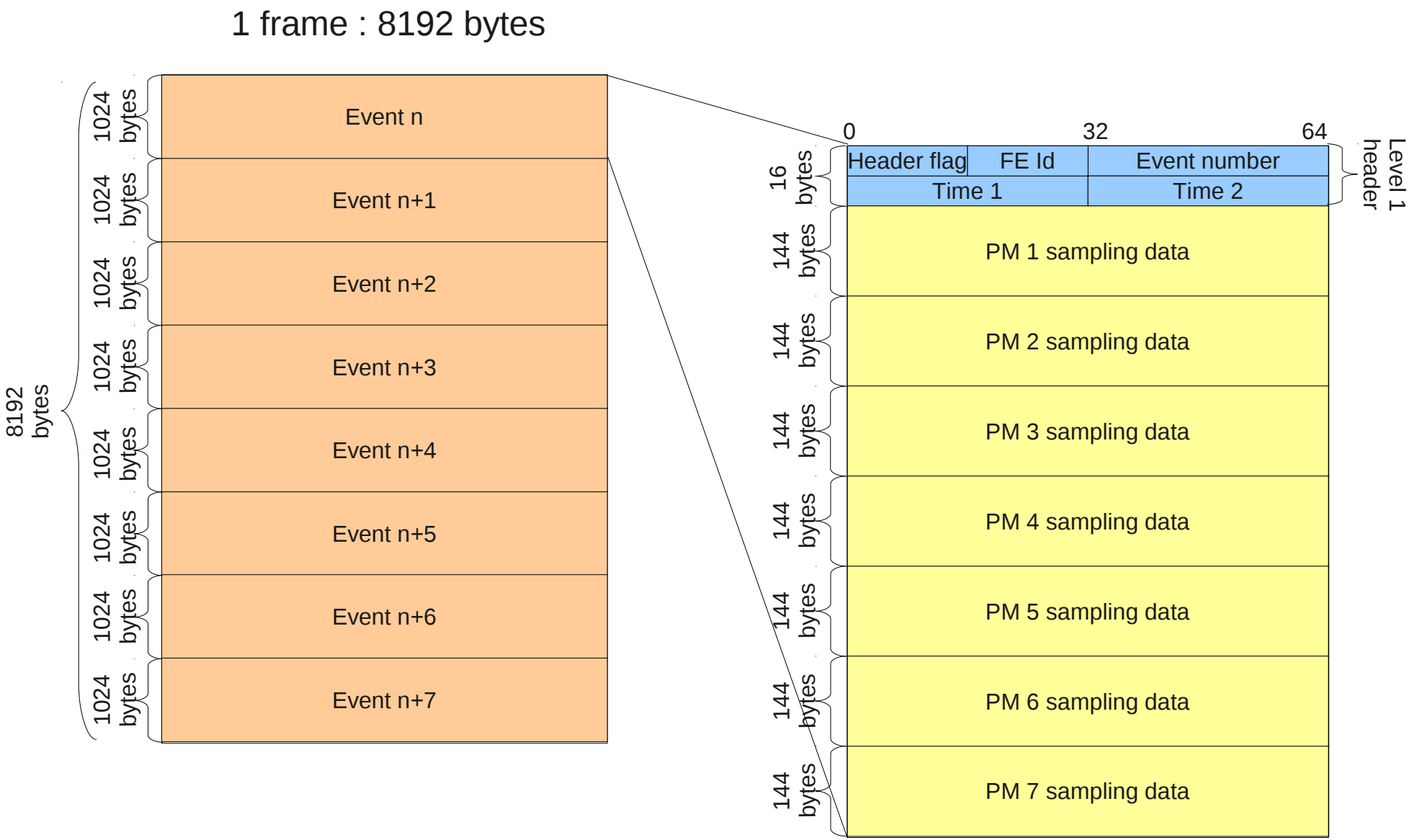
The front end electronics send data :

- through Ethernet links
- Using UDP protocol

Data format hypothesis: regular frame

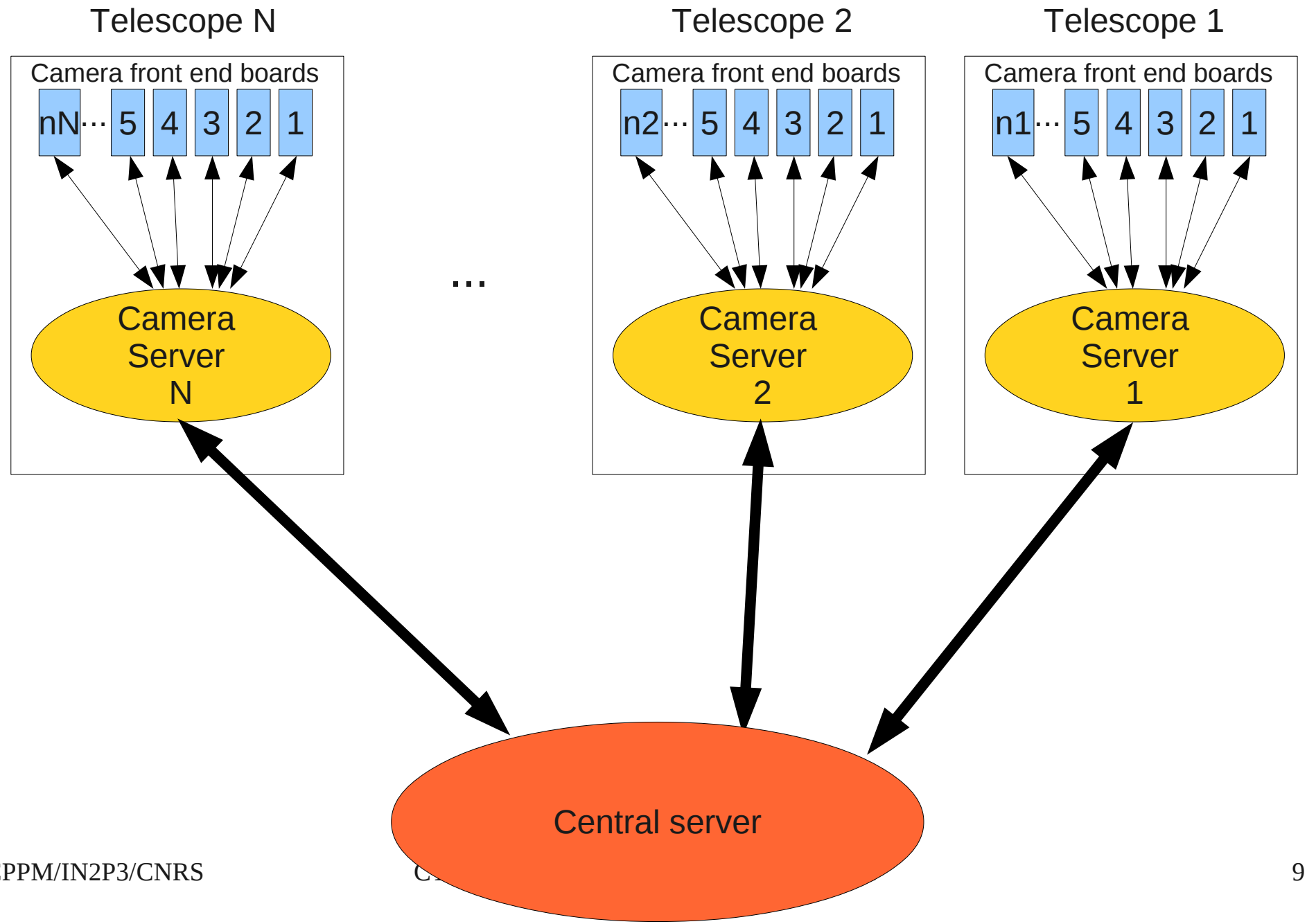


Data format hypothesis : jumbo frame



Camera server

Global architecture



Camera server skills

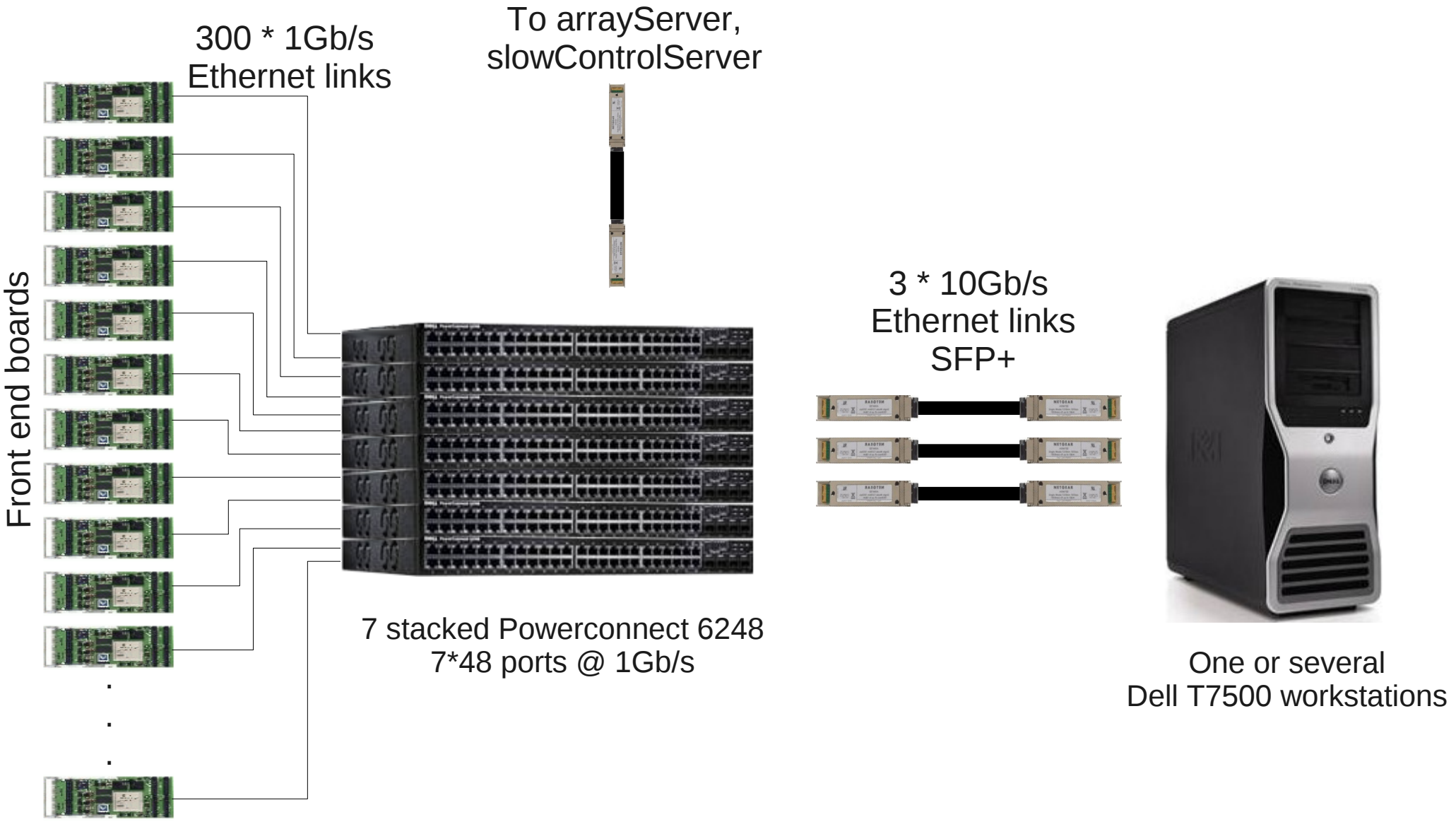
- Centralizes data from front end electronics
- Builds events
- Reduces data (soft trigger, compression...)
- Sends data to central server (array level)

Camera server requirements

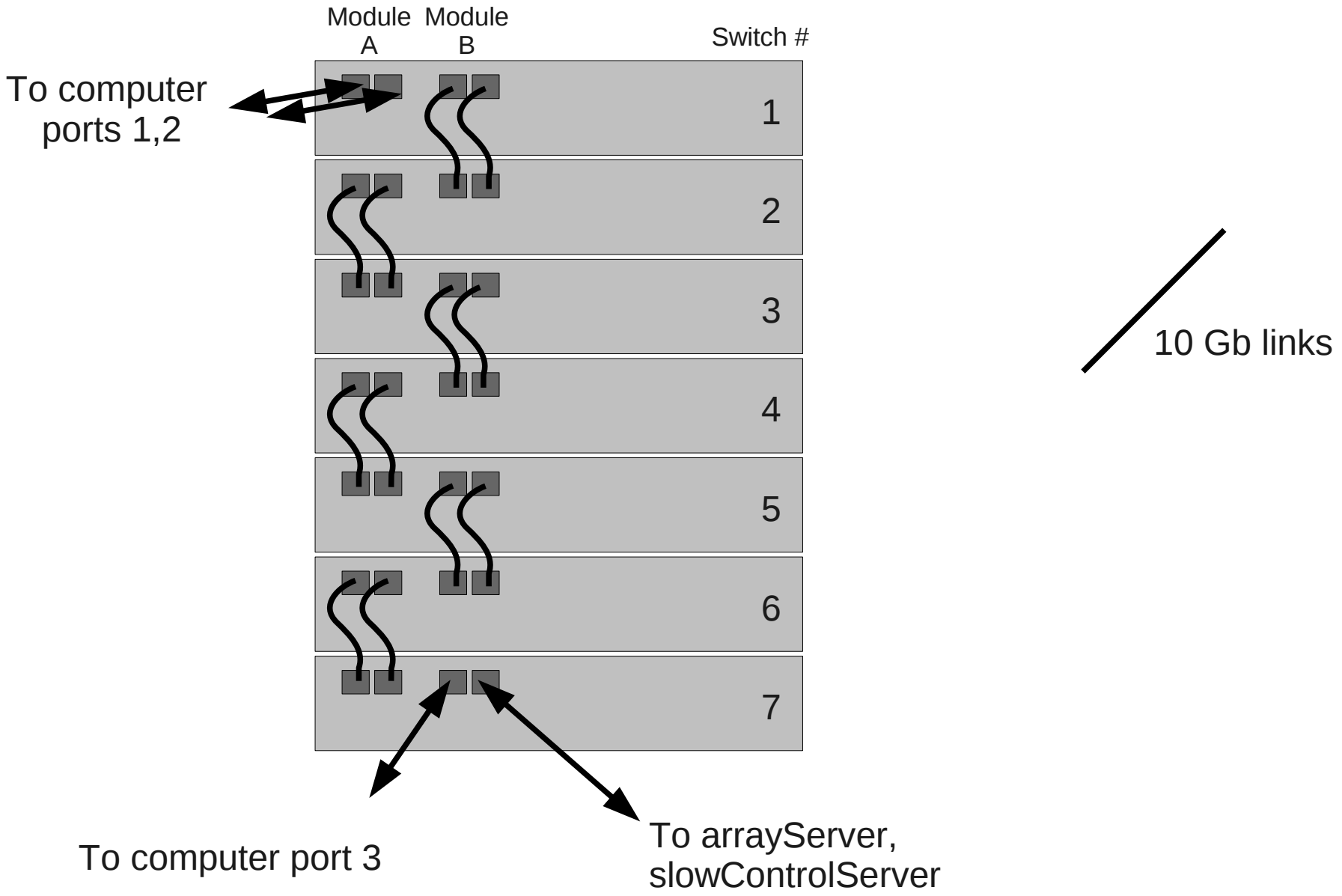
- 300 Ethernet inputs
- At least 80 Mb/s per input
- Jumbo frames support
- Able to process a 24 Gb/s data rate

- Direct access to front end electronics for slow control
- 10 Gb downlink to central server (less for uplink)

Camera server infrastructure

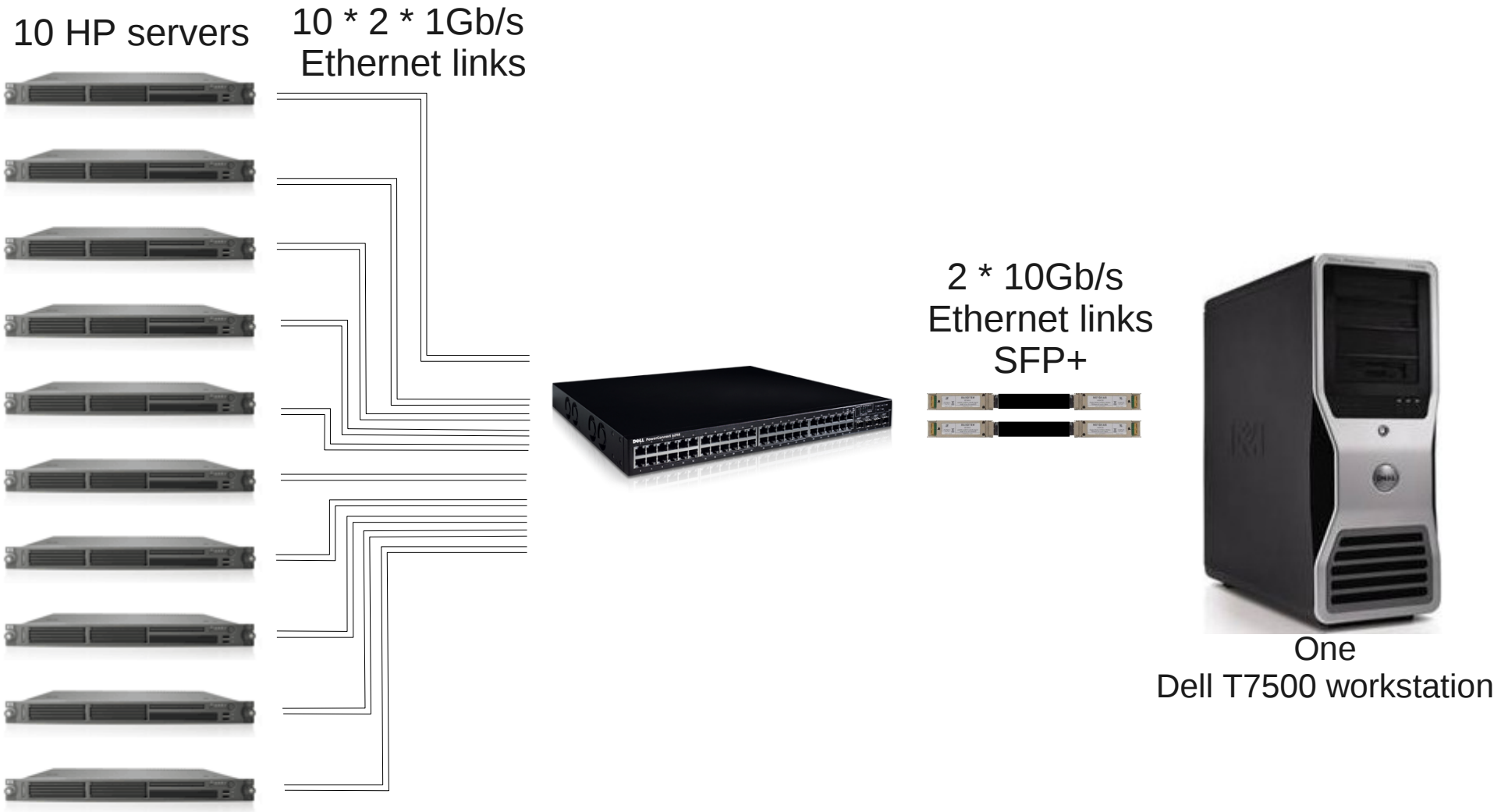


Switches stacking



Test configuration

Test configuration



Stimulation nodes

- 10 servers running a standard Linux distribution
- two 1 Gb network adapters per server
- Each server runs a stimulation software which sends data through 15 logical ports per network adapter, synchronized by a timer

10 servers * 2 adapters * 15 ports = 300 nodes

This system simulates data sent from 300 front end electronics through 20 physical ports

Software

Why a prototype ?

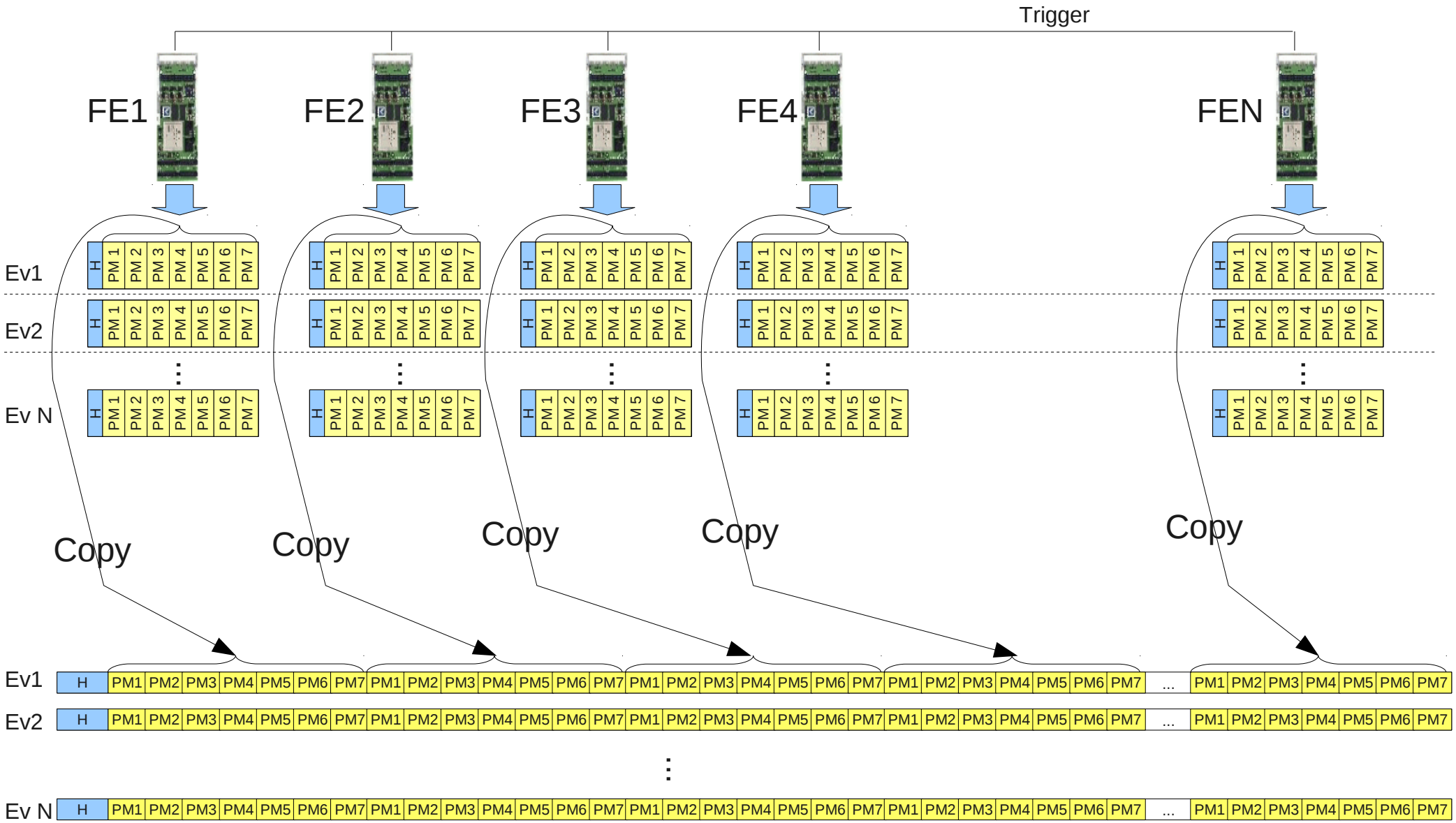
We need a prototype :

- To evaluate the maximum speed reachable
- To test several technologies
- To validate different approaches of the data processing
- To adapt scientific requirements to what we can do
- To identify bottlenecks

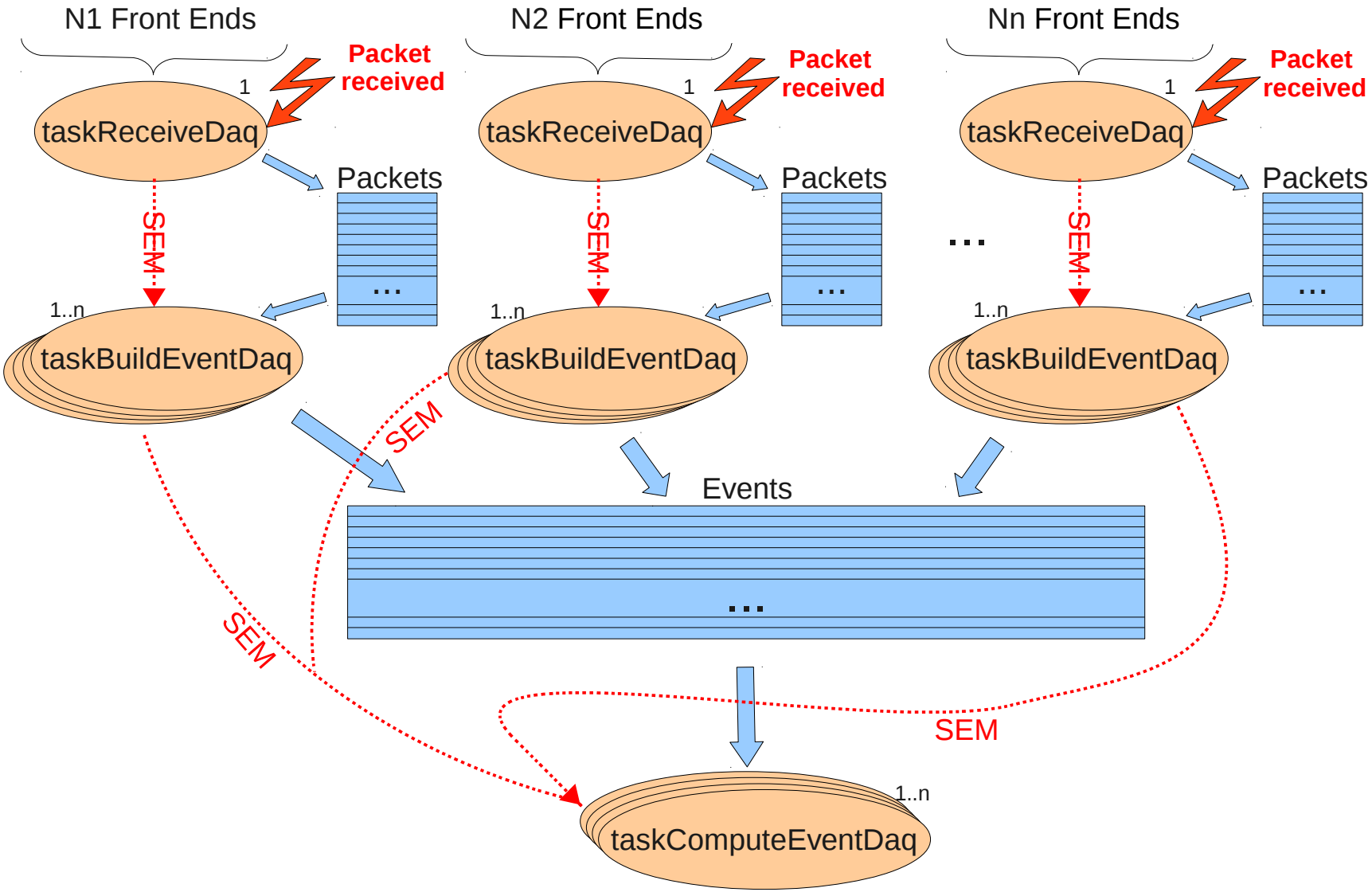
Our first approach

- High modularity to make adaptation to different front end electronics easier
- Multitask approach to divide the flow processing if needed
- Use of a standard Linux distribution but take control on scheduling, memory allocation and fine tuning of the OS
- Constrained electronics to reach the best performances (in a first time)

Event builder



Software overview : 1st architecture

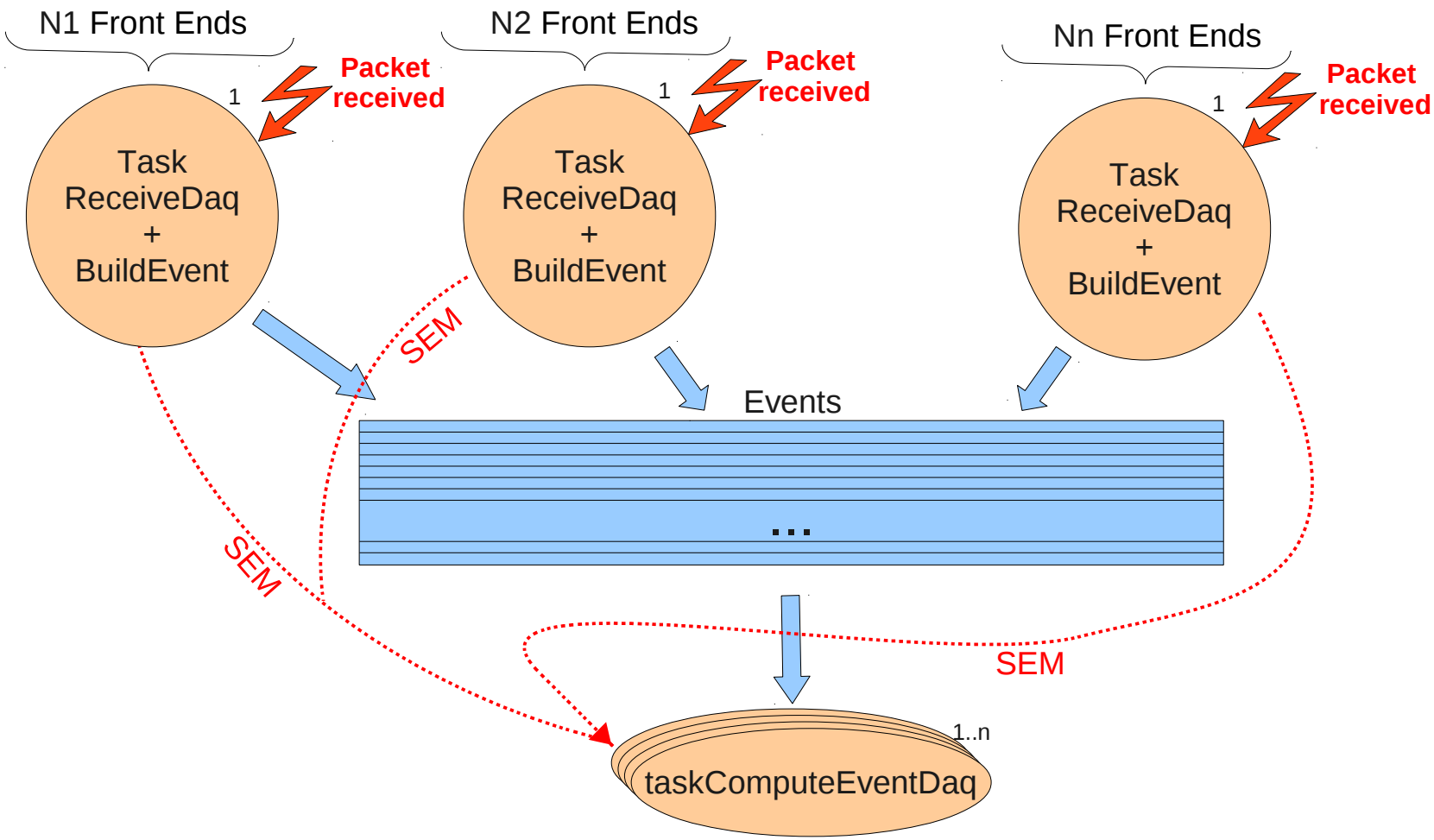


1st architecture measurements

1st Achitecture : Online Event building speed	
Jumbo frames (8192 bytes) :	Regular frames (1024 bytes) :
19,2 Gb/s (2,4 GB/s) with no loss	6,5 Gb/s (0,82 GB/s) with no loss
CPU usage : 300 % (3 cores/12)	CPU usage : 300 % (3 cores/12)
~ 8000 events/s reconstructed	~ 2700 events/s reconstructed

- 300 stimulation nodes generate data
- Incoming data through two 10 Gb adapters
- Two « channels » used
- CPU load spread on 4 cores (on the same multiprocessor)

Software overview : 2nd architecture

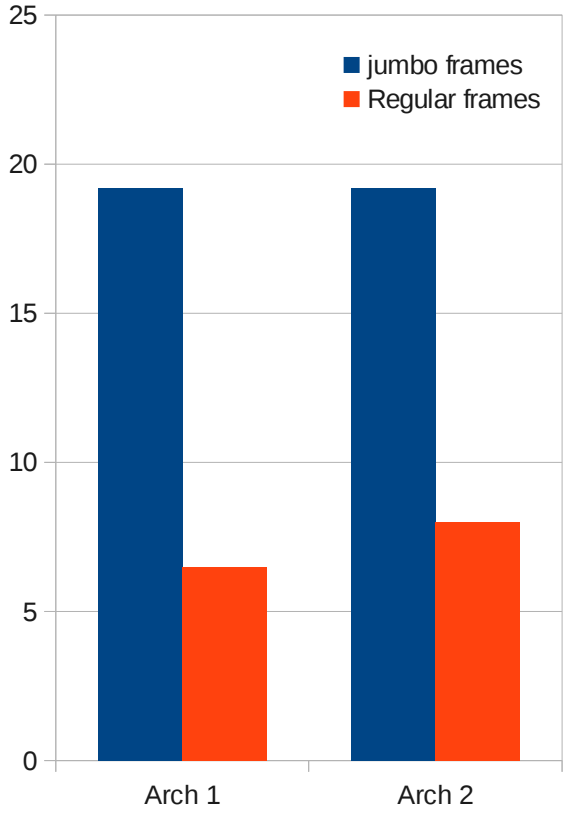


2nd architecture measurements

2nd Achitecture : Online Event building speed	
Jumbo frames (8192 bytes) :	Regular frames (1024 bytes) :
19,2 Gb/s (2,4 GB/s) with no loss	8 Gb/s (1 GB/s) with no loss
CPU usage : 160 % (1.6 cores/12)	CPU usage : 170 % (1.7 cores/12)
~ 8000 events/s reconstructed	~ 3300 events/s reconstructed

- 300 stimulation nodes generate data
- Incoming data through two 10 Gb adapters
- Two « channels » used
- CPU load spread on 2 cores (on the same multiprocessor)

The regular frames problem



- Better results with jumbo frames
- But must keep on being compatible with any elctronics

  → Must improve smaller packets reception

- Replace very old network software achitectures

  → Solutions exist and are being tested :

  Promising results

Conclusion

Conclusion